EXPERT INSIGHT

Mastering Active Directory

Design, deploy, and protect Active Directory Domain Services for Windows Server 2022

Third Edition

Dishan Francis

Packt

Mastering Active Directory

Third Edition

Design, deploy, and protect Active Directory Domain Services for Windows Server 2022

Dishan Francis

BIRMINGHAM—MUMBAI

Mastering Active Directory

Third Edition

Producer: Tushar Gupta

Acquisition Editor – Peer Reviews: Saby Dsilva

Project Editor: Namrata Katare

Content Development Editor: Alex Patterson

Copy Editor: Safis Editor

Technical Editor: Aditya Sawant

Proofreader: Safis Editor

Indexer: Tejal Daruwale Soni

Presentation Designer: Ganesh Bhadwalkar

First published: June 2017
Second Edition: August 2019
Third Edition: November 2021

Production reference: 1261121
Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham
B3 2PB, UK.

ISBN 978-1-80107-039-3

www.packt.com

Contributors

About the author

Dishan Francis is an IT professional with over 15 years' industry experience. He was a six-time Microsoft MVP in Enterprise Mobility before joining Microsoft UK as a security consultant. He has maintained the RebelAdmin technology blog over the years, with lots of useful articles that focus on on-prem Active Directory services and Azure Active Directory. He has also written for other Microsoft managed blogs such as canitpro and ITOpsTalk.

It would have been impossible to write this book without many people behind me. I'd like to thank my wife Kanchana, daughter Selena, and son Andrew for their great support, and my parents and all my relations for their encouragement. Thanks too to my publisher, reviewers, and my friends in Microsoft for their support on this journey.

About the reviewer

Chris Spanougakis is a Microsoft Certified Trainer since 2000 and holds a Master of Science in computer studies. He has participated in various Microsoft local and international events, such as Teched Europe, Microsoft Ignite, Microsoft Sinergija, IT Pro|Dev Connections, and ShowIT as a technology speaker, and he is specialized in Microsoft products, such as Windows Server, Exchange Server, System Center products, etc. He was also a Microsoft **Most Valuable Professional** (**MVP**) in Identity and Access Management from 2008 to 2019. Chris has more than 20 years of experience as an IT consultant, specialized in the implementation of Microsoft technologies in organizations of any size. Today he works as an Azure solution architect and trainer in Greece and abroad. You can reach him at `https://systemplus.gr`

Table of Contents

Preface

Microsoft Active Directory is the most widely used identity management solution. It can centrally manage identities across its infrastructure. It is equipped with different role services, features, and components that help us handle identities securely and effectively according to business requirements. For the last 20 years, Microsoft has continued improving Active Directory, and Active Directory 2022 further consolidates its approach in terms of rectifying industry requirements and protecting identity infrastructures from emerging security threats. However, a technology-rich product is not simply going to make a productive, reliable, scalable, and secure identity infrastructure. It requires knowledge of Active Directory roles services, components, and features. It also requires knowledge of how to use those effectively to match different operational requirements. Only then can we plan, design, manage, and maintain a robust identity infrastructure. Over the past few years, more and more organizations have adopted cloud technologies for a variety of reasons. With the growth of the cloud footprint, organizations' identity requirements have also changed. We can no longer limit corporate identities to on-prem infrastructures. By using Microsoft Azure Active Directory, we can extend our on-prem identities to the cloud. The hybrid AD approach provides lots of benefits for modern authentication requirements. However, security-wise, it also opens up a whole new level of challenges. Therefore, the majority of new content in the third edition is related to designing the Azure AD hybrid cloud, securing a hybrid AD environment, and protecting sensitive data.

Who this book is for

If you are an Active Directory administrator, system administrator, or network professional who has basic knowledge of Active Directory and is looking to become an expert in this topic, this book is for you.

What this book covers

Chapter 1, *Active Directory Fundamentals*, explains what Active Directory is and its capabilities. This chapter also explains the main components (physical and logical structure), object types, and role services of Active Directory. Last but not least, this chapter also covers why we need an advanced identity management solution such as Azure Active Directory.

Chapter 2, *Active Directory Domain Services 2022*, explains what we can expect with **Active Directory Domain Services** (**AD DS**) 2022 and how we can use the features introduced in AD DS 2016 (as there is no new **Domain Functional Level** (**DFL**) or **Forest Functional Level** (**FFL**)) to improve your existing Active Directory environment.

Chapter 3, *Designing an Active Directory Infrastructure*, talks about what needs to be considered in Active Directory infrastructure design. This chapter discusses how to place the AD DS logical and physical components in the AD DS environment according to best practices. It also covers the design concepts for hybrid identity.

Chapter 4, *Active Directory Domain Name System*, explains how DNS works with AD DS. This chapter also includes information about the DNS server component, different types of DNS records, zones, DNS delegation, and DNS policies.

Chapter 5, *Placing Operations Master Roles*, talks about the **Flexible Single Master Operations** (**FSMO**) roles and their responsibilities. This chapter also describes things we need to consider when placing FSMO roles in an Active Directory environment.

Chapter 6, *Migrating to Active Directory 2022*, covers the different AD DS deployment models. This chapter also provides a step-by-step guide to migrating from an older version of AD DS to AD DS 2022.

Chapter 7, *Managing Active Directory Objects*, discusses how to create objects, find objects, modify objects, and remove objects (small-scale and large-scale) by using built-in Active Directory management tools and PowerShell commands.

Chapter 8, *Managing Users, Groups, and Devices*, further explores the Active Directory objects by deep diving into attributes, managed service accounts, and management of different object types. Last but not least, you will also learn how to sync custom attributes to Azure Active Directory.

Chapter 9, *Designing the OU Structure*, teaches you how to design the **organizational unit** (**OU**) structure properly, using different models to suit business requirements. This chapter also describes how to create, update, and remove OUs. Furthermore, this chapter also discusses how we can delegate AD administration by using OUs.

Chapter 10, *Managing Group Policies*, mainly discusses Group Policy objects and their capabilities. Group Policy processing in an AD environment depends on many different things. In this chapter, we will deep dive into group policy processing to understand the technology behind it. We are also going to look into the different methods we can use for group policy filtering. Last but not least, we will learn about most commonly use group policies.

Chapter 11, *Active Directory Services – Part 01*, walks us through the more advanced Active Directory topics, such as AD **Lightweight Directory Services** (**LDS**), Active Directory replication, and Active Directory sites.

Chapter 12, *Active Directory Services – Part 02*, sees you learn about Active Directory trusts in detail. This chapter also covers topics such as Active Directory database maintenance, **Read-Only Domain Controller** (**RODC**), AD DS backup, and recovery.

Chapter 13, *Active Directory Certificate Services*, discusses the planning, deployment, and maintenance of Active Directory Certificate Services. Furthermore, we will also learn how signing, encryption, and decryption work in a **public key infrastructure** (**PKI**).

Chapter 14, *Active Directory Federation Services*, focuses on **Active Directory Federation Services** (**AD FS**) such as planning, designing, deployment, and maintenance. This chapter also covers new features of AD FS, such as built-in Azure MFA support. At the end you will also learn how to establish a federated connection with Azure AD.

Chapter 15, *Active Directory Rights Management Services*, covers the **Active Directory Rights Management Service** (**AD RMS**) role, which we can use to protect sensitive data in a business. Data is the new oil, and the value of data keeps increasing. Therefore, protection of data is important for every business. In this chapter, we will learn how AD RMS works and how to configure it.

Chapter 16, *Active Directory Security Best Practices*, covers the protection of the Active Directory environment. Recent attacks and studies prove that adversaries are increasingly targeting identities. So, we need to be mindful of protecting our Active Directory infrastructure at any cost. In this chapter, we will learn about different tools, services, and methods we can use to protect the Active Directory environment such as Secure LDAP, Microsoft LAPS, delegated permissions, restricted RDP, and Azure AD password protection.

Chapter 17, *Advanced AD Management with PowerShell*, is full of PowerShell scripts that can be used to manage, secure, and audit an Active Directory environment. We will also learn about the Azure Active Directory PowerShell for Graph module, which we can use to manage, query, and update AD objects in a hybrid AD environment.

Chapter 18, *Hybrid Identity*, discusses how we can extend our on-prem AD DS infrastructure to Azure Active Directory. Before we work on the implementation, we will deep dive into the planning process of the Azure AD hybrid setup. In this chapter, we will also learn about different Azure AD connects deployment models, Azure AD cloud sync, Secure LDAP, and replica sets.

Chapter 19, *Active Directory Audit and Monitoring*, teaches you how to monitor your on-prem/hybrid AD DS infrastructure using different tools and methods (cloud based and on-prem). This chapter also demonstrates how to audit the health of an Active Directory environment.

Chapter 20, *Active Directory Troubleshooting*, discusses how to troubleshoot the most common Active Directory infrastructure issues using different tools and methods. Furthermore, we will also look into the most common Azure AD Connect errors, which can have a direct impact on the health of the Azure AD hybrid environment. You can find this chapter available online at: `https://static.packt-cdn.com/downloads/9781801070393_Chapter_20.pdf`

Appendix A, *References*, covers the *Further reading* section chapter wise. It's freely available online for our readers and here is the link: `https://static.packt-cdn.com/downloads/Mastering_Active_Directory_References.pdf`.

To get the most out of this book

This book is ideal for IT professionals, system engineers, and administrators who have a basic knowledge of Active Directory Domain Services. A basic knowledge of PowerShell is also required, since most of the role deployment, configuration, and management is done by using PowerShell commands and scripts.

Download the example code files

The code bundle for the book is hosted on GitHub at `https://github.com/PacktPublishing/Mastering-Active-Directory-Third-Edition`. We also have other code bundles from our rich catalog of books and videos available at `https://github.com/PacktPublishing/`. Check them out!

Download the color images

We also provide a PDF file that has color images of the screenshots/diagrams used in this book. You can download it here: `https://static.packt-cdn.com/downloads/9781801070393_ColorImages.pdf`.

Conventions used

There are a number of text conventions used throughout this book.

`CodeInText`: Indicates code words in the text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles. Here is an example: "SLDs are domain names that don't have DNS suffixes such as `.com`, `.org`, or `.net`."

Any command-line input or output is written as follows:

```
Get-ADDomain | fl Name,DomainMode
```

Bold: Indicates a new term, an important word, or words that you see on screen. For example, words in menus or dialog boxes appear in the text like this. Here is an example: "Go to **All Services** | **Azure AD Domain Services**."

Warnings or important notes appear like this.

Tips and tricks appear like this.

Get in touch

Feedback from our readers is always welcome.

General feedback: If you have questions about any aspect of this book, mention the book title in the subject of your message and email us at `customercare@packtpub.com`.

Errata: Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you have found a mistake in this book, we would be grateful if you would report this to us. Please visit `www.packt.com/submit-errata`, selecting your book, clicking on the Errata Submission Form link, and entering the details.

Piracy: If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at `copyright@packt.com` with a link to the material.

If you are interested in becoming an author: If there is a topic that you have expertise in, and you are interested in either writing or contributing to a book, please visit `authors.packtpub.com`.

Share your thoughts

Once you've read *Mastering Active Directory, Third Edition*, we'd love to hear your thoughts! `Please click here to go straight to the Amazon review page` for this book and share your feedback.

Your review is important to us and the tech community and will help us make sure we're delivering excellent quality content.

1

Active Directory Fundamentals

"Despite all this rapid change in the computing industry, we are still at the beginning of the digital revolution."

- Satya Nadella

It has been two years since the release of the second edition of this book, *Mastering Active Directory*. First of all, I would like to thank all my readers for their valuable feedback, which encouraged me to write this third edition. I am sure that you will all benefit from the additional content that has been added to this new edition.

We are going to start this book by refreshing our knowledge of the fundamentals of Windows Active Directory. The main topics covered in this chapter are as follows:

- Modern access management
- The future of access management
- The role of Active Directory in hybrid identity
- Benefits of using Active Directory
- Understanding Active Directory components
- Understanding Active Directory objects

To start with, let's talk about how the pandemic and other factors have shaped modern access management.

Modern access management

The Covid-19 pandemic has heightened our sense of uncertainty as humans over physical and mental health, economy, family, society, and work. Most of us have experienced long-lasting effects on our lives that we never envisioned. Some of these profound effects may drag our lives backward or forward by years. A paradigm shift hastened by the pandemic is the accelerated digital transformation of society. The lockdown rules and increased demand for secure remote work has pushed some "offline" businesses and industries into the "online" realm sooner than we thought. My nine-year-old daughter is having her piano lessons via Zoom meetings now. I never thought it was practical to learn an instrument "online" but I was proven wrong. At the beginning of the pandemic, the financial sector wasn't ready to embrace the working from home culture. But a recent survey carried out by Deloitte confirms almost three quarters (70%) of employees working in financial services rate their working from home experience as positive. (Source: `https://bit.ly/3CSjC08`) Twilio is a leading cloud communications and customer engagement platform. They recently surveyed over 2,500 enterprise decision makers in the United States, the United Kingdom, Germany, Australia, France, Spain, Italy, Japan, and Singapore to evaluate their views on digital transformation as a result of Covid-19. According to the survey results, "97% of enterprise decision makers believe the pandemic sped up their company's digital transformation." (Source: `https://bit.ly/2ZSnTlK`.) McKinsey & Company is an American worldwide management consulting firm and they recently did a survey using 900 C-level executives and senior managers representing the full range of regions, industries, company sizes, and functional specialties. According to the study, respondents confirmed their companies acted 20 to 25 times faster than expected in implementing digital transformation strategies. When it comes to remote working, companies moved 40 times faster.

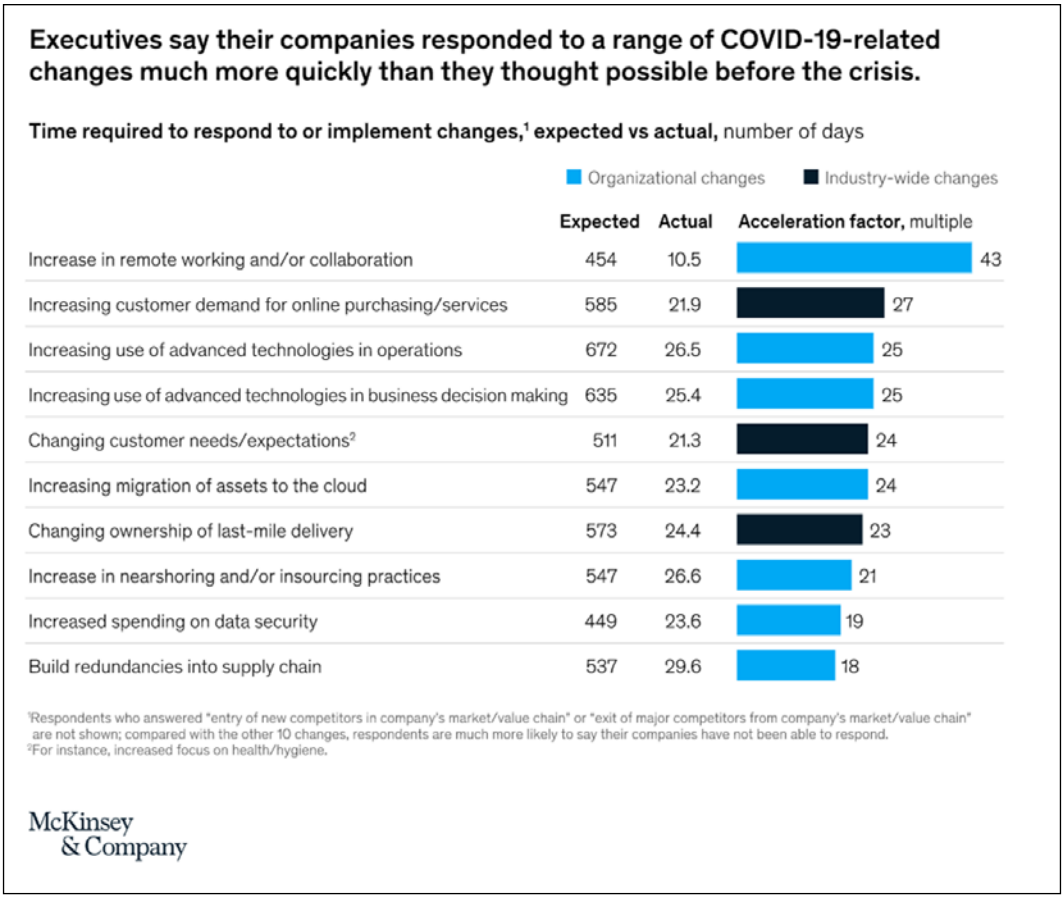


Figure 1.1: Speed of responses to pandemic challenges

Source: `https://mck.co/2Ykj9Fd`

With the rise of digital transformation, working from home has become the new normal. Businesses have had to implement applications, services, and collaboration tools that allow remote workers to carry out their day-to-day tasks seamlessly. On this journey, the hurdle wasn't the investment or the technology.

It was the "time." This was the same for many businesses when adopting this ubiquitous nature of operations. When "time" starts to cost us money or when "time" starts to affect sales, the manufacturing process, supplies, or workforce productivity, we do not have time to evaluate all the pros and cons. We do not have time to do all the ground work. We will have to take risks. We will have to bend the rules. When we rush things, as humans, we tend to make mistakes. Some of these mistakes opened up opportunities for cyber criminals throughout 2020:

- According to iomart, large-scale data breaches increased 273% in the first quarter of 2020. (Source: `https://bit.ly/3mNckFB`)
- Data from the UK's **Information Commissioner's Office** (**ICO**) confirms 90% of cyber data breaches were caused by user errors. (Source: `https://bit.ly/3whpgGV`)
- According to RiskBased Security's 2020 Q3 report (Source: `https://bit.ly/3mMxjbu`), healthcare (11.5%) and IT (10.3%) are the two industries that reported the most data breaches. Also, we know these industries have been the most active industries during the pandemic.
- The report also says, when we consider all of the breaches in 2020, that 29% of those exposed passwords, 36% exposed email addresses, and 45% exposed names.

If we summarize the above findings, we can see there was a massive increase in data breaches in 2020 and the majority of those breaches were due to human error. The healthcare and IT industries have been the top targets financially motivating cyber criminals in 2020. The above data also confirms cyber criminals are mainly after **identities**.

Identity is the new perimeter. The perimeter defense model is no longer valid against modern identity threats. Identity and access management is the cornerstone of digital transformation. A study done by Ping Identity says 90% of IT decision makers believe identity and access management is the key enabler of digital transformation (Source: `https://bit.ly/3BNw0gS`). Identity and access management solutions depend on directory services such as Windows Active Directory to store/retrieve data relating to user identities. Windows Active Directory was first released on February 17, 2000, and for 21 years it has been helping organizations to manage identities. But now we have a new set of challenges.

According to FireEye's Cyber Security Predictions 2021 report (`https://bit.ly/3nZBpfQ`), about 95% of companies have some type of cloud presence.

So the questions are:

1. How can we allow users to use the same Active Directory user accounts to access cloud resources?
2. How can we enable a **single sign-on** (**SSO**) experience for cloud-based applications?
3. How can we protect identities when they start to appear in cloud and unsecured networks?
4. How can we maintain compliance when we start using cloud resources?
5. How can we detect/handle a potential breach?

To address the above questions, we need a distributed, highly available identity and access management solution such as Azure Active Directory. It doesn't mean Azure Active Directory is a replacement for Windows Active Directory. These are two different products with many different characteristics. But these two solutions can work together to address access and security challenges in both worlds (On-prem and the cloud). In this edition, you will find many topics and a lot of content related to **hybrid identity**. Also, throughout the book, content will be positioned to accentuate the importance of identity protection.

What is an Identity?

Elephants are truly fascinating creatures. Female elephants stay in their herd for life. When a baby elephant is born, young female elephants in the herd will help the mother to take care of the baby. A baby elephant usually weighs about 250 pounds and is three feet tall. In the beginning, a baby elephant can't see clearly. But it can identify its mother among the other young female elephants by touch, smell, and sound. Social insects such as ants recognize various castes in their colony based on "ant body odor." The same method is also used to recognize ants from other colonies.

When it comes to humans, we use many different ways to uniquely identify a person. In day-to-day life, we recognize people based on their name, face, voice, smell, body language, uniforms, and so on. The uniqueness of individuals describes an "identity." However, if we need to prove our identity, we need to use formal methods of identification such as a passport, driving license, and residence card. These formal methods are well recognized by many authorities. So far, we have talked about physical identity. But how can we bring this to the digital world? To do that, we need our digital identity to represent our physical identity.

As an example, when I registered with my GP for the first time, they checked a form of identification and verified my identity. Then they issued me with a unique NHS number; this unique number is the way their computer system will recognize me. When I signed up for my broadband connection, the service provider asked me to set up a unique password. This password will be used to prove my identity when I call them for support next time. Different systems, applications, and services uses different methods to verify someone's digital identity. These systems use databases and directories to store the data related to digital identities.

It is also important to remember a digital identity does not always represent a human. It can represent other entities such as devices, applications, services, groups, and organizations. Digital identities are also becoming more and more dynamic. As an example, your Facebook profile represents a digital identity. It keeps updating based on pictures you upload, posts you share, and friends you make. It is a living identity. A digital identity can get frequently updated based on attributes and access privileges. Nowadays, we can see different systems allow users to use one form of digital identity to get access. As an example, a Microsoft account can be used to access on-prem applications as well as SaaS applications. These federated digital identities provide a better consumer experience. The Active Directory service is capable of managing digital identities as well as federated digital identities.

Before we go and look into Active Directory fundamentals, I think it is better to share some of the identity and access management trends that lie ahead of us in 2021 and see how Active Directory will fit in to the picture.

The future of Identity and Access Management (IAM)

In the previous two sections, I used the words "identity and access management" a few times. What exactly does identity and access management mean? Identity and access management is a solution used to regulate the "access life cycle" of a user within an organization. The main role of it is to make sure the right person has the right access to the right resources for the right reason. Identity and access management solutions mainly have four components.

1. A directory which stores user identity data (directory service)
2. A set of tools to provision, modify, and delete users and privileges
3. A service to regulate access and privileges using policies and workflows
4. A system for auditing and reporting

According to the above definition, Active Directory is not an identity and access management system. But it plays a major role in an identity and access management system. The directory element of an identity and access management system doesn't represent Microsoft Active Directory only, it could be any directory. But we know that the most commonly used directory service on the market is Microsoft Active Directory. The success of an IAM solution depends on all four pillars that I mentioned before. As I explained in the introduction, IAM is the key enabler of digital transformation. So what does the future look like for IAM in 2021 and beyond.

The Rise of Cybercrime

It's been a roller-coaster year for most of us. With the Covid-19 pandemic, uncertainty is all around us. That's changed the future for us in many ways. You may have had to reorganize your priorities and push back some of your plans years. On top of that, we have all had to do a lot to maintain our mental health. Cyber criminals are also humans. So we might think that the pandemic has also struck a blow to their activities. But it seems it hasn't. They seem to have found opportunities even during a pandemic. Instead of a reduction in cybercrime, we have seen a huge increase in the number of incidents. The FBI says it saw a 300% increase in cybercrime in 2020 (Source: `https://bit.ly/3o3uguL`). When it comes to the healthcare industry, we would expect some dignity as it has been a lifeline during the pandemic. But for criminals, it was just another opportunity. Verizon's Data Breach Investigations Report 2020 (`https://vz.to/3CQvPCL`) confirms a 58% increase in data breaches in the healthcare industry and the majority of them were financially motivated attacks. Also, these attacks are getting more sophisticated day by day. The recent **Nobelium** attack is a great example of that. SolarWinds Inc. is a software company that develops solutions to monitor and manage network devices, servers, storage, and applications. On December 12, 2020, they announced a sophisticated attack on their Orion platform. This affected 18,000 SolarWinds customers, including the US departments of Commerce, Defense, Energy, Homeland Security, State, and Health. This attack was one of the biggest cyber incidents the public has witnessed in years. According to Microsoft (`https://bit.ly/3q6wSec`), 44% of victims of this attack were in the IT industry and 18% were government institutions. This attack marked a milestone in cybercrimes due to the following reasons:

- Instead of attacking high-profile targets directly, the attackers chose a common "supplier" as the target.
- The attackers gained access to SolarWinds back in September 2019.

- The attackers did a dry run with the October 2019 version of the Orion platform to test their ability to include malicious code in a software build.
- The attackers injected malicious code into **SolarWinds.Orion.Core. BusinessLayer.dll** on February 20, 2020.
- SolarWinds updates with this malicious code were available to customers from March 26, 2020.
- The attackers removed malicious code from the SolarWinds environment on June 2020.
- According to a FireEye report (`https://bit.ly/3ER8Isq`), the initial dormant period of the attack could have been up to 2 weeks. This means even if your system had the malicious code, you wouldn't have noticed anything immediately.
- On a compromised system, attackers were able to initiate jobs such as transferring files/data to third-party servers, executing files, collecting information about the system including credentials, rebooting the server, and disabling system services.
- Once attackers had credentials, they moved laterally through on-prem systems to gain access to ADFS (Active Directory Federation Server).
- Once the attackers had privileges to create SAML tokens, they used them to access cloud services such as Microsoft 365.
- The SolarWinds attack was the first occasion when the Golden SAML attack method was used.

This particular attack taught us a few things:

- **The importance of the zero trust security approach –** The zero trust approach to cybersecurity is not only to prevent a breach but also to prevent lateral movement if there is a breach. We always have to assume a breach. More details about the zero trust approach will be discussed later on in this section.
- **Target on-prem to gain access to cloud resources –** In this attack, cyber criminals gained privileges to access the ADFS environment to create SAML tokens. These tokens allowed them to access cloud services without a password. Typically, businesses are more focused on protecting cloud resources, but this attack proves we need to think about the whole access life cycle.

All attacks have something in common. They are all after some sort of "access" to systems first.

It could be a username and password, certificate, or even an SAML token. Once attackers have initial access, then they start to laterally move until they have access to accounts with privileges, which can help them to do their tasks such as stealing data, causing disruptions, or conducting espionage. So it is a greater challenge for IAM to protect digital identities from these rising cybercrimes.

However, in the fight against cybercrime, organizations have to overcome some other challenges as well. According to the *COVID-19 on Enterprise IT Security Teams Report* issued by (ISC)² (`https://bit.ly/3mLiJkq`), organizations face the following challenges:

- About 20% of enterprises were forced to reduce their IT security operations budgets this year.
- 36.4% of IT security organizations froze hiring during the pandemic.
- 31.5% of IT security organizations reduced the work hours of engineers.
- 25.1% used temporary furlough methods to reduce operation costs.
- 21.7% of IT security organizations reduced the salary of engineers during the pandemic.
- 17.4% of IT security organizations reduced the number of staff with layoffs.

We already have a huge skill shortage in cybersecurity. Covid-19 has had a negative financial impact on some businesses. Because of that, businesses will have difficulties funding cybersecurity projects and developing cybersecurity skills in the coming years.

Zero trust security

With the Covid-19 pandemic, most businesses have not had the option of allowing their employees to work from home. We can't protect corporate data and identities appearing in unsecured home networks by using the same security approach we use in closed networks. This has created a huge opportunity for cyber criminals as most companies didn't have time to evaluate the risks involved in remote working and prepare themselves beforehand. Most companies are still "catching up" on cybersecurity risks related to remote working. According to an IBM report (`https://ibm.co/3wwOSjf`), remote working has increased the average cost of a data breach by $137,000. According to a survey done by Malwarebytes (`https://bit.ly/3HUQWXc`), 20% of their responders said they faced a security breach as a result of a remote worker. 44% confirmed they did not provide any cybersecurity training to employees that focused on the potential threats of working from home.

Interestingly, this study also confirmed that only 47% of employees are aware of the cybersecurity best practices when working from home.

The above stats show that the sudden shift to working from home creates risks for companies. This also confirms that the traditional parameter defense approach is not going to meet modern cybersecurity requirements. The best way to address this challenge is to take a Zero Trust security approach. The Zero Trust security model has three main principles:

- **Verify explicitly** – This means we need to verify each and every access request equally. This shouldn't change based on the network location, person, or role. In the **Nobelium** attack, we can clearly see that if there was explicit verification in place, it could have been prevented at many stages. Traditional security models are based on the "trust but verify" approach, but the zero-trust model takes a completely opposite approach, which is "never trust, always verify."
- **Least privileges access –** Almost all engineers in IT departments usually have Domain Administrator or Enterprise Administrator rights. But some of them only use it to do basic administrative tasks such as password resets. Least privilege access means users will only have privileges to do the tasks they are supposed to do. This will prevent the lateral movement of attackers and stop them from owning privileged accounts.
- **Assume breach –** Cyber criminals are also humans. We can't close all the doors. These criminals always find ways to get in. They change their tactics and methods from time to time. We need to assume a breach. The important questions are, if there is a breach, how can we recognize it? How fast can we recognize it? To do that, we need to have tools and services:
    - To collect various logs from systems
    - To analyze that data effectively
    - To do user behavior analytics
    - To detect anomalies

More information about the Nobelium attack is available in the following articles, which are published by Microsoft:

- `https://bit.ly/3wl8fvx`
- `https://bit.ly/3BJfbDv`
- `https://bit.ly/3bI09Dx`

To enforce the principles of the Zero Trust model, we need IAM solutions such as Azure Active Directory. Based on the lessons we learned from attacks such as **Nobelium**, more and more businesses will start to follow this security approach in the next few years.

Password-less authentication

Back in 2004 at the RSA Security Conference (San Francisco), Bill Gates said "There is no doubt that over time, people are going to rely less and less on passwords. People use the same password on different systems, they write them down, and they just don't meet the challenge for anything you really want to secure." Over the years, this statement has been proven over and over. Passwords are no longer secure. Passwords are breakable. The UK's National Cyber Security Center has done a study to examine the passwords leaked by data breaches. According to them, the number one password used is "123456."

So, if passwords are failing, what else can we do to improve security in the authentication process? Multi-factor authentication can add another layer of security into the authentication process. It can be SMS, a phone call, an OTP code, or a phone app notification to further confirm the authenticity of the access request. There are many different MFA products available on the market. However, MFA doesn't eliminate the requirement for passwords.

But now we have an option to replace traditional authentication with password-less authentication. This is basically to replace passwords with biometrics, PIN, certificates, and security keys.

Fast Identity Online (**FIDO**) is an open standard for password-less authentication. This allows authenticating in systems using an external security key built into a device.

Windows Hello for Business and Azure Active Directory support password-less authentication based on FIDO2 security keys.

FIDO2 is the third standard that came out of the FIDO Alliance. FIDO2 consists of a **Client to Authenticator Protocol** (**CTAP**) and the W3C standard WebAuthn. When we use FIDO2 security keys for authentication:

1. The user registers with the WebAuthn remote peer (FIDO2 server) and generates a new key pair (public and private).
2. The private key is stored in the device and is only available on the client side.
3. The public key will be registered in the web service's database.

4. After that, in the sign-in process, the system will verify the private key, which always needs to be unlocked by a user action such as a biomimetic process or a PIN.

More information about WebAuthn is available at the following links:

- `https://bit.ly/3wkLW93`
- `https://bit.ly/3bQfamF`

Over the last few years, password-less authentication has grown significantly and it will continue to do so in the coming years. According to Gartner, "By 2022, Gartner predicts that 60% of large and global enterprises, and 90% of midsize enterprises, will implement password-less methods in more than 50% of use cases—up from 5% in 2018." Azure AD now supports password-less authentication using FIDO2 keys. This can be used to authenticate into cloud resources as well as on-prem resources. I have already written some articles about the configuration of FIDO2 keys. You can access those here:

Step-by-step guide: Azure AD password-less sign-in using FIDO2 security keys: `https://bit.ly/3GTjHmG`.

Step-by-step guide: Enable Windows 10 password-less authentication with FIDO2 security keys (Azure AD + Microsoft Intune): `https://bit.ly/3wl8wyz`.

Digital ID

So far in this chapter, I have used the term "digital identity" a few times. Digital identity is a form of identification that can be used to recognize a person using digital channels. When I log in to my LinkedIn account, I use a username and password. The username and password were created when I signed up to LinkedIn. When I log in to my bank's online service portal, I use a different username and password. Both of these accounts represent my identity. Instead of using multiple digital identities, what if we can agree on one digital ID that allows you to use multiple online services such as healthcare, banking, travel, and leisure. It will reduce the complexity of proving identity. According to a study done by McKinsey Digital (`https://mck.co/3bJK14p`), one billion people in the world don't have any legal form of ID to prove their identity. Imagine the opportunities they are missing in their day-to-day lives.

It could be preventing them from accessing public services such as education and healthcare, it could be affecting their rights, and it could be affecting their loved ones. With the Covid-19 pandemic, more and more countries are in the process of adopting this unified digital identity concept. The UK government has already created a framework for digital identity (Source: `https://bit.ly/3ENOtvz`). According to the UK government, the cost of proving identity manually offline could be as high as £3.3 billion per year. The government believes "The new digital identity will not only make people's lives easier but also give a boost to the country's £149 billion digital economy by creating new opportunities for innovation, enabling smoother, cheaper, and more secure online transactions, and saving businesses time and money." (Source: `https://bit.ly/3BQImER`.) The US has recently introduced the Improving Digital Identity Act of 2020 (Source: `https://bit.ly/3BQb5cK`) to establish a government-wide approach to improving digital identity. The **Digital ID & Authentication Council of Canada** (**DIACC**) created the Pan-Canadian Trust Framework (Source: `https://bit.ly/3nYg7z3`), which defines the conformance criteria necessary for a digital identity ecosystem and explains how digital IDs will roll out across Canada.

As we can see above, countries around the globe are already working toward regularizing digital identity. On this journey, IAM also has a role to play. There will be new laws related to digital identity. There will be new rules to comply with. Organizations will have to find an efficient way to manage these new digital identities. More importantly, we need protection from identity theft. We can't do this only by using a legacy directory service. We need IAM solutions in place to manage the complete life cycle of a digital identity.

We can clearly see a challenging time ahead for IAM. We can't talk about IAM without talking about directory services. So this is why an on-prem directory service such as Windows Active Directory still has paramount value.

Hybrid Identity and Active Directory Domain Services

Active Directory Domain Services was first introduced to the world with Windows Server 2000. For more than 21 years, AD DS has helped organizations to manage digital identities.

However, modern access management requirements are complicated. Businesses are using more and more cloud services now. The majority of the workforce is still working from home and accessing sensitive corporate data via unsecured networks. Most software vendors are moving to the **Software as a Service** (**SaaS**) model. Cybercrimes are skyrocketing and identity protection is at stake. To address these requirements, we need to go beyond legacy access management. Azure Active Directory is a cloud-based, managed, **Identity as a Service** (**IDaaS**) provider that can provide world-class security, strong authentication, and seamless collaboration. Azure Active Directory can span on-prem identities to the cloud and provides a unified authentication and authorization platform to all resources, regardless of location. This is called hybrid identity.

Azure Active Directory is often referred to as a cloud version of AD DS, but this is completely wrong. It is like comparing an iPhone with a Samsung phone. Both can be used to make calls, take pictures, watch videos, and so on. Some apps are also available for both types of devices. But you can't replace one with another as each has its uniqueness. AD DS and Azure Active Directory are the same. They have their similarities as well as differences. Let's go ahead and compare both products based on different focus areas:

Focus Area	**Active Directory Domain Service**	**Azure Active Directory**
User Provision	User accounts can be created manually or use a third-party AD management and automation solution such as Adaxes to automate the user provisioning process.	We can sync user accounts from on-prem Active Directory by using Azure AD Connect. We can also create cloud-only users manually or use SaaS applications with SCIM to create users automatically.
Group Membership	Administrators have to manage group memberships manually or use PowerShell scripts or a third-party tool like Adaxes to manage memberships automatically.	Supports dynamic group membership.
Privileged Access Management	Active Directory doesn't natively support Privileged Access Management. We have to use a solution such as Microsoft Identity Manager or Adaxes to manage privileged access (sensitive group memberships, workflows).	Azure AD Privileged Identity Management (PIM) can be used to provide just-in-time workflow-based access to privileged roles.

Identity Governance	Active Directory doesn't natively support identity governance. We have to use PowerShell scripts, third-party solutions to review permissions, group memberships, and access behaviors.	Azure Active Directory Identity Governance can be used to make sure that the right people have the right access to the right resources at the right time.
Advanced Authentication	Active Directory doesn't have MFA or password-less authentication built in. We can integrate Azure MFA or another third-party MFA solution with Active Directory. We can enable password-less authentication using Windows Hello for Business (in a hybrid setup).	Azure MFA is free for Azure AD and can use to improve security with few clicks. Azure AD also supports password-less authentication based on FIDO2 standards.
Evaluate Access risks	Active Directory doesn't have the capabilities to evaluate access risks based on user location, sign-in behaviour, user account risks, and so on.	Azure AD Conditional Access can evaluate user risks based on many policy settings and allow or deny access.
SaaS Application Integration	Active Directory can integrate SaaS applications by using Active Directory Federation Service (AD FS).	Azure AD supports direct integration with SaaS applications, which support OAuth2, SAML, and WS-* authentication.
Legacy Apps	Active Directory supports app integration based on LDAP or Windows-integrated authentication.	Azure Active Directory can provide a modern authentication experience to on-prem legacy apps by using the Azure AD application proxy.
External Identities	Active Directory uses federation trusts, forest trusts, and domain trusts to collaborate with external identities. This comes with a management overhead and security risks.	Azure AD B2B simplifies integration with external identities. It doesn't require infrastructure-level changes.
Windows Device Management	Group Policy allows you to manage Windows device state at a very granular level. We can introduce standards easily to incorporate devices without additional tools or services.	Azure AD Join endpoints can manage by using Microsoft Endpoint Manager
Mobile Device Management	Active Directory doesn't natively support mobile device management. We require third-party tools to do that.	Azure AD integrated Microsoft Endpoint Manager can manage mobile devices.

As we can see in the above comparison, we can't simply replace one solution using another. But hybrid identity with Azure AD allows organizations to revamp traditional identity management and prepare themselves for the cloud era. So, the biggest question is what does the future hold for Active Directory Domain Service on this journey?

For most companies, the cloud journey starts with SaaS applications. On the majority of occasions, it is Office 365. And not only Microsoft; in general, most software vendors are transforming their services into the SaaS model. SaaS applications support different types of authentication. If an organization is looking for a single-sign-on experience, we have two options. We can set up **Active Directory Federation Service** (**ADFS**) and configure SAML-based authentication to provide SSO. However, this comes with additional costs and administrative overheads. Instead of that, we can simply sync on-prem identities to Azure Active Directory and integrate an SaaS application with Azure Active Directory for authentication. This method gives us a few advantages:

- **Fewer Changes –** We do not need to make many changes in an existing on-premises environment to enable cloud-based authentication. It only requires lightweight agents, simple firewall rules, and a reliable internet connection.
- **Advanced Authentication –** Azure Active Directory supports modern authentication standards such as OAuth2, SAML, and WS-*.
- **Advanced Identity Protection –** Azure Active Directory enriched with features and services that you can use to protect identities. Azure MFA, password-less authentication, Azure PIM, Azure Identity Governance, and Conditional Access are some of the examples of that. To start using these features and services, we do not need to make drastic changes to the existing environment. We can start by protecting identities in the cloud and then slowly extend it to on-prem as required.

As we can see, it doesn't mean we need to get rid of on-prem Active Directory to use Azure Active Directory and its features. Both can work side by side to provide a unified access experience to users. Active Directory was the top choice in industry for the last 21 years and it is the most widely used directory service. If we can move everything to the cloud, yes, it has benefits but it is not practical and not as easy as it sounds. We may have rules with which we have to comply. We may have legacy business applications that can't shift to cloud services. We may have skills and security gaps to embrace cloud technologies. Therefore, hybrid identity will not be a short-term solution for most businesses. Most businesses prefer hybrid identity instead of the cloud-only method because of the flexibility.

In the **Nobelium** attack, cyber criminals moved laterally after the initial security breach and gained control of **Active Directory Federation Services** (**ADFS**).

This allowed attackers to forge SAML tokens and get access to cloud services. Security is one of the key focus areas for public cloud services. There are various services and features available for customers to choose from, to protect identities and data in the cloud. There has been an increase in public cloud attacks recently, but the success rate is still relatively low compared to on-prem attacks. The **Nobelium** attack confirms cyber criminals are now targeting on-prem services to gain access to cloud services. Identity protection is a shared responsibility between **cloud service providers** (**CSPs**) and cloud customers. Therefore, it is the customer's responsibility to protect on-prem identities from attacks. Even if there is an attack, lateral movement needs to be prevented to protect cloud services. According to the Oracle and KPMG Cloud Threat Report 2020 (`https://bit.ly/3BUNAj6`), 92% of responders had a cloud security readiness gap. It shows we can't protect the cloud if we can't protect an on-prem environment.

In hybrid identity, Active Directory Domain Service is responsible for managing and protecting on-prem identities. There are many things we can do to protect on-prem identities from sophisticated attacks similar to **Nobelium**. We can prevent lateral movement by introducing the Active Directory tier model. We can use group policies to standardize the device and user state. We can introduce Microsoft LAPS to protect local administrator accounts. We can limit privileged accounts' appearances to **privileged access workstations** (**PAW**). If we are in a hybrid environment, we can further use cloud-based solutions such as Microsoft Defender for Identity, Microsoft Defender for Endpoint, and Azure Sentinel to identify potential security risks in the environment and address those proactively.

As we can see, in hybrid identity, we can't take our eyes off on-prem Active Directory by thinking extended identities to the cloud is going to take care of identity protection. Later in this book, we will further explore the things we can do to protect identities. Before that, let's go ahead and look into some fundamentals of Active Directory.

Benefits of using Active Directory

A few years ago, I was working on an Active Directory restructuring project for a world-famous pharmaceutical company. According to the company policies, I had to travel to their headquarters to perform the project tasks. On the day of my visit, I walked into the company's reception area. After I explained who I was and why I was there, the receptionist handed me a form to fill in. The form included questions such as name, phone number, the duration of the visit, and which department I was visiting. Once I had completed the form, I handed it over to the receptionist, and she had to make a few calls to verify whether my visit was expected, and then confirm my access to different buildings with the respective department managers.

Then, she produced a magnetic card with my details on it and handed it over to me. She instructed me how to use it and which buildings I was allowed into.

The following diagram outlines this process:

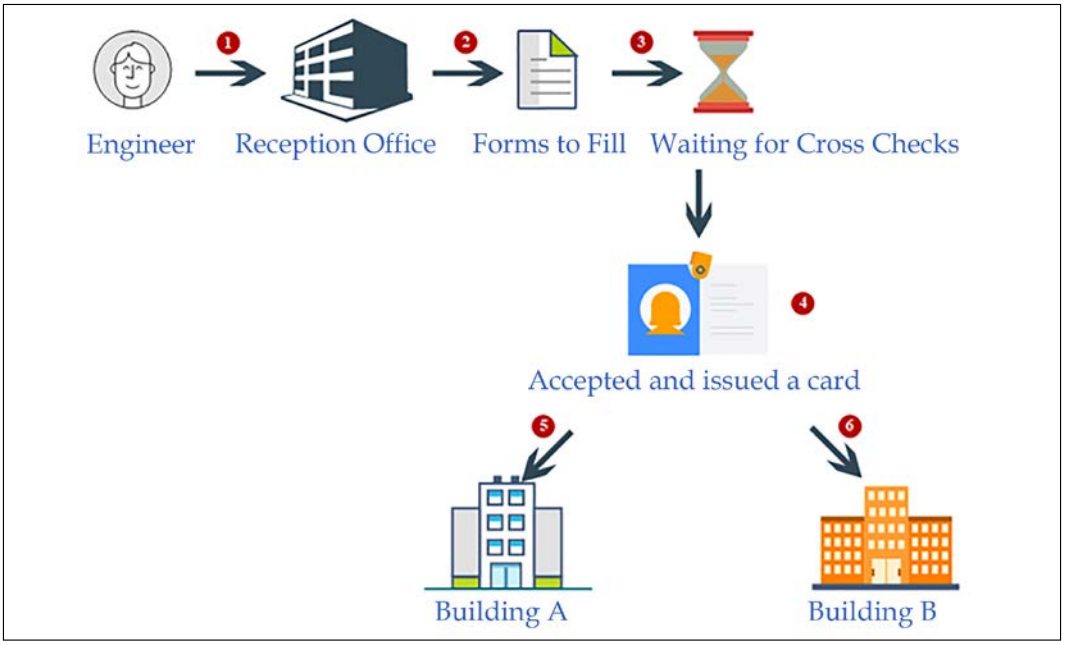


Figure 1.2: Process of entering physical headquarters

When we think about this process, we find that it contains the functions of a directory service:

- The form that the receptionist handed over to me contained certain questions to help her understand who I was. They were predefined questions, and I had to answer them in order to register my information in their system. Similar to this form, in a directory service, we have to provide values for specific attributes.
- Once I had submitted the form, she didn't hand over the magnetic card right away. She made a few calls to verify my identity, and also to confirm which buildings I would have access to. Then, my details were registered on the system, and it generated a magnetic card that had my photo and a barcode. With that, I became a part of their system, and that particular card was my unique identity within their organization. There would be no other visitor with the same barcode and identification number at the same time. Similarly, in a directory service, each identity is unique. If I needed to get access to buildings, I needed to tap the card at the entrance.

- Could I use my name or any other card to get through? No! The locking system of the building doors only recognized me if I presented the correct card. So, having a unique identity in their system was not enough; I needed to present it in the correct way to get the required access. Likewise, in an identity infrastructure, you need to validate your identity according to the method that the system has defined. It can be a username and password, a certificate, biometric information, and so on.
- I went to another building and tried to tap the card. Even when I used it correctly, the doors wouldn't open. The guard in the building asked for my card to check. Once I handed it over, he scanned it with a barcode reader and checked some information on his computer screen. Then he informed me that I was not allowed into that building, and he guided me to the correct building. This means that my information can be accessed from any building through their system in order to verify my identity and access permissions. In a similar way, in a directory, identities are saved in a central repository. This data can be accessed and verified from any system or person who has the authority.
- When I used the card in the correct buildings, it allowed me in. In the system, it first verified my identity and then checked whether I was *authorized* to work in that facility. If I was authorized, the system allowed access; if not, it rejected my request to enter.
- When I entered or left the building, I did not have to record my time. But the managers in that department knew how many hours I had worked, as my check-in and check-out times had been recorded in the system every time I tapped the card at the entrance or exit. These points collected the data, and the managers could review the information at any time. Similarly, as identities are unique in a directory, it helps to identify who has done what in a system in a given period (based on authentication and authorization data).

This system acted as an *authentication* and *authorization* system. It used different protocols and standards to manage and protect the identities that were saved in a central database. This is the primary function of a directory service.

Every organization has its own organizational structure. The most common way is to group roles, assets, and responsibilities into different departments; for example, sales, IT, production, and quality assurance. Apart from skills and knowledge, employers use company resources such as applications and hardware devices to reach their goals. In order to use these resources efficiently, it's important to have some kind of access control in place. The resources should be available for the required users at the required time. This is very easy if all the data about users, applications, and resources is recorded in a central repository that uses authentication and authorization in order to manage resources. These are the main characteristics of a directory service.

Different service providers have different directory services; for example, Novell directory services, the Oracle directory service, and the Red Hat directory service. However, the Microsoft Active Directory service is the most commonly used directory service in the industry.

In 1988, the **ITU Telecommunication Standardization Sector** (**ITU-T**) developed industry standards for directory services, called **X.500**. This was the foundation for Microsoft Active Directory services. In X.500, the **Directory Access Protocol** (**DAP**) was defined, and many alternatives were made available to enable its use with the TCP/IP networking stack. The most popular alternative was the **Lightweight Directory Access Protocol** (**LDAP**). The first version of this was released in 1993 with limited features. The University of Michigan released the first **standalone LDAP daemon** (**slapd**) server in 1995. The matured version of LDAP, LDAPv3, was released in 1997, and most vendors, including Microsoft, started developing directory services based on LDAP. Microsoft released its first Active Directory version with Windows 2000.

Centralized data repository

Active Directory stores the digital identities of users, applications, and resources in a **multi-master** database. This database is a file called `ntds.dit`, and is based on the **Joint Engine Technology** (**JET**) database engine. The data in this database can be modified using any alternative domain controller. The Active Directory database can store almost two billion objects. Users can use the digital identities that are stored in Active Directory from anywhere in the network in order to access resources. Administrators can manage the authentication and authorization of the organizational digital identities from a centralized location. Without directory services, these digital identities would be duplicated across different systems, which would add administrative overheads to managing the data.

The replication of data

There are organizations that use a single domain controller. But when it comes to complex business requirements, such as branch offices and high availability, multiple domain controllers are required (we are going to look at domain controller placement in *Chapter 11*, *Active Directory Services Part 01*). If the digital identities are managed from a centralized system, it's important that each domain controller is aware of the changes that have been made to the Active Directory database. Say a user, Jane, in the sales department, forgets her password and asks the IT department to reset it.

In 30 minutes, she's going to be working from a branch office located in a different city. The IT administrator resets her password from the headquarters' domain controller, DC01. In order to have a successful login from the branch office, this change to the directory needs to be replicated over to the domain controller in the branch office, DC05.

Microsoft Active Directory has two types of replication. If a domain controller advertises the changes made on that particular domain controller to neighboring domain controllers, it is called **outbound replication**. If a domain controller accepts the changes advertised by neighboring domain controllers, it is called **inbound replication**. The replication connections (from who and to whom) and replication schedule can be modified based on the business requirements.

High availability

High availability is important for any business-critical system in an organization. This is also applicable to domain controllers. On other systems, in order to implement high availability, we need to make software or hardware changes. With built-in fault-tolerance capabilities, Active Directory domain controllers do not need additional changes. A multi-master database and the replication of domain controllers allow users to continue with authentication and authorization from any available domain controller at any time. The Microsoft recommendation is to run at least two domain controllers to maintain the high availability of the service.

Security

Data and identity protection are very important in modern businesses. We are living in a world where identity is the new perimeter. A significant portion of this book is focused on the features of Active Directory that can secure your digital identities from emerging threats. Active Directory allows you to use different authentication types, group policies, and workflows to protect the resources in your network. Even applications benefit from these technologies and methodologies.

Auditing capabilities

Setting up advanced security policies will not be enough to protect your digital identities. Periodic audits will help you to understand new security threats. Active Directory roles come with in-built auditing capabilities. In an Active Directory setup, there can be events related to user authentication, directory service modifications, or access violation. All these events will be recorded in the event viewer under different logs.

Single sign-on (SSO)

In an organization, there can be many different applications in use. Each of these applications may have a different authentication mechanism. It will be difficult to maintain different user credentials for authentication on different applications. Most application vendors now support integration with Active Directory for authentication. This means that with Active Directory credentials, you can authenticate on different systems and applications that are used by your organization. You will not need to keep typing your credentials in order to get access. Once you authenticate on a computer, the same session will be used to authenticate other Active Directory-integrated applications.

Schema modification

Any kind of database has its own structure, called the **schema**. This is also applicable to an Active Directory database. The Active Directory schema mainly contains two types of information.

1. A definition of every object class in Active Directory
2. A definition of every attribute in an Active Directory object

Based on the AD version, the number of object classes and the number of attributes defined in the schema can be different. By knowing the schema, you can modify or extend it. This is important for the development of Active Directory-integrated applications. Microsoft publishes **Active Directory Service Interfaces** (**ADSI**) with a set of **Component Object Model** (**COM**) interfaces, and it can be used to access Active Directory service features from different network providers. Application developers can use it to develop their application to be Active Directory-integrated and publish it to the directory. Users can search for the service through Active Directory, and applications can access Active Directory objects as required.

Querying and indexing

By maintaining a central data repository, Active Directory also allows users and applications to query objects and retrieve accurate data. If I need to find the user John's account, I do not need to know which branch he is in, or to which department he belongs. With a simple Active Directory query, I will be provided with information about the user account. In a manner similar to when we add a new object to the directory, objects will publish their attributes and make them available to users and applications for queries.

These are some of the main capabilities of the Active Directory service, and these features will be explained in detail in later chapters, including how to plan, implement, and maintain them within your identity infrastructure.

Understanding Active Directory components

Active Directory components can be divided into two main categories:

- Logical components
- Physical components

When you design your Active Directory setup, you need to consider both components. The logical components of the Active Directory structure can change at any given time according to business requirements. But you won't be able to modify the physical components as easily as the logical components. The placement of these components will define the efficiency, security, reliability, and manageability of your identity infrastructure. So, it's crucial that we get it right at the beginning before we move on to advanced planning.

Logical components

Each business has its own hierarchical organization layout. It may contain multiple branch offices, multiple groups of companies, and many different departments. Each of these components in the business carries out different operations. Operations in the sales department are completely different from in the IT department. Everyone is bound to the company by following different operational guidelines and targets. When we design the Active Directory setup, we need to match it with the company hierarchical layout, in order to effectively manage resources and security. The logical components of Active Directory help you to structure the identity infrastructure by considering design, administration, extensibility, security, and scalability.

The Active Directory logical structure contains two types of objects. Objects can be either **container objects** or **leaf objects**. Container objects can be associated with other objects in the logical structure. Leaf objects are the smallest components in the logical structure. They will not have any other child objects associated with them.

In the following section, we are going to explore more about logical components in the Active Directory environment.

Forests

The Amazon is the world's largest rainforest. There are many different animal species, and more than 400 tribes live there. Each of these animal species is different. Reptiles, mammals, snakes, and fish all have different characteristics, and we can group each of them by considering their characteristics. The tribes that live in the forest also have their own languages, cultures, and boundaries. But all these animals and tribes share one forest. They use food, water, and other resources from the Amazon rainforest in order to survive. The Amazon rainforest has well-defined boundaries. Another forest 100 miles away from the Amazon is not called the Amazon rainforest. Its name and boundaries are unique.

The Active Directory forest can also be explained in a similar way. It represents a complete Active Directory instance. It is made of one or more domains and domain trees. We will explore what domains and domain trees are in the following sections. Each domain has its own characteristics, boundaries, and resources. But, at the same time, it shares a common logical structure, schema, and directory configuration within the forest. Similarly, tribes have a relationship with the forest and other tribes, and domains in the Active Directory forest will have a two-way trust relationship. Different tribes in the Amazon forest aren't named after the *Amazon*; each tribe has its own name. Similarly, domains in a forest can contain any domain name:

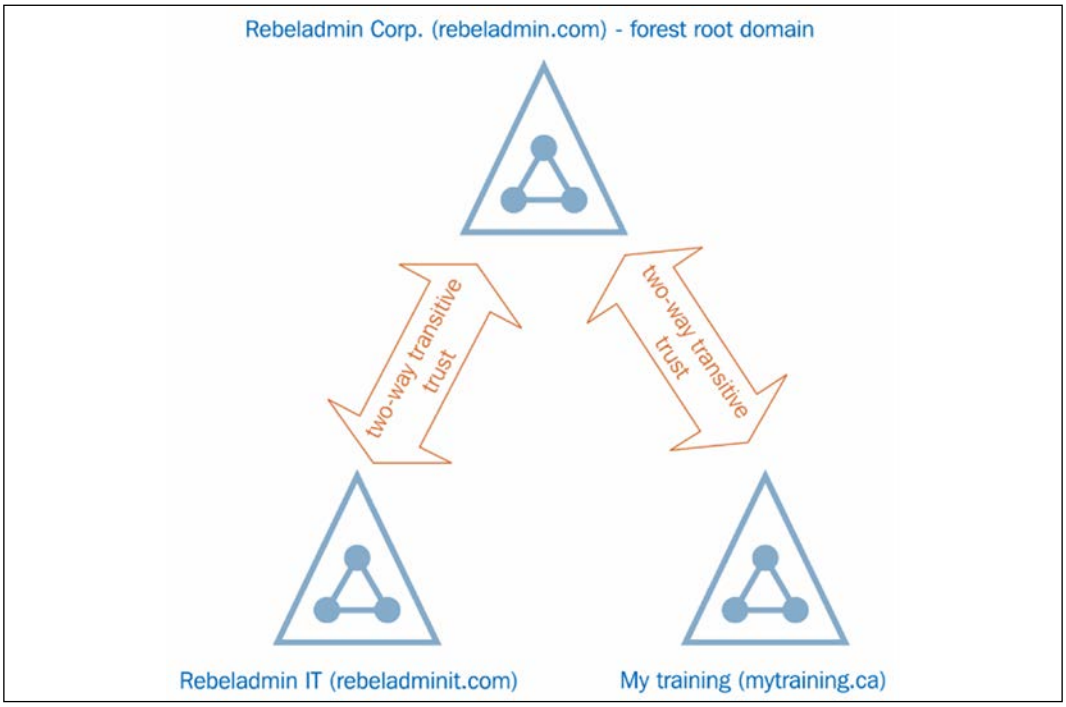


Figure 1.3: Domains in the rebeladmin.com forest

The first domain controller in the Active Directory service deployment is important. When you create the first domain, it will also create the forest. Then, the first domain will become the forest root domain. A domain tree contains its own root domain, but forests can contain multiple root domains.

In the previous diagram, **Rebeladmin Corp.** is an IT solution provider. `rebeladmin.com` is the forest root domain. It does have another two companies: one is **Rebeladmin IT** with the `rebeladminit.com` domain name, and it provides managed IT services. The other company is **My training**, with the `mytraining.ca` domain name, and it provides IT training to professionals. `rebeladminit.com` and `mytraining.ca` are both root domains in their own domain trees. Both domains in the forest will trust each other with **two-way transitive trust**.

Two-way transitive trust is a logical link between domains, where the trusting domain honors the logon authentication of the trusted domain. When considering the previous example, users in `rebeladminit.com` can authenticate into `mytraining.ca`, and vice versa. Any object located in a particular domain inherently trusts other objects in other domains in the same forest. This is not the same as when considering authentication between forests. For that, it may (depending on the trust method) require additional login credentials. An organization can have a single forest or multiple forests based on the company's business requirements.

When Microsoft releases a new Active Directory service version, new features are bound to the forest and domain functional levels. If you want to use Active Directory Domain Services 2016 forest-level features, your directory's Active Directory forest should use the Windows Server 2016 forest functional level. Before Windows Server 2012 R2, forest functional-level upgrades were one-way. Now, it is possible to roll back to the lower forest functional level as long as you are not using the features specific to the current functional level. The forest functional level is dependent on the oldest domain controller version in the network.

For example, if the forest functional level is Windows Server 2008, it is allowed to install the domain controller inside the forest with the operating system, Windows Server 2022. But this doesn't mean it can use the features provided by Windows Directory Services 2022 until it upgrades its domain and forest functional levels. If you upgrade the forest functional level to Windows Server 2016, you can only have domain controllers running a minimum of Windows Server 2022.

The following table explains the supported Domain Controller operating systems for each functional level.

Functional Level	**Domain Controller Operating System**
Windows Server 2016	Windows Server 2022 Windows Server 2019 Windows Server 2016
Windows Server 2012R2	Windows Server 2022 Windows Server 2019 Windows Server 2016 Windows Server 2012 R2
Windows Server 2012	Windows Server 2022 Windows Server 2019 Windows Server 2016 Windows Server 2012 R2 Windows Server 2012
Windows Server 2008R2	Windows Server 2022 Windows Server 2019 Windows Server 2016 Windows Server 2012 R2 Windows Server 2012 Windows Server 2008 R2
Windows Server 2008	Windows Server 2022 Windows Server 2019 Windows Server 2016 Windows Server 2012 R2 Windows Server 2012 Windows Server 2008 R2 Windows Server 2008

Domains

Referring back to my example about the Amazon rainforest, we can say that there are more than 400 tribes living there. Each of these tribes is unique in certain ways. Each tribe has a different language and culture. Each tribe has its own territory for hunting, farming, and fishing. Each tribe knows its boundaries and does not cross others' boundaries as that can lead to war between tribes. Each tribe has its own tools and methods for hunting and farming. Also, each tribe has different groups assigned to different tasks. Some are good at hunting, some are good at farming, and some are good at cooking. All their contributions help them to survive and grow as a tribe.

The Active Directory domain can also be explained in a similar way. The domain contains the logical components to achieve the administrative goals of the organization. By default, the domain becomes the security boundary for the objects inside it. Each object has its own administrative goals. Individuals in tribes have different identities and responsibilities, but all of them are part of the tribe and the forest. In the same way, all the objects in the domain are part of a common database. Also, everyone in the tribe still needs to follow some of the common rules. Objects in the domain are also controlled by the defined security rules. These security rules are only applicable within that particular domain and are not valid for any object outside the domain boundaries. A domain also allows you to set smaller administrative boundaries within the organization. In the previous section, I explained that a forest can contain multiple domains.

Managing a forest is difficult, as its administrative boundary is large, but the domain allows you to set smaller administrative targets. Active Directory is divided into multiple partitions in order to improve its efficiency. The domain is also a partition of Active Directory. When I described the Active Directory forest, I mentioned that every domain inside the forest shared the same schema. Each of the domain controllers also has a copy of the domain partition, which is shared only by the domain controllers within the same domain tree. All the information about objects in that particular domain is saved in that domain partition. This ensures that only the required data is replicated across the domain trees and forests.

The Active Directory domain's functional levels define the Active Directory capabilities. With every new version of the directory services, new features are added to the domain's functional level. In order to use the features within the domain, the domain functional level needs to be upgraded. The version of the domain's functional level that you can run on the domain depends on the forest's functional level. You cannot have a domain's functional level that is higher than the forest's functional level.

Domain trees

A **domain tree** is a collection of domains that reflects the organization's structure. My parents and I are bound by a parent-child relationship. It is obviously different from other kinds of relationships. Similarly, domains inside the domain tree have a parent-child relationship. The first domain in the domain tree is called the **parent** domain. This is also the root domain. All other domains in the domain tree are called the **child** domains. There will be only one parent domain in a domain tree.

In some documentation, child domains are also called **subdomains**. When dealing with internet domains, the creation of an additional placeholder, a sub-URL, is sometimes required. For example, `rebeladmin.com` is the domain name that is used for the website and organization needed to host another website, in order to maintain support requests. But it needs to use the same contiguous namespace. To do that, we can create another folder in the domain root, and create a **Domain Name System** (**DNS**) record for the `support.rebeladmin.com` subdomain:

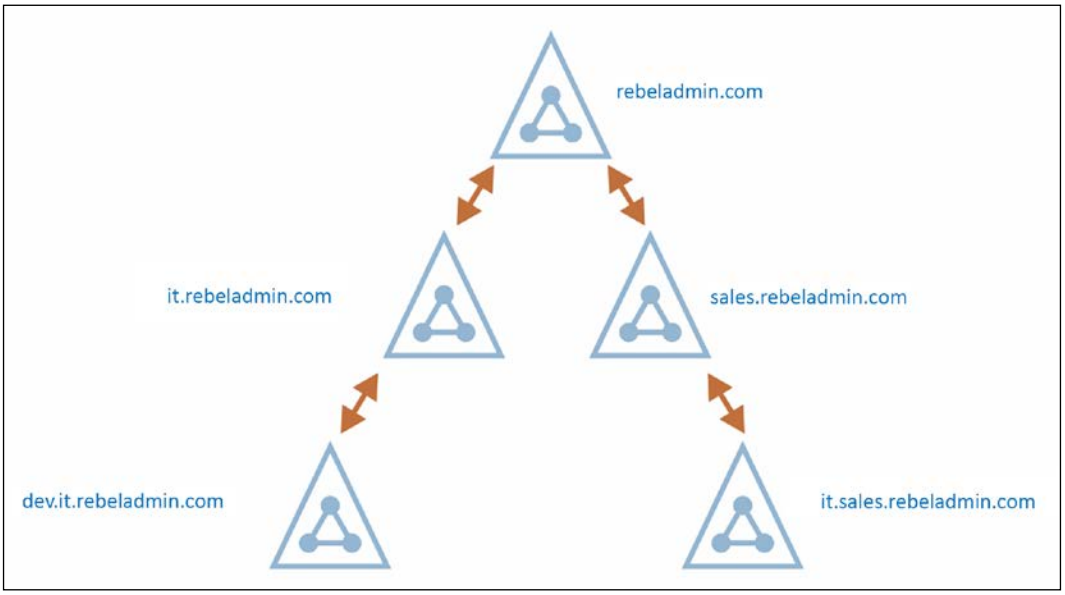


Figure 1.4: Subdomains in a contiguous namespace

An Active Directory forest can contain non-contiguous domain names. But within the domain tree, it will share the same contiguous namespace. In the previous example, `rebeladmin.com` is the parent domain for the domain tree. It has two child domains, `it.rebeladmin.com` and `sales.rebeladmin.com`. As you can see, it shares the same `rebeladmin.com` namespace. Similarly, when it goes down to the next level in the domain tree, it shares the namespace from the preceding level. Each child domain maintains its own domain partition.

This configuration data will be replicated only to the domain controllers of the same child domain. When the child domain is introduced to the domain tree, it will automatically create a trust relationship with the parent domain. If two child domains on different domain trees want to authenticate, authenticated traffic must pass through the forest root domains.

All domain trusts within the Active Directory forest are two-way transitive trusts. Two-way trust means that the authentication requests can be processed between two domains in both directions. Transitive means it goes beyond the initial two-way trust between domains, and trusts its child domains too, even though there is no direct connection.

Organizational units

In the preceding section, I explained how we can group objects using domains and forests. But within the organization, objects can be categorized into different groups according to operations, organizational structure, geographical locations, or roles and responsibilities. As an example, organizations have multiple departments. We can convert each of these departments into child domains and group each of the department objects. But the child domain needs a separate domain controller, as it will have a separate domain partition.

Isn't there a better way to group these objects within the domain? That's where organizational units come in. Organizational units help group objects on a smaller scale within the domain. The most common way is to group objects that have similar security and administrative requirements together. For example, there are more than 50 users in the sales department. The sales department uses common shared folders and printers. Their security requirements for data and networks are similar. Therefore, we can create an **organizational unit** (**OU**) called *sales* and group all the sales department users into it. We can now apply security policies at the OU level, instead of the user level.

When deploying a domain controller, it creates a default OU structure to segment the most common object types, such as users, computers, and domain controllers. The administrator can add, remove, and delete an OU as required. An OU also helps to apply group policies to a selected group of objects. As an example, if an organization has an OU for each department, we can apply different group policies for each department by simply targeting the OU.

Sometimes, I have seen engineers removing/modifying the default OU structure. All these default OUs have different security policies attached. If it really needs to be changed, it is important to compare the security policies that are applied and reattached to the new OU if required. I highly recommend that you do not modify/remove domain controllers' default OU at least. That said, you are still allowed to add or change security policies applied to default OUs.

Once an object is assigned to an OU, it inherits the security settings and permissions that are applied to the OU level. If the same object is moved to a different OU, then it will apply the settings from the new OU, and discard the settings that were applied from the previous OU. OUs also help to delegate administrative control to individuals for specific tasks. Domain administrators have privileges that allow them to manage any object within the domain. But it's possible to create administrators and assign them to manage objects and resources at an OU level. For these administrators, the OU will be the security boundary. They will not be able to modify any other object outside that particular OU. I will be explaining delegated administration later in this book. OUs are container objects. They can be associated with similar or other objects. Similar to parent-child domains, OUs can also contain child OUs. These are also **nested organization units**.

OUs can also contain object types, such as users, groups, contacts, computers, organizational units, and printers:

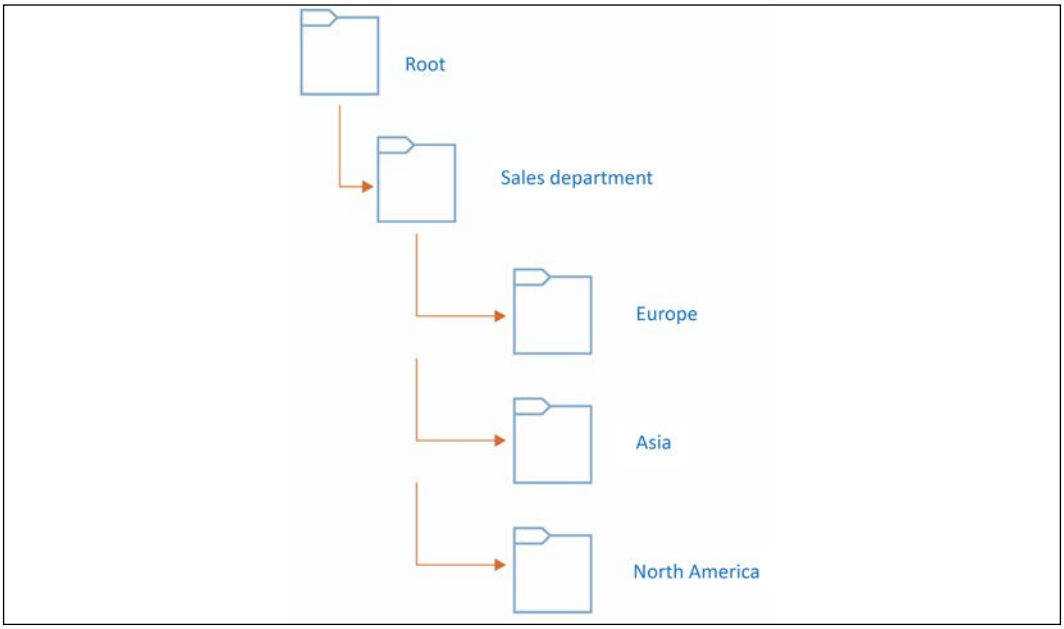


Figure 1.5: Organizational unit hierarchy

In the previous example, Rebeladmin Corp. has a **Sales department**. In the OU hierarchy, the first thing you need to do is create an OU called **Sales department**. All the regional offices have their own sales department. Most of the security and administrative requirements for objects in the sales department are the same. But creating OUs based on geographical areas will allow domain administrators to delegate control over those objects to individuals or groups in the regional offices. Also, if a specific security policy needs to be applied to a regional office sales department, it can be applied on a relevant OU level, rather than applying it to the entire **Sales department** across the branch offices. All the child OUs inherit the permissions that are applied to its parent OU by default.

In the previous example, individuals or groups who have permission to control **Sales department** objects have control over the objects in the **Europe**, **Asia**, and **North America** OUs by default. The OU hierarchy is independent. It is not going to affect any other domain's OU hierarchy. The OU can also contain objects only from the same domain.

Physical components

In the previous section, I explained the logical components of Active Directory. Now, it's time to look into the physical components. Even though the logical and physical components are equally important in Active Directory Domain Services' design, they are independent. Replication is the core feature of Active Directory Domain Services. If a system has multiple domain controllers, changes made in one domain controller should be replicated to others. Physical component placement can affect Active Directory replications in certain ways. Logical components can easily be rearranged compared to physical components.

Domain controllers

The domain controller is a computer that runs a Windows Server operating system, and holds the Active Directory Domain Services role. It can be either a physical server or a virtual server.

The domain controller holds the directory partition that will be replicated to the other domain controllers in the same domain. The domain can have any number of domain controllers. The number of domain controllers is dependent on the enterprise's size, geographical placement, and network segmentation. In Windows NT, it uses multiple domain controllers, but it maintains a single-master schema. This means that directory changes can only be made from a specific domain controller. Since Windows 2000, there has been support for the multi-master mode. Any object-level changes made in one domain controller will be replicated to all other domain controllers (directory service-related).

That said, some of the Active Directory-related operational role changes can only be modified by the designated operation master role owner (FSMO roles).

Before Windows 2000 Domain Services, one of the domain controllers acted as the **primary domain controller** (**PDC**), and all other additional domain controllers were called **backup domain controllers** (**BDCs**). Some people still use this terminology to describe the operations of the domain controllers in the infrastructure. But after Windows Server 2000, the only difference between domain controllers was either their **flexible single master operation** (**FSMO**) role holder or the global catalog server.

The global catalog server

The global catalog server holds the full writable copy of objects in its host domain, and the partial copy of the objects in other domains in the same forest. The partial replica contains a copy of every object in the forest and the most commonly used attributes in queries. Applications and users in one domain can query for the objects in another domain (in the same forest) via the global catalog server. All domain controllers in the domain will not be global catalog servers by default. When installing the first domain controller, it will become the global catalog server, and other domain controllers can be promoted as global catalog servers according to the business requirements. Not every domain controller in the domain needs to be a global catalog server.

More details about global catalog servers and global catalog server placement can be found in *Chapter 3*, *Designing an Active Directory Infrastructure*.

Active Directory sites

The Active Directory site defines a physical topology of the network. Sites can be separate buildings in a campus network, with the branch office in a separate city or even in a separate country. For example, the head office of Rebeladmin Corp. is located in London, UK. It runs a few domain controllers (**DC01** and **DC02**) within its physical network. It uses IP address allocation for the network with the subnets of `192.168.148.0/24`, `10.10.10.0/24` and `172.25.16.0/24`. Due to business requirements, the company opened a branch office in Toronto, Canada. It got its own domain controllers (**DC03** and **DC04**) running, but logically, it is in the same Active Directory forest and domain. Both networks are interconnected with a leased line.

The Canada network uses the IP subnets of `10.11.11.0/24` and `172.0.2.0/24`:

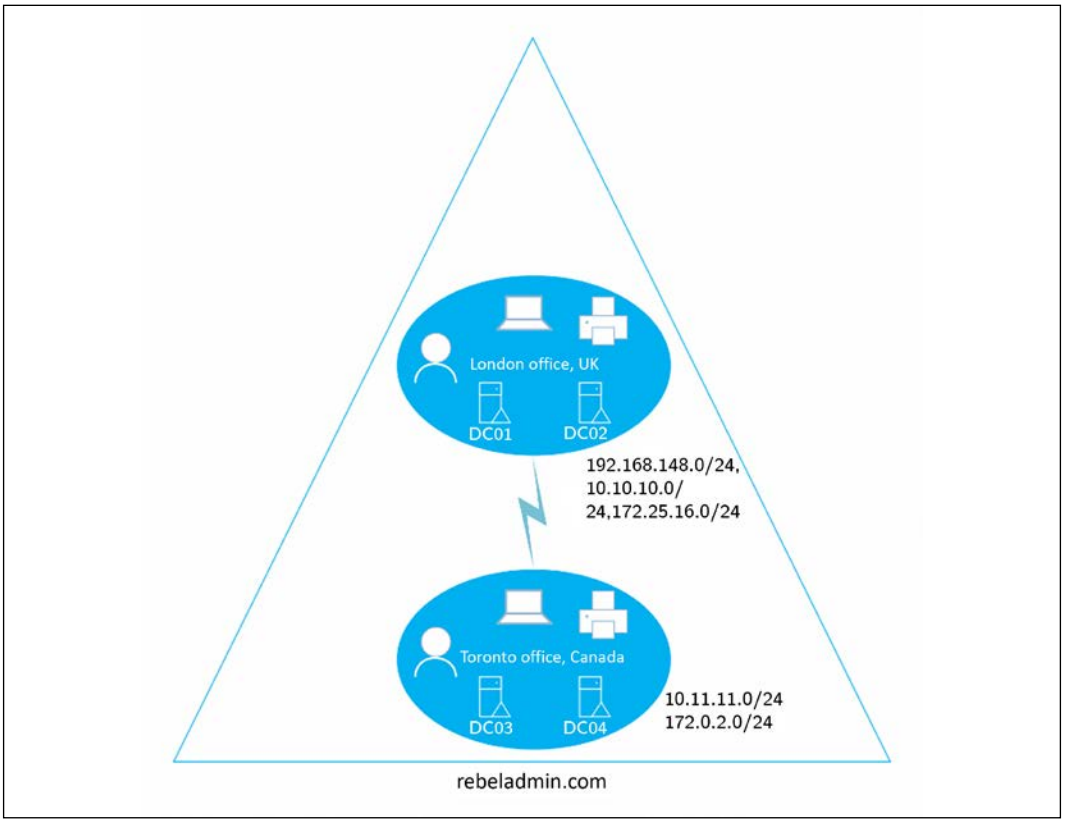


Figure 1.6: Active Directory connection

In the preceding diagram, the two offices can be identified as two sites. This is because there are clearly two network segments. Active Directory's logical design does not really consider physical network segmentation. Since they are in the same domain and forest, **DC01** to **DC04** should replicate changes to each other in order to maintain a healthy identity infrastructure.

Mainly, there are three benefits that we can identify:

- **Replication**: In a typical AD DS setup, all domain controllers are set to replicate changes between each other, assuming all are connected via fast network links. But in the real world, they're not. Sometimes, connections between two sites are 256 kbps or 512 kbps. The same links will also be used for other enterprise operations. Using AD DS sites, it's possible to perform bandwidth optimization and replication schedules for reliable replication across domain controllers.

- **Service location**: In an infrastructure, there can be Active Directory-integrated applications/services; for example, Active Directory certificate services and exchange services. Using sites and the subnet setup, we can point users to the nearest server for the services. So, users on the Toronto site are served by the Microsoft Exchange Server (mail server) on the Toronto site when they try to access an email, instead of passing the request to the London site.
- **Authentication**: When a user logs in to the domain, they need to communicate with the domain controller to gain authentication. In the preceding example, a user on the Toronto site does not need to connect to a domain controller on the London site for authentication. AD DS sites will allow you to ensure that users on the Toronto site will use the nearest domain controller for authentication. This will reduce latency and bandwidth through the site links.

More information about Active Directory sites is available in *Chapter 11*, *Active Directory Services – Part 01*.

Since AD DS sites represent a physical network topology, when changes are made to the physical topology, they also need to be updated on the AD DS site configuration. For example, if a new subnet is added, this information needs to be updated in the AD DS site subnet section, too. Sometimes, engineers forget to do this, which prevents infrastructures from having the full benefits of AD DS sites.

Understanding Active Directory objects

If we need to describe a person or thing, we use different adjectives. This can include personality, ethnic background, physical appearance, or other characteristics. Most of these are not unique. For example, when you talk about a 6-foot-tall boy, there could be lots of 6-foot-tall boys in the city. But it still explains that the *person* that we're trying to describe is definitely not a girl. If we need to uniquely identify a person or thing, we need to identify some unique attributes associated with them. If it's a person, then their passport number, telephone number, or social security number will make it easier to uniquely identify them from others. If it's an object, the unique identifier could be the serial number or barcode.

Within an organization, there are many physical entities. These can be either employees or resources. In order to manage these using Active Directory Domain Services, each of these physical entities needs to be presented to Active Directory. Active Directory will understand these entities as objects.

In Active Directory, there are two types of objects. *Container objects* can store other objects in Active Directory. The domain itself is an example of a container object. The organizational unit is also a container object. *Leaf objects* cannot store other objects in Active Directory. A service account is an example of a leaf object.

In the same way that we use adjectives to describe a person or a thing, Active Directory objects use attributes to describe their nature. For example, the following screenshot shows the wizard you will get when you create a new user account by using **Active Directory Users and Computers** (**ADUC**). In the wizard, in the following screenshot (on the left-hand side), **First name**, **Last name**, **Full name**, and **User logon name** are attributes. In the same way, when you create a computer account, it needs a **Computer name** attribute to describe it (on the right-hand side):

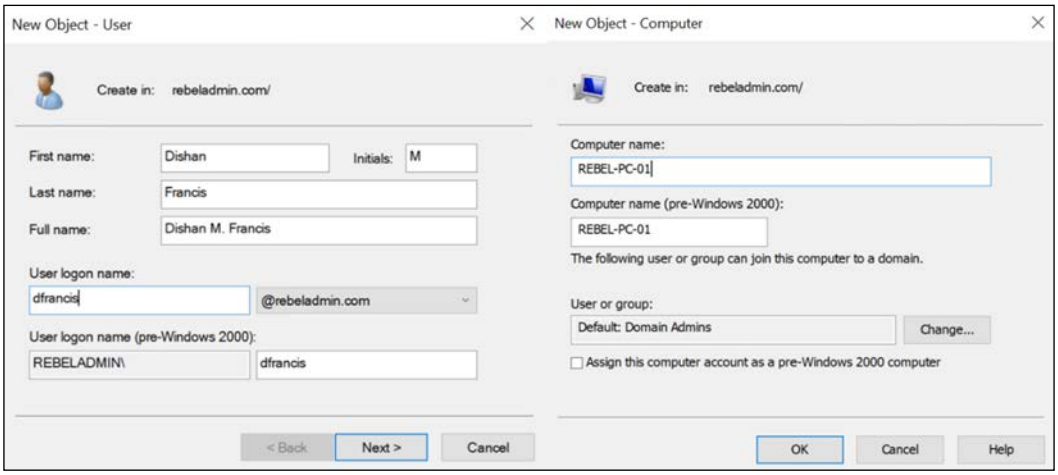


Figure 1.7: Creating new objects

According to the preceding screenshot, depending on the object type, the associated attributes are also changed. Also, it doesn't matter if you create one user object or hundreds of user objects in Active Directory. You still need to use the exact same attributes to describe the object you are creating. This is because each of the objects is attached to an *object class*. Within the Active Directory schema, the attributes that are attached to each object class are defined.

Let's look at a simple scenario to understand this further. When you sign up for an online service for the first time, it will provide you with an online form to complete. At the backend, it is attached to a database. The information you provide will be recorded in the database for future use. If you need to sign up for the service, you must provide the answers to the questions that are asked. You cannot change the questions that you need to answer because the database will not be able to understand it. The database contains a table designed with columns, rows, and data types to store the data that will be captured from the form.

Similarly, object class attributes are defined by a schema. Active Directory does have different types of object classes. Users, groups, computers, printers, and domain controllers are examples of object classes.

Some of these attributes are mandatory for object classes. For example, in user account creation, **User logon name** must be provided in order to continue. But if we do not provide the **last name**, we can still proceed with the user account creation. Attribute values also need to be provided with an acceptable data format that is defined by the schema. Sometimes, due to operational requirements, organizations may require custom attributes. By modifying the Active Directory schema, it is possible to add additional attributes to the object classes. This will be demonstrated further in *Chapter 7*, *Managing Active Directory Objects*.

Globally unique identifiers and security identifiers

In a city or organization, there can be multiple people with the same name. But their passport number or social security number will be unique to them. So, in order to identify a person or thing accurately from a group of similar things, we need to consider the associated unique value.

In an Active Directory database, nearly two billion objects can be stored. How will it uniquely identify each and every object? Every time we create an object in Active Directory, it will be assigned with one or two unique values. If it is a user or group object, it will receive a **globally unique identifier** (**GUID**) and a **security identifier** (**SID**). The GUID value will be saved in the `objectGUID` attribute in each object and the SID value will be saved in the `objectSid` attribute in each object.

In order to view the GUID and SID values for the user account, the following PowerShell command can be run from the domain controller:

```
Get-ADUser username
```

`username` can be replaced by the actual username of the user.

In the following screenshot, `ObjectGUID` lists the GUID value and `SID` lists the SID value associated with the user account:

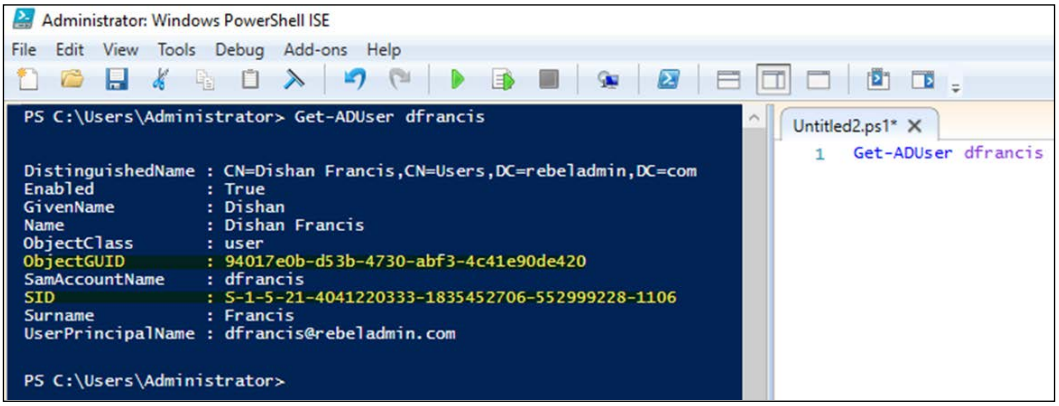


Figure 1.8: ObjectGUID and SID as they appear in PowerShell

`ObjectGUID` is a 128-bit value and is applied to each and every object in Active Directory. This value is not just for the particular Active Directory domain. It is valid globally as well. Once a GUID is assigned to an object, it will be there until the object is deleted from the directory. Modifying or moving objects will not change the value of the GUID. The `ObjectGUID` attribute value will be published to the global catalog servers. If an application in a domain needs to search for a user object, the best method will be to query using `ObjectGUID`, as it will give an accurate result.

There is a misunderstanding that the GUID value is a *unique value*. None of the documentation says that this value is unique. They only say it is quite unlikely to have a duplicated GUID as the method used to generate it is complex.

The `SID` value for an object is unique within its domain. The `SID` values associated with the user will be changed if the user object is migrated to another domain. An `SID` value assigned by one domain will not be accepted by another domain. As soon as a user object is migrated to another domain, a new `SID` value will be generated. Then, the old `SID` value will be saved in the `sIDHistory` attribute.

This attribute can contain multiple values. When the system creates a Kerberos ticket for user authentication, it will consider a new `SID` value and all other `SID` values listed in the `sIDHistory` attribute. `sIDHistory` is important, especially in Active Directory restructuring. The resources in the domain decide whether to access or deny permission to a user account based on the **access control list** (**ACL**). This ACL uses the `SID` values. So, if an object moves to a different domain without `sIDHistory`, it will lose its access to resources until the ACL is modified. But if the system considers `sIDHistory` when granting an access token, and if the old `SID` value is moved over to the new domain, the user is still allowed to access the resources they were assigned.

Distinguished names

Distinguished names in Active Directory can also be used to uniquely identify an object. This is very similar to the way your postal address works. A postal address uses a hierarchical path to uniquely identify you. Starting from the country, it goes to the province, then to the city, the street, and the house number. In the same way, using the full path to the object within the directory will help you uniquely identify an object.

There are three types of Active Directory naming attributes that have been used to generate distinguishing names:

- `organizationName` (**O**) or `organizationalUnitName` (**OU**): Organization represents the root-level domain. The OU is where the object is located.
- `domainComponent` (**DC**): This is the naming attribute for the domain and the DNS. If the DNS name for the domain is `rebeladmin.com`, the domain components for it will be `DC=rebeladmin,DC=com`.
- `commonName` (**CN**): This refers to the objects and containers within the directory.

In the previous screenshot, when the query for the domain user is returned, the distinguishing name for the user is as follows:

```
CN=Dishan Francis,CN=Users,DC=rebeladmin,DC=com
```

Here, `DC=rebeladmin,DC=com` represents the domain name, `CN=Users` represents the user container, and at the end, `CN=Dishan Francis` represents the actual object name.

The **relative distinguished name** (**RDN**) is a unique value within its parent container. For the preceding example, the RDN for the object is `CN=Dishan Francis`. Active Directory allows you to have the same RDN for multiple objects within the directory, but all of them need to be in separate containers. It is not permitted to have the same RDN for the object within the same container.

In the previous section, you learned that the `SID` value for the object will not be changed unless it's migrated to a different domain controller. Changing values in the object will not modify the `SID` value. But if the hierarchical path was changed for an object, the **Distinguished Name** (**DN**) would be changed. For example, if you move a user object from one OU to another, the DN value for the user object will be changed.

Active Directory server roles

There are five main Active Directory server roles. These roles are grouped together in the required Active Directory environment in order to set up and configure Active Directory server roles:

- **Active Directory Domain Services** (**AD DS**)
- **Active Directory Federation Services** (**AD FS**)
- **Active Directory Lightweight Directory Services** (**AD LDS**)
- **Active Directory Rights Management Services** (**AD RMS**)
- **Active Directory Certificate Services** (**AD CS**)

Since Windows Server 2008, these roles can be installed and configured using Windows Server Manager. It is the same in Windows Server 2022.

Each of these server roles can also be installed and configured using PowerShell. The following PowerShell cmdlets can be used to install Active Directory server roles:

PowerShell cmdlets	Description
`Install-WindowsFeature AD-Domain-Services`	This cmdlet will install the AD DS role. Please note this will only install the AD DS role on the server. This is further explained in *Chapter 6, Migrating to Active Directory 2022*
`Install-WindowsFeature AD FS-Federation`	This cmdlet will install the AD FS role.
`Install-WindowsFeature ADLDS`	This cmdlet will install AD LDS.

`Install-WindowsFeature ADRMS`	This cmdlet will install AD RMS. This role has two subfeatures, which are AD Rights Management Server and Identity Federation Support. If required, these individual roles can be installed using `Install-WindowsFeature ADRMS`, `ADRMS-Server`, `ADRMS-Identity`, or `Install-WindowsFeature ADRMS -IncludeAllSubFeature`. It will install all the subfeatures.
`Install-WindowsFeature AD-Certificate`	This cmdlet will install AD CS. This role has six subroles, which are certification authority (`ADCS-Cert-Authority`), Certificate Enrollment Policy Web Service (`ADCS-Enroll-Web-Pol`), Certificate Enrollment Web Service (`ADCS-Enroll-Web-Svc`), Certification Authority Web Enrollment (`ADCS-Web-Enrollment`), Network Device Enrollment Service (`ADCS-Device-Enrollment`), and Online Responder (`ADCS-Online-Cert`). These subfeatures can be added individually or together.

The `Get-WindowsFeature` command will list all the roles and subfeatures that are available, along with the names that can be used with PowerShell to install the roles. When you install the roles, it is important to add `-IncludeManagementTools` as management tools, as the role will not be installed by default.

You can learn more about these roles from the following chapters:

- *Chapter 11*, *Active Directory Domain Services (AD DS) – Part 01*
- *Chapter 12*, *Active Directory Domain Services (AD DS) – Part 02*
- *Chapter 13*, *Active Directory Certificate Services (AD CS)*
- *Chapter 14*, *Active Directory Federation Services (AD FS)*
- *Chapter 15*, *Active Directory Rights Management Services (AD RMS)*

Summary

This is the end of the introductory chapter on Active Directory fundamentals. I am sure most of you are already aware of most of the functions of AD DS, but refreshing your knowledge about Active Directory components and their operations before we dive deep into the advanced topics wasn't in vain. In this chapter, we looked into the future of identity management, and then we covered Active Directory's hybrid identity role, Active Directory objects, GUID and SID values, and DNs.

In *Chapter 2*, *Active Directory Domain Services 2022*, you will learn about new features and enhancements to AD DS 2022, specifically about the approach to protect digital identities from modern security threats.

2

Active Directory Domain Services 2022

Microsoft **Active Directory Domain Services** (**AD DS**) has been in the industry for over 21 years now. The first Microsoft Active Directory version was released on February 17, 2000, along with Windows Server 2000. The first production Active Directory domain was `redmond.corp.microsoft.com` and it was upgraded from the Windows NT4 domain to the pre-release version of Active Directory 2000 on April 9, 1999. After Windows Server 2000, with each and every Microsoft Server release, a new AD DS version has been released as well.

Each and every time Microsoft releases a new version of their software, as IT engineers, we talk about it, we learn about it, and then we go ahead and try it. It's good practice to be on top of industry updates. However, simply migrating to the latest version of AD DS is not going to fix or improve anything related to corporate digital identities. Before we upgrade to a newer version, we need to identify any existing problems in our current Active Directory setup and make plans to fix them. Then we need to evaluate our requirements to understand what we need and how we can get there. As an example, if our company is moving to the cloud, we need to think about how we can extend on-prem identity management to the cloud and which integration method to use (password hash sync, federation, or pass-through). If the company is embracing a zero-trust security approach, which features of Active Directory can help us achieve that? And what do we need to change in our Active Directory setup? After requirements analysis, we can upgrade and introduce new features wisely.

We are at a very interesting point in the technological timeline. As I stated in the previous chapter, today's digital identity requirements are complicated. The majority of businesses today use at least one cloud service. With the Covid-19 pandemic, a lot of businesses have accelerated their digital transformation journey. Working from home has now become the new normal. Even during this challenging time, cybercrime occurrences have increased and have become more sophisticated. This has left us with some questions:

- What can Active Directory do to help us on our cloud journeys?
- How can we improve the security of digital identities when they appear on unsecured networks?
- How can we protect our Active Directory setup from security breaches?
- If there is a breach, how can we stop lateral movement and prevent attackers from gaining control of privileged accounts?

In this chapter, we are going to look for answers to the above questions and see what AD DS 2022 can provide.

We will be covering the following topics in detail:

- The features of AD DS 2022
- **Privileged Access Management** (**PAM**)
- Time-based group memberships
- Windows Hello for Business
- Time sync improvements

We'll start with a look at AD DS 2022's most important features.

The features of AD DS 2022

AD DS's improvements are found in its forest and domain functional levels. Upgrading the operating system or adding domain controllers that run Windows Server 2022 to an existing AD infrastructure isn't going to upgrade the forest and domain functional levels automatically. We need to upgrade them manually once all older domain controllers are decommissioned. When it comes to forest and domain functional levels, there is a big difference in Windows Server 2022. Up to the Windows Server 2016 release, there have been new forest and domain functional levels. But starting from Windows Server 2019 there are **NO** new forest or domain functional levels.

The most recent forest and domain functional levels we can choose are still from Windows Server 2016. But what does this mean? If the improvements are bound to forest and domain functional levels, does this mean there are no new features on AD DS 2022? Yes, that's correct: there are no new AD DS features on Windows Server 2022. With Windows Server 2019 there was a very minor change to the schema. There was a new attribute called **msDS-preferredDataLocation**, and this can be used to specify a Microsoft 365 user's geolocation. Other than that, there were no changes with AD 2019 either.

1. When you are upgrading Active Directory from Windows Server 2016 or Windows Server 2019 to Windows Server 2022, there is no need for domain or forest functional change
2. The Active Directory schema version stays at 88 and it is the same as the Active Directory 2019 schema version

AD is the most widely used directory service on the market. It does what it is supposed to do. However, identity management requirements have changed over the years. Let's take the zero-trust security approach as an example. In the first chapter, I explained the importance of this model and how we can use it to protect identities. Micro-segmentation is a practice that's used in the zero-trust security model to create small security zones within a large network. To address access management requirements in these small security zones, we can introduce AD read-only domain controllers and maintain security standards. But micro-segmentation should be done at the network level. Also, to prevent the lateral movement of attackers within an AD setup, we can use the AD tier model. This is not an AD feature; it is more of a way to manage privileges in an AD setup. This can be done using any version of AD. Least-privilege access is also a key requirement of the zero-trust model. We can use AD time-based group memberships, account delegation, **Just-in-Time** (**JIT**), and **Just-Enough Administration** (**JEA**) to enforce least-privilege access. Except for time-based group memberships (an AD DS 2016 feature), all of the above tasks can be done using an older version of AD. But we still need tools/services such as Microsoft Identity Manager (for PAM), Microsoft Defender for Identity (for monitoring), and Microsoft Defender for Endpoint (for end-user device threat management) to manage the complete identity lifecycle and have a complete solution. As we can see, AD is already mature enough to address some of the modern identity and security requirements but not all of them. We need to understand that just having a feature-rich product is not going to solve our problems. Rather, it is the way we configure it and the way we use it that will make the difference. This is what we are going to focus on in this book. Even though we do not have any new features to explore in AD DS 2022, we are going to look into the current capabilities of AD and see how it accommodates today's access management requirements. Let's go ahead and look into some of the features that came with AD DS 2016 that are also present in AD DS 2022.

The deprecation of Windows Server 2003's forest and domain functional levels

Windows Server 2003 is no longer supported by Microsoft.

Therefore, Windows Server 2003's forest and domain functional levels have been deprecated in AD DS 2022. This was the case even with Windows Server 2012 R2. But we were able to add domain controllers on the Windows Server 2016, Windows Server 2012 R2, Windows Server 2012, Windows Server 2008 R2, and Windows Server 2008 operating systems to an AD DS 2003 domain. But Windows Server 2022 domain controllers require a minimum of the Windows Server 2008 functional level. If you are looking to migrate from AD DS 2003, first you need to upgrade to at least the Windows Server 2008 functional level and then add the Windows Server 2022 domain controllers.

The following screenshot shows the available forest functional levels for Windows Server 2022:

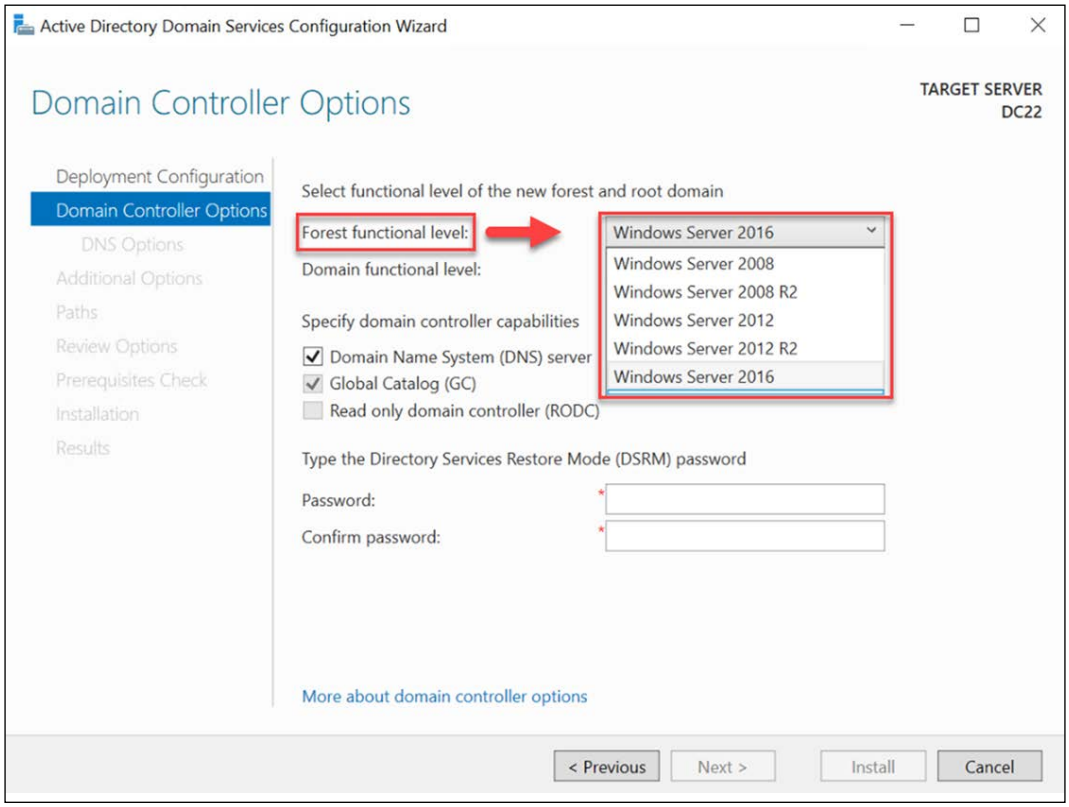


Figure 2.1: Forest functional levels for Windows Server 2019

Next, let's take a look at why the File Replication service was deprecated and what it was replaced with.

The deprecation of the File Replication service

In Windows Server 2000, Microsoft introduced the **File Replication Service** (**FRS**); it was used to replicate the `SYSVOL` AD folder.

SYSVOL is a folder that contains public files from the domain that need to be replicated to other domain controllers in the domain. It contains files required for group policies and user login scripts.

FRS had a lot of issues when it came to replication, especially performance-wise. FRS always replicated the entire file, no matter the kind of changes you've made. This was an issue for AD sites connected by slow WAN links. FRS did not have API or **Windows Management Instrumentation** (**WMI**) support to monitor performance. It also didn't support any health-reporting mechanisms. FRS was replaced by **Distributed File System Replication** (**DFSR**) in Windows Server 2003 R2 and, since Windows Server 2008, DFSR has been the default method used for `SYSVOL` folder replication.

DFSR supports partial file-change (block-level) replication instead of entire file replication. This leads to faster replication and optimized bandwidth usage between AD sites connected by WAN links. It also supports file compression on a per-file-type basis. The number of files that can be transferred (either inbound or outbound) has been increased compared to FRS. DFSR also has a self-healing mechanism for filesystem problems.

With Windows Server 2008 R2, FRS has been deprecated, and if you create a new domain with a Windows Server 2008 functional level or higher, it will use DFSR by default to replicate `SYSVOL`. But if you're migrating from a Windows 2003 domain environment, FRS-to-DFSR migration is also required. Windows Server 2022 domain controllers can only be added to an AD environment that uses DFSR for `SYSVOL` replication. If FRS is still in use, before we add any domain controllers, we need to migrate `SYSVOL` replication from FRS to DFSR. The steps for migrating from FRS to DFSR are covered in one of my previous blog posts, which can be accessed using the following link: `https://bit.ly/3q4Re7E`.

Privileged Access Management (PAM)

Most banks have a safety deposit box service. A safety deposit box is a safe place where you're able to keep belongings that are very valuable. Once you've decided what to protect, you can place it in your safety deposit box at the bank, located in a highly secure facility. When you sign up with a banking service, the bank will provide you with a key card, PIN, or key to open your deposit box. When you need to access your box, first you need to go to the facility and prove who you are. After a successful verification process, you will be allowed to access your deposit box. This facility may have thousands of different deposit boxes with lots of valuable assets, but your key will only give you access to the box that belongs to you. But imagine if there was a master key that could open all the boxes in the entire facility. Which key would have more value to a thief? Your key or the master key that has the "privilege" to open all the safety deposit boxes? Obviously, it would be the master key. This is very similar to how privilege access works in an AD environment. We have different accounts in an AD environment with different levels of access to systems and data. The level of access and permissions can differ from one account to another based on the role or department. But there are certain accounts with "privileges" that can be used to do whatever the user wants in the AD environment without any restrictions. The Domain Admin or Enterprise Admin accounts are great examples of that. If someone has access to one of these accounts, he/she has control over the entire AD environment. It is becoming easier and easier for attackers to get access to a domain account, but after the initial breach, they always try to laterally move and get access to "privileged" accounts. The role of the **Privileged Access Management** (**PAM**) solution is to prevent attackers from gaining access to privileged accounts even if they have a successful initial breach.

PAM holds a prominent position in the modern threatscape. Centrify carried out a survey (`https://bit.ly/3BOPJMX`) of 1,000 IT decision makers in the UK and the USA. According to their report, "*74% of respondents whose organizations have been breached acknowledge that it involved access to a privileged account*." The *2020 RSA Conference Edition of the Attacker Behavior Industry Report*, which was based on data collected from more than five million workloads and devices from customer clouds, data centers, and enterprise environments, revealed that across all industries, 215 privileged attacker behavior detections per 10,000 hosts were observed. It also says that finance and insurance, healthcare, and education organizations exhibited the most privileged access anomaly behaviors. These three industries together account for 47% of all privileged access anomaly behavior detections. Based on their data, RSA also observed 112 lateral movement behaviors per 10,000 hosts. All this data confirms a surge in attacker access to privileged accounts. The above findings emphasize the importance of PAM in the fight against cyber criminals.

Privileged access relates to accounts that have control over a large portion of enterprise identities and business-critical assets. Securing these accounts must be the top priority for any business. If an attacker has privileged access, we can expect the following:

- **High business impact –** When an attacker has privileged access, they have control over enterprise identities and the majority of business-critical assets. That means the attacker can steal intellectual property data, disclose business confidential data, and stop business operations.
- **Financial gain for attackers –** Attackers may also encrypt/gain control of sensitive data and demand money to release it. This can result in a financial and reputational loss for the business.

When we look to secure privileged access, we need to focus on two things.

- Limiting the number of ways a user can gain privileged access
- Protecting and monitoring these ways continuously to identify potential risks

There are two types of access in an environment: user access and privileged access. User access refers to a standard user accessing end-user applications and services. Privileged access refers to a user with privileges performing administrative tasks in business-critical systems. If we can keep these two types of access separate then that's ideal, but in most scenarios that's not possible. Sometimes businesses need to allow standard users to perform administrative tasks. If there is such a requirement, we need to use services and workflows to allow access in a controlled manner using processes such as JIT and JEA in PAM.

First of all, PAM is not a piece of software that we can just install and then be done with it. It is a combination of many things. Before we introduce PAM to an AD environment, we need to do some groundwork, which will be covered later in this section. More importantly, PAM changes the way we work with privileged access. Replacing a product is easy, but changing a process is more complicated and challenging.

The evolution of cyber crime

I started my career in 2003 with one of the largest North American hosting companies. I was a systems administrator at the time, and one of my tasks was to identify hacking attempts and prevent servers from being compromised. In order to do that, I had to review lots of logs on different systems. However, around that time, the intention of most attackers, whether individuals or groups, was to put their names on websites in order to prove that they could cause damage. The average daily number of hacking attempts per server was around 20 to 50.

Some collocation customers were even running their websites and workloads without any sort of protection (even when advised against it). But as time went by, year by year, the number of attempts dramatically increased, and we were Noticing hundreds or thousands of attempts per day. According to a study done by *Cybersecurity Ventures* in 2021, a cyber-attack incident occurs every 11 seconds. That is almost twice the rate in 2019 (every 19 seconds), and four times what it was in 2016 (every 40 seconds). The *Cyber Security Breaches Survey 2020* done by the Department for Digital, Culture, Media and Sport in the UK (`https://bit.ly/3whrR3D`) says "*Almost half of businesses (46%) and a quarter of charities (26%) report having cyber security breaches or attacks in the last 12 months*":

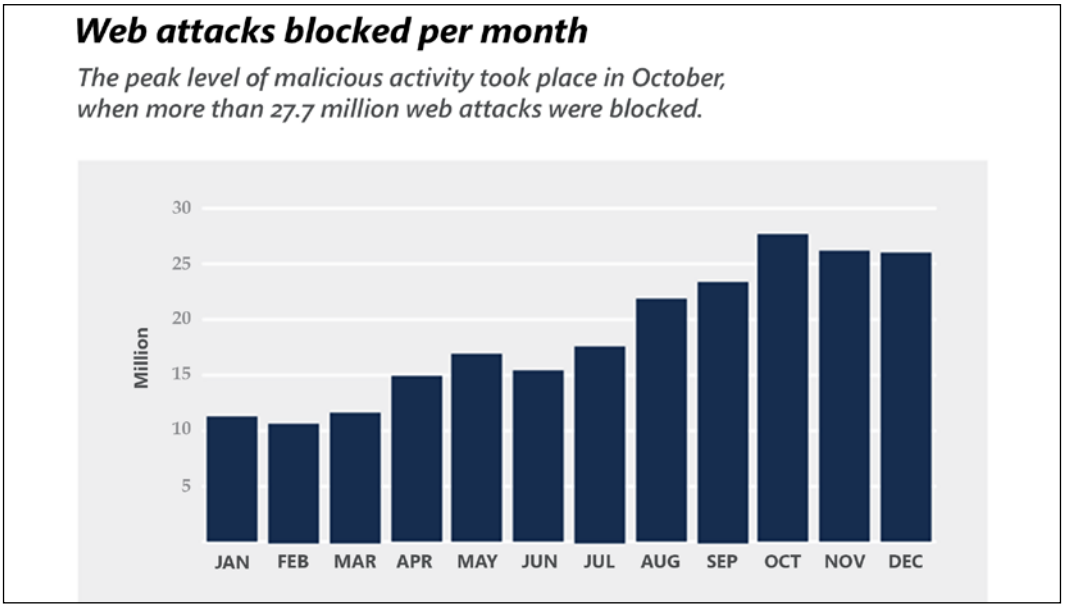


Figure 2.2: Web attacks blocked per month

Not only have the numbers changed, but so has the motive behind the attacks. As I said, in the old days, it was script kiddies who were after fame. Later, as users started to use more and more online services, the focus of the attacks changed to financial gain. These types of attacks are still popular. According to the *Data Breach Investigations Report 2020* by Verizon, 86% of breaches were financially motivated (from 3,950 breaches). Back in 2018, it was 76%.

When considering the progression of threats, after the year 2012, a number of things changed. Attackers started to focus more on *identities*. In the old days, data about a person was stored in different formats. For example, when I walked into my medical center as a child, before I saw the doctor, the administrative staff had to go and find the file with my name on it.

They had a number of racks filled with files and papers, which included patient records, treatment histories, test reports, and more. But now, things have changed: when I walk into the center, no one in administration needs to worry about my file. The doctor can see all of my records from their computer screen with just a few clicks. more and more data about people is being transformed into a digital format. In this healthcare system, I have become an *identity*, and my identity is attached to data, as well as to certain *privileges*. Consider your online banking system: you've got your own username and password to type in when you log in to the portal. This means that you have your own identity in the banking system; once you have logged in, you can access all your accounts, transfer money, make payments, and more. The bank has granted some *privileges* to your identity. With your privileges, you cannot look into your neighbor's bank account. But your bank manager can view your account and your neighbor's account too. This means that the privileges attached to the bank manager's identity are different. The amount of data that can be retrieved from the system depends on the level of access.

In addition to this, one digital identity can be integrated with multiple systems. Businesses use different systems for different operations; for example, email systems, CMS systems, or billing systems. Each of these systems holds data. To make operations smoother, these systems may be integrated with one directory service, such as Microsoft AD. This will allow users to have a single sign-on experience instead of using different identities for each and every application. This is making identities more and more powerful. For an attacker, consider what is worth more—focusing on one system, or targeting an identity attached to data and privileges on many different systems? Which one would cause more damage?

Additionally, when it comes to identities, is it all just about usernames and passwords? Well, no, it's not; data that belongs to a digital identity has the most value. It could be personal data, credit card information, medical records, travel records, and so on. If the digital identity belongs to a corporate system, it can give access to sensitive corporate data or even state secrets. Let's look into some recent well-known cyber-attacks.

Recent cyber-attacks

In June 2019, the **American Medical Collection Agency** (**AMCA**) had a data breach, exposing the **personal and payment information of 20 million patients** after attackers hacked their web payment portal. The attack exposed personal data such as names, dates of birth, addresses, phone numbers, dates of service, providers, balance information, and credit card/bank account details. AMCA filed for bankruptcy later as the breach led to both financial and legal issues.

In August 2019, Capital One, one of the largest banking institutions in the United States, had a data breach, exposing the personal information of over **106 million credit card applicants** between 2005 and 2019.

In January 2020, Travelex, a London-based foreign exchange company, experienced an attack by the Sodinokibi ransomware group. Travelex had a conversation with the group but refused to pay the ransom demand of **$6 million** in exchange for the decryption keys. The attackers threatened to publish 5 GB of **customers' personal information** that had been stolen and exfiltrated prior to encryption.

In May 2020, the budget airline EasyJet was hacked, affecting 9 million customers and exposing the details of over 2,000 credit and debit cards. EasyJet said the attacker had gained access to and stolen **customers' email addresses and travel details**.

In June 2020, the University of California experienced a ransomware attack. It affected the university's School of Medicine **COVID-19 research data** and the university paid around **$1 million** to get access to the data again.

In August 2020, the Maze ransomware group published **50.2 GB of data** that was stolen from LG's internal network. They also published 25.8 GB of data belonging to Xerox's network.

As we can see in the above examples, personal data and corporate data have high value when it comes to financially motivated attacks. To access this data, attackers need some sort of an initial breach. This is where usernames, passwords, and privileges come into the picture. It is also important to understand that it's not only about permissions attached to an identity; individual identities themselves are more important as well.

At the time of writing, it has been a few months since Donald Trump's defeat in the 2020 US election. We can see how much news can be generated from a single tweet of his. It isn't necessary to have special privileges to post a tweet; it is the identity of the person posting it that makes that tweet important. On the other hand, if that Twitter account got hacked and someone posted a fake tweet on behalf of the actual person who owned it, what kind of damage could that cause to the whole world?

Whenever a cyber-attack occurs, the most common responses from businesses include "*Those attacks were so sophisticated!*", "*It was too complex to identify!*", "*They were so clever!*", "*It was a zero-day attack,*" and so on. But is that really true?

Zero-day attacks are based on system bugs and errors unknown to vendors. The latest report shows that the average time it takes to detect a breach is less than 7 days, and it takes 1 day to release a patch (*Symantec Internet Security Threat Report, 2016*).

According to NIST security vulnerability trends in the 2020 report at `https://bit.ly/3bMk6sO`, there has been a growth in the number of low-complexity vulnerabilities over the last 3 years. In 2020, 63% of reported vulnerabilities were low-complexity. Low-complexity means that an attacker with low technical skills could exploit such a vulnerability. This proves that attackers are still after low-hanging fruit:

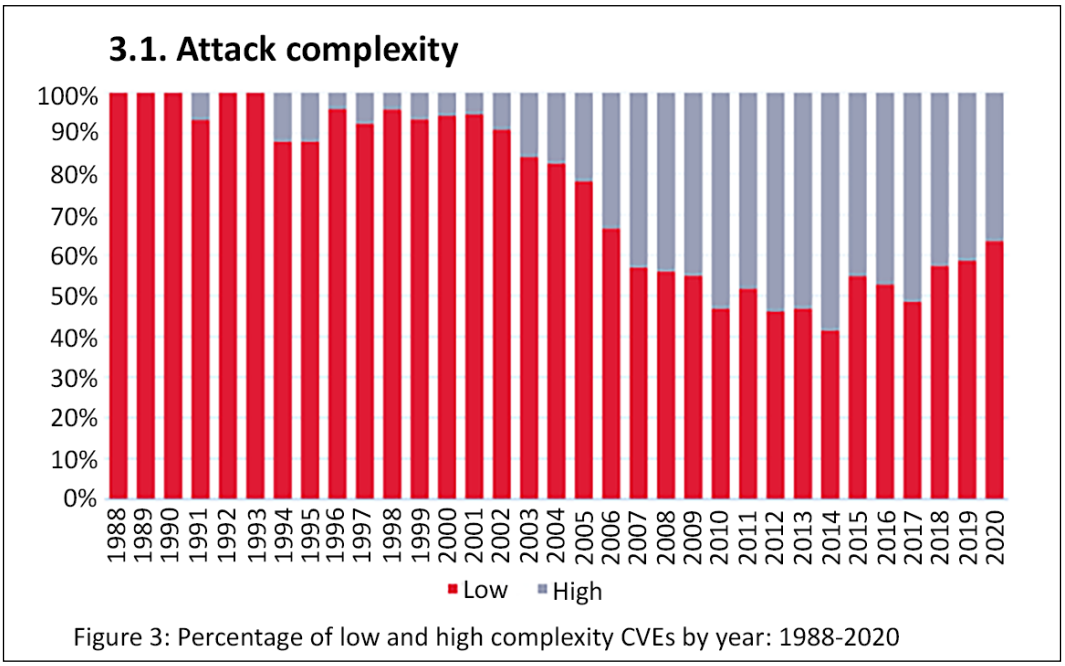


Figure 2.3: Attack complexity

Microsoft AD is a leader in providing identity and access management solutions. In all this constant news about identity breaches, AD's name also appears. Interestingly, people have started to question why Microsoft can't fix these problems. But if we evaluate these problems, it's obvious that just providing a technology-rich product is not enough to solve the issues. With each new server OS version, Microsoft releases a new AD version. With every release, there are new features to improve identity infrastructure security. However, when I go to work on an AD project, I find that the majority of engineers don't even follow the security best practices defined by AD versions that were released 10 years ago!

Let's consider a car race: race categories are usually based on engine capacity; for example, 3 L, 5 L, and so on. In a race, most of the time, it's the same models or the same manufacturer's cars racing together. If it's the same manufacturer's cars, and if they all have the same engine capacity, how is it that one driver wins and the others lose? Well, it's the car's tuning and the driver's skills that decide the winner.

If AD DS 2022 can address all identity threats, that's really good, but simply providing a product or technology doesn't seem to have worked so far. That's why we need to change the way we think about identity protection. We should not forget that we are fighting against human adversaries—the tactics, methods, and approaches they use are changing every day. The products we use do not have such frequent updates, but we can change their ability to execute an attack on an infrastructure by understanding the fundamentals and using products, technologies, and workflows to prevent it.

A typical AD attack

Before we move on to identity-theft prevention mechanisms, let's take a look at a typical AD attack:

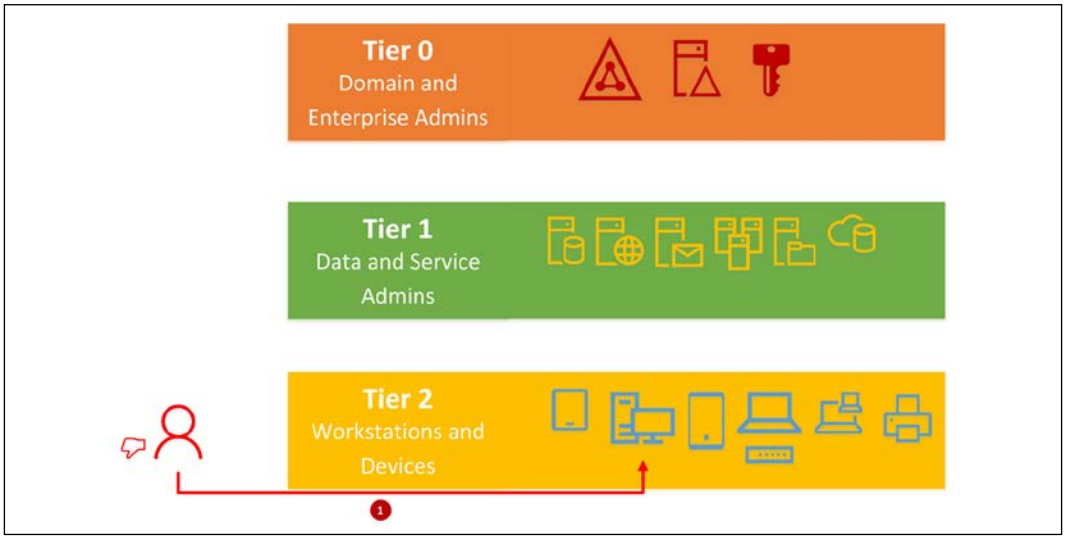


Figure 2.4: AD attack – initial breach

We can group user accounts into three tiers based on the privileges they have. All identity attacks start by gaining some kind of access to a system before the attackers then move laterally until they have the keys to the kingdom, that is, the domain and enterprise admin credentials. Then, they have full ownership of the entire identity infrastructure.

As the preceding diagram shows, the first step of an identity attack is to get initial access to the system. Attackers usually do not target the domain or enterprise admin account first. Getting access to a standard user account is much easier than doing so for a domain admin account; all the attacker needs is some kind of intial access. For that, even now, the most common attack technique is to send out a phishing email.

According to the ***Verizon – Data Breach Investigations Report 2020***, 20% of breaches are based on phishing attacks. The *Cyber Security Breaches Survey 2020* done by the Department for Digital, Culture, Media and Sport in the UK (`https://bit.ly/3whR6Tc`) says 86% of businesses experience phishing attacks. It's typical for someone to still fall for a phishing email and click on it. Then, once the attacker has some sort of access to your identity infrastructure, the next step for them is to start moving laterally to gain more privileges. How many of you have completely eliminated local administrator accounts from your infrastructure? I'm sure the answer will be almost none of you. Users ask for software installations and system-level modifications to their systems frequently, and most of the time, engineers end up assigning local administrator privileges to users. If a compromised account has local administrator access, it is easy for the attacker to move to the next level.

If the compromised account lacks local administrator access, then the attacker will make the compromised system misbehave. So, who will come to the rescue? Well, the IT Administrators, of course. In lots of organizations, IT Administrators are domain admins. If not, they're at least local administrators on systems. So, when they receive a call about a misbehaving computer, they may use a **Remote Desktop Protocol** (**RDP**) connection or log in locally using a privileged account. RDP always sends your credentials in plain text. If the attacker is running a password-harvesting tool, it's extremely easy to capture the credentials. You may be wondering how a compromised standard user account can execute such programs. Well, it just so happens that Windows operating systems do not prevent users from running any application they want in their user context. Standard user contexts do not allow users to change any system-level settings, but they can still run scripts or user-level executables:

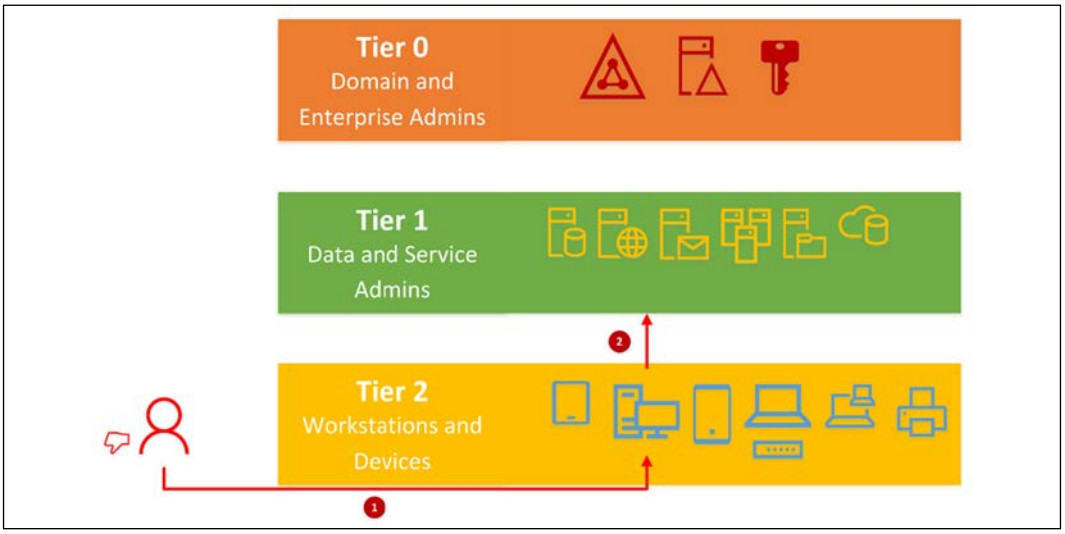


Figure 2.5: AD attack – lateral movement

Once an attacker has gained access to an account in an organization, the next level of privileges to own will be **Tier 1**. This is where the application administrator, data administrator, and SaaS application administrator accounts reside. In modern-day infrastructure, we have too many administrators. First, we have domain administrators and enterprise administrators, and then we have local administrators. Different applications running on an infrastructure have their own administrators, such as Exchange administrators, SQL administrators, and SharePoint administrators. Other third-party applications, such as CMS applications or billing portals, may have their own administrators. Additionally, if you are using cloud services, SaaS applications have another set of administrators. So, are we really aware of what's going on in these accounts? Mostly, engineers only worry about protecting domain admin accounts but, at the same time, forget about the other kinds of administrators in the infrastructure. Some of these administrator roles can cause more damage to a business than a domain admin. These applications and services decentralize access management in the organization. In order to move laterally with privileges, these attackers only need to log in to a machine or server where administrators log in. **Local Security Authority Subsystem Service** (**LSASS**) stores credentials in its memory for active Windows sessions. This avoids the hassle of users having to enter credentials for each and every service they access. It also stores Kerberos tickets. This allows attackers to perform pass-the-hash attacks and retrieve locally stored credentials. The decentralized access management of administrator accounts makes this process easier.

There are features and security best practices that can be used to prevent pass-the-hash attacks in an identity infrastructure. I will explain them in detail in *Chapter 16*, *Active Directory Security Best Practices*.

Another problem with these types of accounts is that once they become service admin accounts, they can eventually become domain or enterprise admin accounts. I have seen engineers create service accounts and, when they can't figure out the exact permissions required for the program, they go ahead and add them to the domain admin group as an easy fix. However, it's not only infrastructure attacks that can expose such credentials. Service admins are attached to the application too, so compromised applications can also expose identities:

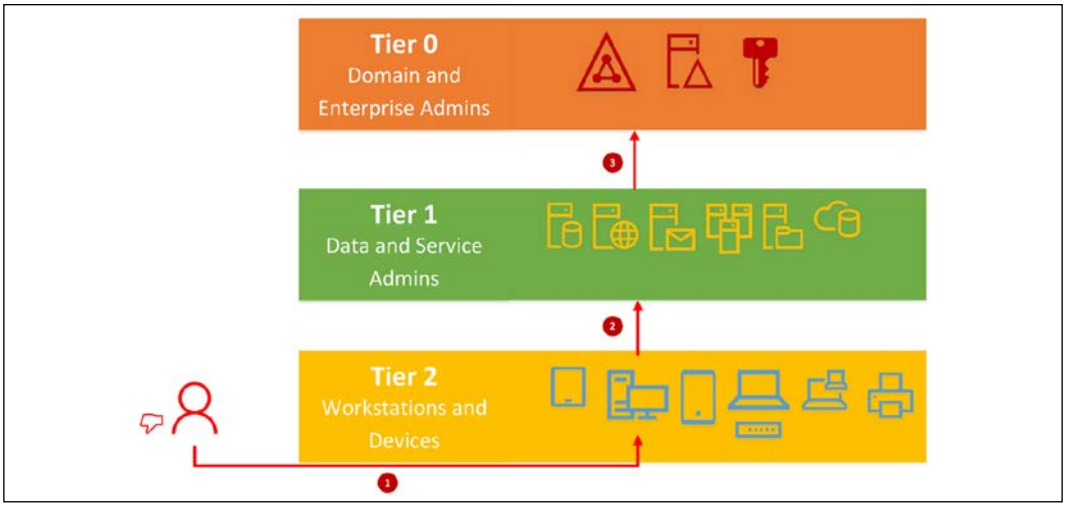


Figure 2.6: AD Attack – access to privileged accounts

Tier 0 is where the domain and enterprise admins operate. This is what the ultimate target of an identity attack is; once the attacker obtains access to **Tier 0**, they own your entire identity infrastructure. The latest reports show that, after the initial breach, it only takes less than 48 hours to gain **Tier 0** privileges. According to these reports, once attackers gain access, it takes at least 7-8 months to identify the breach. This is because once they have the highest privileges, they can create backdoors, clean up logs, and hide forever if needed. The systems we use always treat administrators as trustworthy people. This is no longer the case in the modern world; how many times do you check your system's logs to see what your domain admins are doing? Even though engineers look at the logs of other users, rarely do any of them check domain admin accounts. The same thing applies to an internal security breach too: most people are good people, but you never know. Many high-profile identity attacks have proven this already.

When I discuss identity protection with customers, these are the common comments I hear:

- We have too many administrator accounts
- We do not know how many administrator accounts we've got
- We have fast-changing IT teams, so it's hard to manage permissions
- We do not have visibility over administrator account activities
- If there is an identity infrastructure breach or attempt, how do we identify it?

Answers to all these questions are included in PAM. As I mentioned at the beginning of the chapter, this is not one product; it's a workflow and a new way of working. The main steps and components of this process are as follows:

1. Apply pass-the-hash prevention features to the existing identity infrastructure (for more information, you can refer to *Chapter 16*, *Active Directory Security Best Practices*).
2. Use Microsoft Defender for Identity (formerly known as Azure Advanced Threat Protection) to monitor the domain controller traffic to identify potential real-time identity infrastructure threats (refer to *Chapter 16*, *Active Directory Security Best Practices*).
3. Install and configure Microsoft Identity Manager 2016—this product enables us to manage privileged access to an existing AD forest by providing task-based, time-limited privilege access. I will explain this in detail later in this chapter using examples.

What does PAM have to do with AD DS 2022?

AD DS 2022 allows time-based group membership, which makes the process outlined above possible. This feature was first introduced with AD DS 2016. A user is added to a group with a **Time-to-Live** (**TTL**) value and, once it expires, the user is removed from the group automatically. For example, let's assume your CRM application has administrator rights assigned to the `CRMAdmin` security group. The users in this group only log in to the system once a month to do some maintenance. But the admin rights for the members in that group remain untouched for the remaining 29 days, 24/7. This provides enough of an opportunity for attackers to try and gain access to privileged accounts. So, if it's possible to grant access privileges for a shorter time period, isn't that more useful? Then, we can be assured that, for the majority of the days in a month, the CRM application does not run the risk of being compromised by an account in the `CRMAdmin` group.

What is the logic behind PAM?

PAM is based on the JIT administration concept. Back in 2014, Microsoft released the PowerShell toolkit, which allows JEA.

Let's assume that you are running a web server in your infrastructure; as part of the operation, you need to collect some logs every month to make a report. You've already set up a PowerShell script for this purpose. Someone in your team needs to log in to the system and run it. In order to do so, you require administrative privileges. Using JEA, it is possible to assign the required permissions for the user to run only that particular program. This way, there's no need to add the user to the domain admin group. The user will not be allowed to run any other program with the permission assigned as it is; JIT administration is bound by *time*. This means users will have the required privileges only when they need them; they will not hold privileged access rights all the time.

PAM operations can be divided into four major steps, as shown in the following diagram:

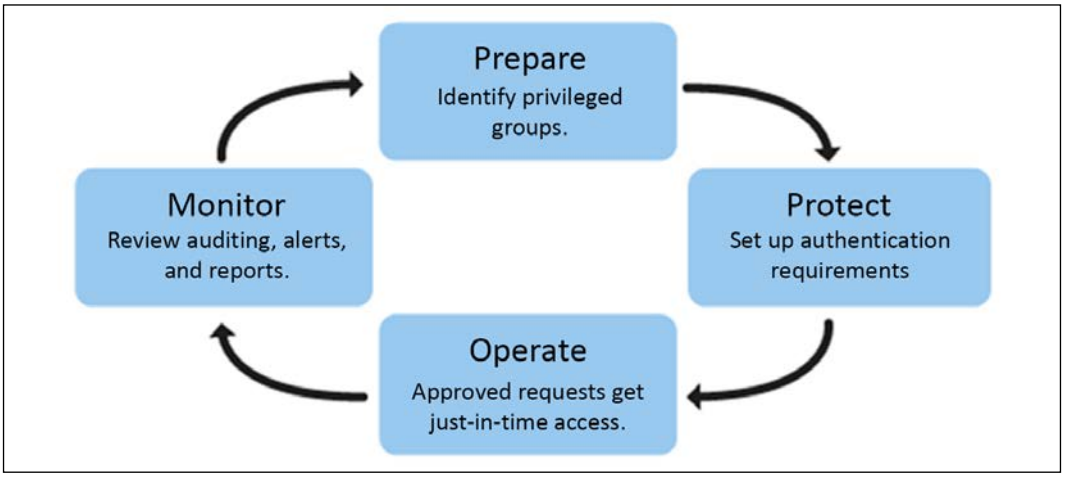


Figure 2.7: The PAM lifecycle

Let's take a look at these four major steps:

- **Prepare**: The first step is to identify the privileged access groups in your existing AD forest and start to remove users from them. You may also need to make certain changes to your application infrastructure to support this setup. For example, if you assign privileged access to user accounts instead of security groups (in applications or services), this will need to change. After an initial assessment, we need to set up equivalent groups in a *bastion forest* without any members.

When setting up **Microsoft Identity Manager** (**MIM**), a bastion forest will be used to manage privileged access in an existing AD forest. This is a special forest and cannot be used for other infrastructure operations. This forest runs on a minimum of a Windows Server 2012 R2 AD forest functional level. When an identity infrastructure is compromised and attackers gain access to **Tier 0**, they can hide their activities for months or years. But how can we be sure that our existing identity infrastructure has not been compromised already? Well, if we implement our solution in the same forest, it will not achieve its core targets. Additionally, domain upgrades are painful, requiring time and money. But with a bastion forest, this solution can be applied to your existing identity infrastructure with minimal changes.

- **Protect**: The next step is to set up a workflow for authentication and authorization. We need to define how a user can request privileged access when required. This can be done using a MIM portal or through an existing support portal. It is possible to set up a system to use **Multi-Factor Authentication** (**MFA**) during this request process to prevent any unauthorized activity. Additionally, it's important to define how the requests will be handled. It can be either an automatic or manual approval process.
- **Operate**: Once the privileged access request is approved, the user account will be added to the security group in the bastion forest. The group itself has an SID value. In both forests, the group will have the exact same SID value. Therefore, the application or service will not see a difference between the two groups in two different forests. Once the permission is granted, it will only be valid for the time defined by the authorization policy. Once it reaches the time limit, the user account will be removed from the security group automatically.
- **Monitor**: PAM provides visibility over privileged access requests. On each and every request, events will be recorded, and it is possible to review them and also generate reports for audits. This helps to fine-tune the process and also identify any potential threats.

Let's examine how it really works:

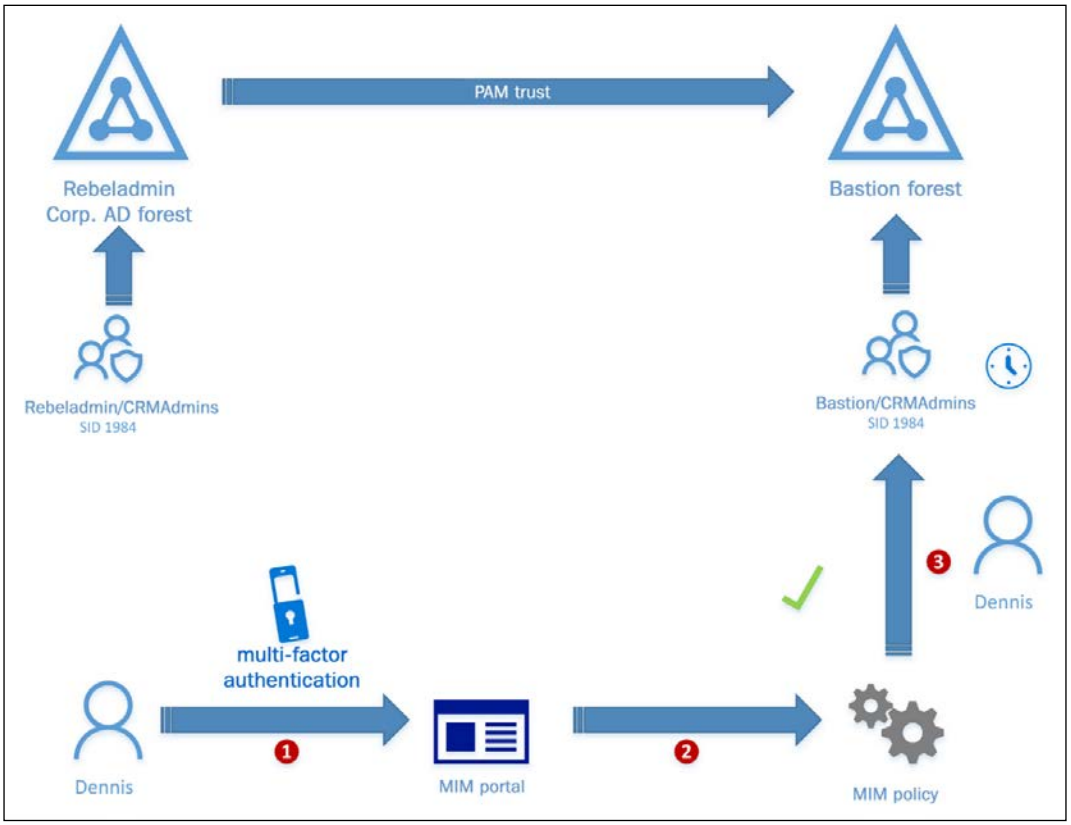


Figure 2.8: PAM in action

Rebeladmin Corp. uses a CRM system for its operations. The application has the administrator role and **Rebeladmin/CRMAdmins** security group assigned to it. Any member of that group will have administrator privileges to the application. Recently, PAM has been introduced to Rebeladmin Corp. As an engineer, I have identified **Rebeladmin/CRMAdmins** as a privileged group and am going to protect it using PAM. The first step is to remove the members of the **Rebeladmin/CRMAdmins** group. After that, I set up the same group in the bastion forest.

It's not just that the name is the same, but both groups have the same SID value: **1984**:

1. User **Dennis** used to be a member of the **Rebeladmin/CRMAdmins** group and was running monthly reports. At the end of one month, he tried to run a report and found that he did not have the required permissions. The next step for him was to request the required permission through the MIM portal. According to the policies, as part of the request, the system wants **Dennis** to use MFA. Once **Dennis** verifies the PIN, the request is logged in the portal.
2. As an administrator, I receive an alert about the request, and I log in to the system to review the request.
3. It's a legitimate request, so I approve his access to the system for 8 hours.

Then, the system automatically adds the user account for Dennis to the **Bastion/CRMAdmins** group. This group has the same SID value as the production group. Therefore, a member of the **Bastion/CRMAdmins** group will be treated as an administrator by the CRM application. This group membership contains the TTL value too. After 8 hours from the approval time pass, Dennis's account will be automatically removed from the **Bastion/CRMAdmins** group. In this process, we didn't add any members to the production security group, which is **Rebeladmin/CRMAdmins**. So, the production forest stays untouched and protected.

Here, the most important thing we need to understand is that the legacy approach to identity protection is no longer valid. We are up against human adversaries. Identity is our new perimeter in the infrastructure and, to protect it, we need to understand how our adversaries are attacking it and stay a step ahead.

Time-based group memberships

In the previous section, I explained PAM features in AD DS 2022. Time-based group membership is a part of that broader topic. It allows administrators to assign temporary group membership, which is expressed by a TTL value. This value will be added to the Kerberos ticket. It is also called the **expiring links** feature. When a user is assigned temporary group membership, their login Kerberos **Ticket-Granting Ticket** (**TGT**) lifetime will be equal to the lowest TTL value they have. For example, let's assume that you grant temporary group membership to user A to be a member of the domain admin group. It is only valid for 60 minutes. But the user logs in 50 minutes after the original assignment and only has 10 minutes left as a member of the domain admin group. Based on this, the domain controller will issue a TGT that is only valid for 10 minutes to user A.

This feature is not enabled by default. The reason for this is that to use this feature, the forest functional level must be Windows Server 2016.

Additionally, once this feature is enabled, it cannot be disabled.

Let's examine how this works in the real world:

1. I have a Windows domain controller installed and it is running with the Windows Server 2016 forest functional level. This can be verified using the following PowerShell command:

```
Get-ADForest | fl Name,ForestMode
```

2. Then, we need to enable the expiring links feature. This can be enabled using the following command:

```
Enable-ADOptionalFeature 'Privileged Access Management Feature'
-Scope ForestOrConfigurationSet -Target rebeladmin.com
```

   The `rebeladmin.com` domain name can be replaced with your **Fully Qualified Domain Name** (**FQDN**).

3. I have a user called Adam Curtiss who I need to assign domain admin group membership to for 60 minutes; take a look at the following command:

```
Get-ADGroupMember "Domain Admins"
```

   It lists the current members of the domain admin group:

```
PS C:\Windows\System32> Get-ADGroupMember "Domain Admins"

distinguishedName : CN=dfrancis,CN=Users,DC=rebeladmin,DC=com
name              : dfrancis
objectClass       : user
objectGUID        : 196f45a5-d966-4063-b947-b598cd15e305
SamAccountName    : dfrancis
SID               : S-1-5-21-2368539100-1613354624-3645015003-500
```

Figure 2.9: Domain admin group membership

4. The next step is to add Adam Curtiss to the domain admin group for 60 minutes:

```
Add-ADGroupMember -Identity 'Domain Admins' -Members 'acurtiss'
-MemberTimeToLive (New-TimeSpan -Minutes 60)
```

5. Once that has run, we can verify the remaining TTL value for the group membership using the following command:

```
Get-ADGroup 'Domain Admins' -Property member
-ShowMemberTimeToLive
```

The following screenshot illustrates the output for the preceding command:

```
PS C:\Windows\System32> Get-ADGroup 'Domain Admins' -Property member -ShowMemberTimeToLive

DistinguishedName : CN=Domain Admins,CN=Users,DC=rebeladmin,DC=com
GroupCategory     : Security
GroupScope        : Global
member            : {<TTL=3435>,CN=Adam Curtiss,CN=Users,DC=rebeladmin,DC=com,
                    CN=dfrancis,CN=Users,DC=rebeladmin,DC=com}
Name              : Domain Admins
ObjectClass       : group
ObjectGUID        : a39fe22a-9303-41e2-9560-a7889083ca1a
SamAccountName    : Domain Admins
SID               : S-1-5-21-2368539100-1613354624-3645015003-512
```

Figure 2.10: TTL for group membership

6. When I log in as the user and list the Kerberos ticket, it shows the renewal time as less than 60 minutes. This is because I've logged in a few minutes after being granted permission:

```
Administrator: PowerShell 7 (x64)
PowerShell 7.1.2
Copyright (c) Microsoft Corporation.

https://aka.ms/powershell
Type 'help' to get help.

PS C:\Users\acurtiss> klist

Current LogonId is 0:0x2168f2

Cached Tickets: (1)

#0>     Client: acurtiss @ REBELADMIN.COM
        Server: krbtgt/REBELADMIN.COM @ REBELADMIN.COM
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40e10000 -> forwardable renewable initial pre_authent name_canonicalize
        Start Time: 2/28/2021 22:09:50 (local)
        End Time:   2/28/2021 23:01:25 (local)
        Renew Time: 2/28/2021 23:01:25 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96
        Cache Flags: 0x1 -> PRIMARY
        Kdc Called: DC01
PS C:\Users\acurtiss>
```

Figure 2.11: Kerberos ticket renewal time

Once the TGT renewal period is crossed, the user will no longer be a member of the domain admin group.

Windows Hello for Business

The most common way of protecting access to a system or resource is to introduce authentication and authorization processes. This is exactly what AD does as well; when a user logs in to a domain-joined device, AD first authenticates the user to see whether they're the user they claim to be. Once authentication is successful, it then checks what the user is allowed to do (authorization). For the authentication process, we use usernames and passwords. This is what all identity infrastructure attackers are after. They need some kind of username and password to get into the system. A password is a symmetric secret that is transmitted to the server every time we authenticate. When passwords appear in different systems, they can be stolen or intercepted on transmission. Back in 2004 at the RSA Security conference, Bill Gates said "People use the same password on different systems; they write them down and they just don't meet the challenge for anything you really want to secure." Over the years this statement has been proven over and over. Passwords are no longer secure. Passwords are breakable. So, if passwords are failing, what else we can do to improve the security of the authentication process? MFA can add another layer of security to the authentication process. However, MFA doesn't eliminate the requirement for passwords.

With Windows 10, Microsoft introduced its new biometric sign-in system. Windows Hello allows us to use face recognition, fingerprints, or a PIN for authentication. After the system identifies its user as legitimate, the user still needs to authenticate to be allowed access to resources. Windows Hello provides strong two-factor authentication instead of passwords. The user needs a specific device and biometric authentication/PIN to be allowed access.

PINs are local to a device. If someone steals a password, they can use it in many places within a network or over the internet. But even if someone has stolen a PIN, it is useless unless they have access to the device where the PIN was generated. PINs use a **Trusted Platform Module** (**TPM**) chip on the device, which is designed to carry out cryptographic operations. A TPM can securely store artifacts that are used to authenticate with a system, such as encryption keys. When a PIN is created, it uses an asymmetric key pair authentication model, which means the credentials never leave the device (the system uses two keys to encrypt). The private key will always be stored on the TPM module. This chip is protected by multiple physical security mechanisms to make sure it is tamper-resistant. Similar to passwords, PINs are also guessable. But even if an attacker has a PIN, it will not useful to them unless they have access to the physical device. So, the preferred approach is to use PINs instead of passwords. Windows Hello also supports password-less authentication using FIDO2 security keys.

The Windows Hello feature eliminates the traditional method of using a username and password and provides a robust, secure, and future-proof way of authentication.

Time sync improvements

Time accuracy is important for AD infrastructures to maintain Kerberos authentication between users and domain controllers. Currently, the time accuracy between two parties should be less than 5 minutes. In an AD environment, domain members sync time with domain controllers (that is, the **Primary Domain Controller** (**PDC**), a domain controller in the root forest, or a domain controller with the **good time server**, or `GTIMESERV`, flag) to maintain accurate time across the environment.

However, sometimes, this doesn't work as expected. As an example, virtual servers sync time with their hosts, which can cause accuracy issues. Depending on the network topology, the reply packets for time requests can take longer to reach the requester. This can also cause accuracy issues between the domain controller and the client. Mobile devices and laptops may not connect with the domain very often, which can also lead to time accuracy issues.

Time accuracy impacts an organization's business and operations in different ways:

- AD replications between domain controllers are the primary requirement of a healthy AD infrastructure. Inaccurate time syncs create replication issues.
- Credit card processing requires 1-second accuracy, according to industry standards.
- Government regulations such as FINRA (in the USA) and ESMA/MiFID II (in the EU) demand a time accuracy level of 100 µs (milliseconds).
- Claiming accuracy is not enough. You must also be able to "trace" your time back to an authoritative time source and prove accuracy.

From Windows Server 2016, Microsoft made several improvements to maintain accurate time synchronization across infrastructures. Its improved algorithms will mitigate the impact of **Network Time Protocol** (**NTP**) data accuracy, resulting in network congestion and network latency. It also uses an improved API for accurate time references. With these improvements, it can provide 1-microsecond time accuracy.

Microsoft made further changes to improve the time accuracy with Windows Server 2019 and it is continued to Windows Server 2022. As the earth's rotation slows, UTC differs from mean solar time. Once UTC reaches a difference of 0.9 seconds, a leap second is inserted to keep UTC in sync with mean solar time. This process started in 1972 and usually occurs every 18 months. Windows Server 2019 & 2022 support the leap second. With this important change, Windows Server 2019 & 2022 comply with US and European Union accuracy and traceability regulatory requirements.

Starting from Windows Server 2019, we have a new time provider called **Precision Time Protocol** (**PTP**). Until now, Windows operating systems have only used NTP as the time synchronization method. During the time sync process, the timing packets pass through a number of network devices. As with any other packets, the status of the network and devices may add latency to the timing packets. NTP doesn't consider this latency. This latency can have an impact on the time accuracy. PTP enables network devices to add the latency introduced by each network device to the timing measurements. This makes the time more accurate. However, by default, Windows Server 2019 & 2022 still use NTP as the time synchronization method. This is because PTP requires network configuration changes. But if the business requires high time accuracy, now we have a solution.

Once an operating system receives a timing packet, it will be processed by the operating system's networking stack before it reaches the time service. Each component in the networking stack adds latency, which can affect the accuracy of the time. This delay can be between 30 µs and 200 µs. That doesn't sound like a lot, but for systems that demand less than 100 µs (due to compliance reasons), it is a lot. To address this problem, starting from Windows Server 2019, it is possible to timestamp timing packets before and after the "Windows networking components." This allows us to calculate software delays on timing packets.

PowerShell 7

In Windows Server 2022, we still have PowerShell version 5.1. But the future of PowerShell lies with the cross-platform PowerShell version. When we open the PowerShell console in Windows Server 2022, we can see a message saying `Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows` and this refer to PowerShell 7. Back in 2017, Microsoft released its first cross-platform PowerShell version, which was PowerShell Core 6.0. It is supported to run on Windows, macOS, and Linux operating systems. It was built on .NET Core 2.x. and it was the first release of PowerShell to be made under an open-source license (MIT).

On November 11, 2020, Microsoft announced the general availability of PowerShell 7. This is built on .NET 5. Most of the modules used in Windows PowerShell 5.1 already work with PowerShell 7, including Azure PowerShell and AD. PowerShell 7 can run side by side with PowerShell 5.1. In this book, I am going to use PowerShell 7 to make it more future-proof and encourage users to use the cross-platform PowerShell instead of the legacy version.

So before we proceed with the upcoming chapters, please make sure to install PowerShell 7 in your test environment. The installation steps for PowerShell 7 can be found here: `https://bit.ly/2ZXsAuI`.

Summary

In this chapter, we looked at the features of AD DS 2022. Even though there are no changes from AD DS 2019, we looked at Microsoft's approach to PAM and why it is more important now than ever. There are many different things to be taken care of in an AD environment, which will be explained further in later chapters. AD DS helps to protect identity infrastructures from novel adversaries, as traditional techniques and technologies are no longer valid in the face of rising threats. We also explored the improvements made to time synchronization to maintain time accuracy across an AD domain.

In the next chapter, we are going to look at designing AD infrastructures.

3

Designing an Active Directory Infrastructure

The **Active Directory** (**AD**) role installation process has been made easy in the last few versions of Windows Server operating systems. Even if you don't have advanced knowledge of AD, with a few clicks, you can install an **Active Directory Domain Services** (**AD DS**) role on a server. But there is a lot more to think about before you install your first domain controller in an IT infrastructure. First, we need to evaluate our business requirements. We need to evaluate our security/compliance requirements, and then we can decide what AD features/services to enable and where to place them in the infrastructure. We need to get the fundamentals of the design correct before the implementation phase.

In this chapter, we are going to learn how to design an On-prem and Hybrid identity Infrastructure properly. In the design phase, we will be looking at information gathering techniques, risk mitigation, capacity planning, component placement, and design best practices.

This chapter covers the following topics:

- Designing a forest structure
- Designing a domain structure
- Designing an OU structure
- Designing the physical topology of Active Directory
- Global catalog server placement
- Designing a hybrid identity

Before we dive into the design concepts, first we need to understand what makes a good system and what we need to consider when designing an Active Directory structure.

What makes a good system?

Fish-keeping is an interesting hobby and when I was a kid I really enjoyed it. I had a to identify issues few fish tanks with different varieties of fish. Me and my dad used to make the fish tanks at home. Before creating a tank, first, we'd decide what size of tank we needed. The type and the thickness of the glass depended on the capacity of the tank. Then we needed the correct glue to assemble the tank.

When assembling, the main goal was to make it waterproof. However, even though we used the correct materials and equipment, it wasn't always waterproof on the first attempt. If there was a leak, we needed to go back and fix it. Once the tank was completely waterproof, we would think about creating a roof, lighting, and decorating inside the tank.

When we are designing a system, it is much like waterproofing a fish tank or a boat. We may not succeed on the first attempt. We need to leave room for faults. We may have to revisit the same task a few times before completely waterproofing the tank or the boat. Once the waterproofing process is completed, we need to maintain it as well. With time, the glue may start to come off and leaks may appear again.

There are a few things we can learn from the above example:

- Don't hesitate to go back to the drawing board when designing a system. Leave room for mistakes and failures.
- Invest in the correct equipment or services.
- Continuously monitor a system to identify issues.
- Have a plan to recover from issues.

These principles apply to an Active Directory design as well. As engineers, we are rarely assigned to build an Active Directory setup from scratch. We're usually assigned to expand or maintain an existing installation. But either way, it's important to understand the logic and key elements of the design.

When I work on Active Directory projects, one of the common questions I get from customers is *do you think our design is correct?* On most occasions, I can't give a yes or no answer. Instead, it's usually a *yes* and *no* answer.

The reason for this is that it may not be valid for the current requirements of the organization, but it was a valid design for the same organization some time ago.

If I ask their IT managers or engineers questions such as *why did you put this domain controller here? or why did you configure this domain controller like this?*, most of the time, they have a valid reason to justify it. It is difficult to draw the line between the right and wrong design unless the design's fundamentals are right or wrong.

If it's not a fresh design, there are occasions on which you may need to revise or redesign the existing Active Directory topology.

New business requirements

When designing an Active Directory environment from the ground up, it would be ideal if the business could confirm what it will become in 10-20 years' time. Then, you would know what to do. But this only happens in a perfect world. It is obvious that, over time, business requirements and technology change. When organizational changes happen, it may affect the existing design too. It could be a change such as introducing a new department, hiring more people, introducing a new business acquisition, or doing business mergers. Some of these changes may be easy to implement, and some may require large, organization-wide changes. These changes can vary from creating a new organization unit to introducing a new Active Directory forest.

For example, Rebeladmin Corp. merges with My-Learning Inc. The management of Rebeladmin Corp. needs a centralized IT administration and sharing of resources between the two organizations. Both organizations maintain their own Active Directory forests. The My-Learning Inc. forest is beyond the Rebeladmin Corp.'s access management security boundary. In order to merge them, we need to create a forest trust or domain trust relationship between the two Active Directory setups. With these changes, Rebeladmin Corp. will have a new security boundary and new assets to manage.

Correcting legacy design mistakes

Correcting legacy design mistakes is pricey for organizations. Most of the time, they end up restructuring domain infrastructures. Recently, I was talking to a company about a domain restructure. It's a multimillion-dollar trading company with offices across the world. The problem was that when they designed the AD infrastructure a long time ago, they used a **single label domain** (**SLD**) name as the primary domain.

SLDs are domain names that don't have DNS suffixes such as `.com`, `.org`, or `.net`. After some time, they realized the limitations of SLDs and didn't fix them as that required some *administrative* changes. No one wanted to have the responsibility, either. The company kept growing, and instead of fixing the fundamental problem, they kept introducing new forests and new domains.

In the end, it became almost unmanageable, with multiple domains and forests, which didn't make sense. Therefore, they had no option other than redesigning the whole Active Directory structure since the organization had decided to move a majority of workloads to the Azure cloud. But it was a very costly and painful change. If you can snip off the weeds growing in your garden with your hands, do it right away; don't wait until you have to use a saw.

There can be design mistakes in any system. This can be due to a lack of knowledge, a lack of resources, or even due to a lack of funding. Sometimes, we recognize things as *mistakes* or *design issues* as they no longer match our business or operational requirements. As an example, we know that it is recommended to use at least two domain controllers in a site for high availability. A start-up company only used one domain controller in the beginning as they were able to afford the downtime of a domain controller. After a couple of years, this is recognized as a *design issue* because the growth of the company and the importance of business operations can no longer afford the downtime of a domain controller. It is important to evaluate the existing design with new business requirements periodically and see what you can do to accommodate those.

Gathering business requirements

Before we start to figure out how many forests, domains, and domain controllers to create, we need to gather some data to help us make an accurate design that agrees with the core business requirements.

Understanding the organizational structure correctly is vital to designing any access management system. An organizational chart is a good place to start with. It will give you an idea of to whom you need to talk to collect the relevant data that will help your design.

For example, if you need to know what your software development department requires from the directory services, the best person to talk to will be the technical lead or architect of the team. They will be able to give you the exact answer you are looking for. If you ask the same question to the managing director, the answer may not be that accurate. So, before you seek the answers to your questions, you need to find the correct source.

When we gather business requirements, we need to consider the following:

- **Operation impact –** Who does this apply to? What benefits will the new changes bring? What are the risks involved?
- **Compliance and legal impact –** Will the new requirements have an impact on existing compliance? Will they have a legal impact?

- **Productivity impact –** What sort of productivity impact on the business may occur? Will it be a positive or negative impact?
- **Security impact –** What sort of impact will it have on security? Do we need to change the current security solutions that are in place?

To find relevant information, we can use questions similar to the following:

- Who are these changes applicable to? Is this going to be a company-wide change or is it only going to apply to certain user groups, business units, or departments?
- Who can explain the requirements in detail?
- What are the authentication and authorization requirements?
- What are the security requirements?
- What are the compliance and legal requirements?
- When does the solution need to be in place?
- What is the budget?

The preceding questions will help you to gather business information that is important for the Active Directory design. Apart from the above, you may have to add additional questions based on the industry, company size, compliance, and so on.

Defining security boundaries

The next step in the process is to define the security boundaries. If you purchase empty land to build a house, what will be the first thing you do? You need to clearly identify the plot's boundaries. Your building/development can't go beyond it. What kind of information do we need to gather in order to identify an identity infrastructure's boundaries? Understanding business operations is vital for this.

Rebeladmin Corp. owns a group of companies. The operations of each business are completely different from one to another. One is a hosting company and another one is an IT training institute. They also have a pharmaceutical company. The operations and business requirements are different for each of those companies. In such scenarios, multiple forests will be ideal as none of the companies depend on each other's resources for their operations.

Sometimes, even if it's a single company, some business units may need logical separation – at least from the directory service point of view. For example, Rebeladmin Corp. has a research and development department. Engineers in that department keep testing new software and services, and most of them are Active Directory-integrated.

Their security requirements rapidly change as well. They need to test different group policies. If it's the same Active Directory forest, the activities of these tests will impact the entire directory. Therefore, the best option will be to isolate their activity in a separate forest.

Identifying the physical computer network structure

Once we've identified the organizational structure and security boundaries, the next thing is to identify the physical computer network structure. It's important to identify how many branch networks there are, how they are connected together, and what kind of bandwidth is available between sites. This information helps us design the domain structure. Also, as part of this exercise, it's important to identify potential issues and bottlenecks between physically separated networks. In network diagrams, it may look nice to have links connected between sites, but if these connections have reliability and bandwidth issues, that's also going to impact your design. To overcome this, gather utilization reports and availability reports for three months and review them. It will give you good insights. **Read-only domain controllers** (**RODCs**) are used on branch networks when they cannot guarantee security and a reliable connection. Even the branch offices that are connected together and linked aren't reliable, and if the links have already been fully utilized, we need to fix that bottleneck first or place RODCs instead of a full-blown domain controller. Gathering evidence will help you make that call.

It is also important to gather information about the company road map and the company's products' road maps, as that will also impact the identity infrastructure design. For example, if a company is in the process of business acquisition or merging, your design should be future-proof to address that requirement. In the same way, if the company is going to downsize, that's also going to impact the design. So, it's best to discuss this with the relevant people and get a better understanding of future changes. As I mentioned previously, identity infrastructure changes are costly and involve a lot of work. By understanding the company's future, you will prevent this kind of awkward situation.

The company IT administration model is also important for identity infrastructure design. It can be either centralized or decentralized. Rebeladmin Corp. has a group of companies. Each of these companies has its own IT department. So, each company's IT teams are responsible for their own infrastructure. In this case, maintaining separate forests helps to divide the responsibilities for IT operations. Also, some companies may outsource their IT operations to a third-party company. This will change the security requirements in the identity infrastructure from in-house IT operations.

Some of the workloads may need to be isolated due to data protection and legal requirements. The design should accommodate these types of IT operation requirements.

Businesses are subject to specific government regulations. For example, banks and hedge funds need to follow specific rules in their operations to protect customer and trade data. Businesses that process credit cards need to be PCI-compliant and follow specific regulations. Also, if organization operations are aligned with ISO standards, it's another set of rules and best practices to follow. If an organization has branch offices in different countries, the government rules and regulations that are applied to those will be different from the rules that are applied to the headquarters. It is important to gather this data as it can also have an impact on the design.

Modern access management requirements are complicated. Some organizations have already extended their identity infrastructures to the cloud. Some organizations have been fully moved to Azure Active Directory-managed domains. Most application vendors have moved their products to the public cloud. Businesses need to collaborate with technology changes that are happening around them. Some products and services aren't going to continue anymore as in-house services, and customers will need to move to the cloud version. The identity infrastructure design should be future-proof as far as possible. Therefore, it's important to research and evaluate new technologies and services that will improve the organization's identity infrastructure and adopt them in the design as required.

In any project, the implementation phase is relatively easy. The design and planning process is complicated and time-consuming, but it is vital for successful output. Once you collect the data as described, go through it a few times and understand it properly. If you have doubts, go and gather more data to clear them. When Jonathan Ive designed the Apple Mac, do you think he designed it in one go? I am sure he must have used an eraser. But, in the end, everyone loved the Apple designs. No one cared about how hard it was or how much time he spent on it. The end result was the ultimate success. Therefore, don't be afraid to use an eraser in the design phase.

Designing the forest structure

The Active Directory design starts with designing the forest structure. The Active Directory forest is the security boundary for the identity infrastructure. When you deploy the first domain controller in your infrastructure, it creates a forest as well. Every Active Directory infrastructure has at least one forest.

There are two types of forest implementations:

- Single forest
- Multiple forests

The type of the forest is decided based on many things, such as the size of the company, legal requirements, operation requirements, mergers and acquisitions, resource isolation, and so on.

Single forest

A single forest deployment is the default deployment mode. Most business models fit into the single forest model. The complexity and cost of implementation are low in this model. One of the main things you need to consider in this mode is replication. Domains are used to partition the directory and manage the replication. But forest-wide data, such as schemas, still needs to be replicated across all domains. If replication involves branch offices, you need to make sure forest-wide replications are handled appropriately. We can always start with the single forest model first and then go to the multiple forest model later if required.

Multiple forests

The multiple forest model is a complex implementation process. The cost of implementation is also higher as it requires additional resources (hardware, software, and maintenance). There are several reasons why you might need the multiple forest model:

- **Business operations isolation**: Businesses can have groups of companies or departments that are required to operate independently. Their dependence on other departments and partner companies may be minimal. In such scenarios, it is good to create a separate forest for them.
- **Rapid changes in directory services**: Businesses may have some departments or business units that involve rapid directory changes. For example, R&D, DevOps test environments, and software development departments may require AD schema changes, Active Directory integrations, and Group Policy changes to test or develop products and services. It is best to keep them in a separate forest in order to minimize the impact on the entire identity infrastructure.
- **IT operation mode**: Some organizations have decentralized IT operations. Groups of companies are an example of this. Each company may have its own IT staff and be required to operate independently. A separate forest will define the security and operation boundaries for each of those companies.
- **Resource isolation**: This is an ideal solution for service providers or organizations with multiple separate forests that like to share resources. For example, Rebeladmin Corp. has a group of companies with separate AD forests.

- Each company has its own IT department. But the mother company still likes to share some common systems among all the companies, such as payroll, email, and CMS. Creating a separate forest for these resources will allow the organization to manage them in an efficient, secure way. Other forests can have the forest trust the resource forest and use the services hosted there. Also, service providers can create resource forests to isolate their products and services.
- **Legal requirements**: Businesses are bound to government rules and regulations. They may also have business agreements with partners and merged companies. Based on that, they may be required to create a separate forest to isolate data, services, and identities.
- **Business acquisitions or divestiture**: Business acquisitions or divestiture will require extended security boundaries or isolated resources and identities. The best way to do that will be to use the multiple forest model. If there is a requirement to share data or resources between forests, a cross-forest trust can be established.

Creating the forest structure

Once the forest mode has been decided on, the next step is to create the forest structure. In order to do that, we need to decide whether we are going to achieve autonomy or isolation.

Autonomy

Autonomy gives you independent control over resources. An Active Directory environment that is focused on autonomy will help administrators manage the resources independently, but there will be more privileged administrators who can manage the resources and privileges of other administrators.

There are two types of autonomy:

- **Service autonomy**: This will provide privileges to an individual or a group of administrators to control the service level of AD DS fully or partially. For example, it will allow administrators to add or remove domain controllers, modify the Active Directory schema, and modify DNS without the forest owner.
- **Data autonomy**: This will provide privileges to an individual or a group of administrators to control data stored in Active Directory or domain-joined computers. This also allows you to perform any administrative tasks regarding data without approval from a privileged user. This autonomy will not prevent forest service administrators from accessing the data.

In most scenarios, organizations opt for isolation rather than autonomy. Let's go ahead and see what isolation can deliver.

Isolation

Isolation gives independent and privileged control over resources. Administrators can control resources independently, and no other accounts can take control.

There are two types of isolation:

- **Service isolation**: This will prevent any other control or interference with AD DS, other than the administrators defined in it. In other words, it will provide full control over the identity infrastructure. Service isolation happens mainly due to operations or legal requirements. As an example, Rebeladmin Corp. has three different services that are built in-house. Each service has its own customer base. Operations in one product should not impact others. Service isolation will allow the organization to isolate the operation for each service.
- **Data isolation**: This will provide ownership of the data that is stored in Active Directory or domain-joined computers to individuals or groups of administrators. However, data administrators cannot prevent the service administrator from accessing the resource they control. In order to isolate a subset of data completely, they will need to create a separate forest.

The number of forests that are needed for an infrastructure depends on the autonomy or isolation requirements.

When deciding on autonomy or isolation, we can stick to the following principles:

- Isolation provides exclusive control over data or services.
- Isolation is more secure and easy to apply as standard as its operation boundary is limited.
- If the operation boundary is limited, the more control you will have (security, IT support, compliance, and so on).
- Autonomy allows organizations to manage their own services or data based on administrative decisions.
- Autonomy provides more flexible data and service management. Privileges must be handled carefully using a delegated administration method.
- Achieving autonomy is less complex and generally more cost-effective than isolation.

In some scenarios, a business may need both autonomy and isolation. This mainly happens due to legal and compliance requirements. We can have both autonomy and isolation in a design but it requires multiple forests.

Selecting forest design models

Once the forest model and the number of forests have been decided, the next step is to select forest design models. There are three forest design models: organizational, resource, and restricted.

The organizational forest model

In an organizational forest model, resources, data, and identities will stay in separate forests and will be managed independently. This model can be used to provide service autonomy, service isolation, or data isolation:

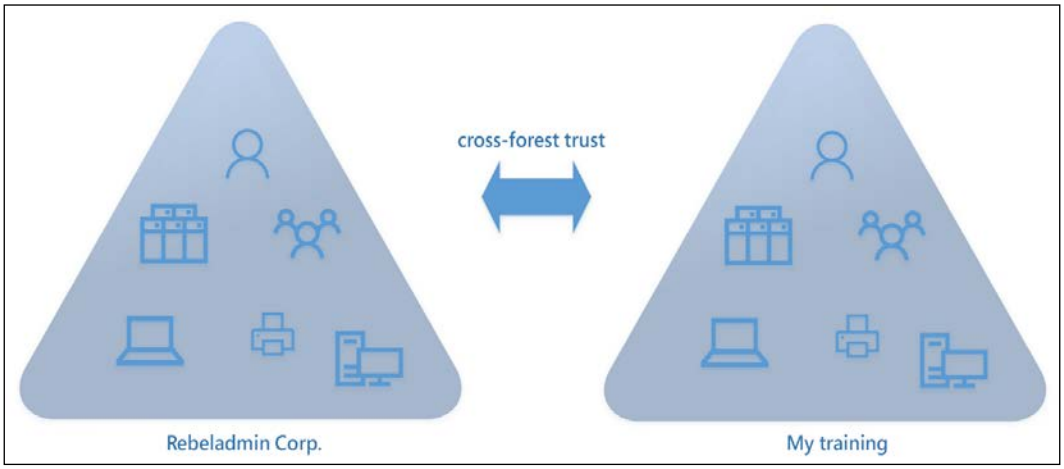


Figure 3.1: Organizational forest model example

In the preceding example, **Rebeladmin Corp.** and **My training** are two companies under the same mother company. Due to the operation requirements, it needs service isolation. In order to do that, engineers have created two separate forests. Each company has its own IT department and manages resources and identities independently. If resources need to be shared between two forests, that can be done via a **cross-forest trust**.

The resource forest model

In the resource forest model, a separate forest is used for resources. A resource forest doesn't contain any user accounts; instead, it contains service accounts and resource forest administration accounts. All of the identities for the organization will be in a separate forest. The cross-forest trust that's created between forests and users for an organization forest can access resources in the resource forest without additional authentication.

Resource forests are really useful when you work with multiple organizational forests located in multiple countries. This will help to isolate services based on legal, compliance, or performance requirements. As an example, Rebeladmin Corp. has branches in the USA, Europe, and Asia. Each region has multiple branches and some of the branches are located in different countries. Let's evaluate some of the possible designs we could come up with and the pros and cons associated with those.

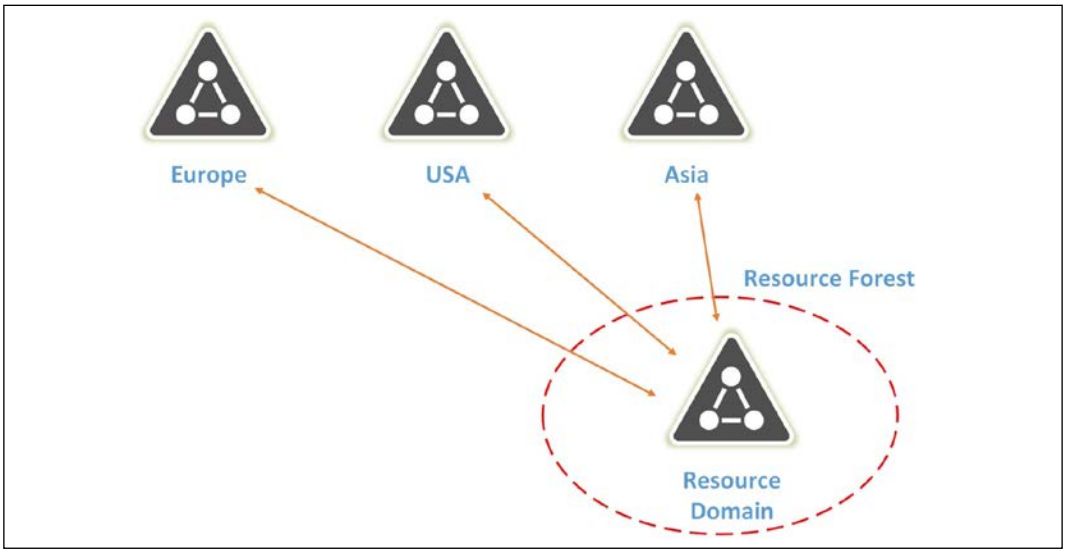


Figure 3.2: Resource forest model example 1

In the above design, each region has its own organizational forest. Each of these domains is managed separately. If each region has multiple branches, those will be represented in each directory as a separate "Organizational Unit." The company has certain resources that need to be shared across the business. These resources will be placed in separate resource forests and each regional forest will have two-way trust with the resource forests.

Pros	**Cons**
Individual sites still have the highest level of control over their own data and access.	As the number of sites increases, the amount of delegation increases, meaning more management.
More control over compliance.	Challenges in maintaining security standards due to the large administrative boundary (for regional domains).
Less dependency on connections between sites.	

Easy to implement SSO by using resource forests.	
Easy to apply global security standards.	
Individual sites still have data and service autonomy.	
Easy to maintain common services and common data via resource forests.	
Less impact on common services or data due to site-level infrastructure changes.	
More control over permission delegation.	

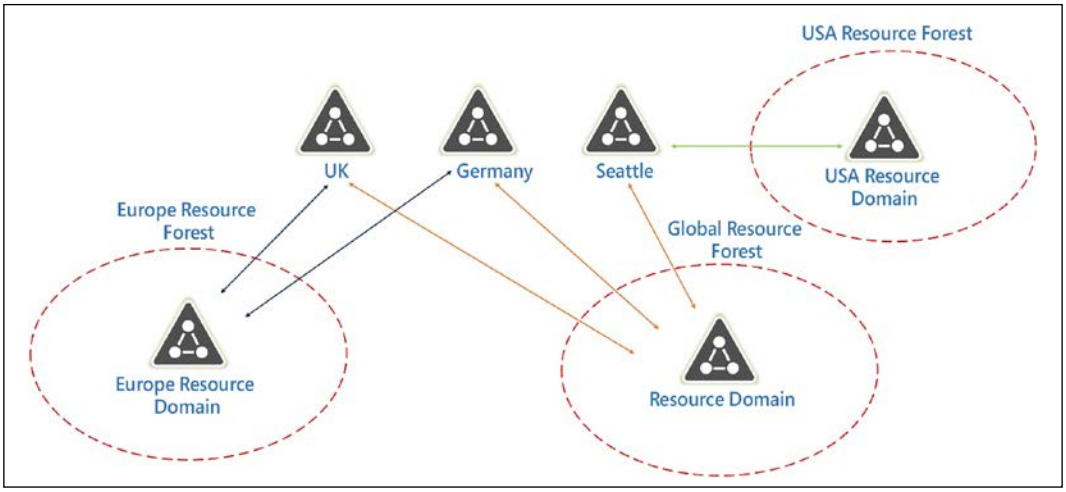


Figure 3.3: Resource forest model example 2

In the above design, I have mainly considered two things. One is maintaining compliance requirements, which apply at a regional level. The other one is improving performance by maintaining regional resource forests. Sometimes it is a legal requirement to store data or use services at least in the same region. In the above example, the UK and Germany domains are connecting to a resource forest in the EU region. So, the owner of the forest can place applications and services that are relevant to that region in this resource forest. Also, this way, the European domains do not have to depend on cross-region connectivity to access the resource forest. In the above example, I also mentioned one global resource forest. This will be used to share applications and services across the business. Also, in this design, each country or branch can have its own organizational forest. This will give more flexibility for management.

Pros	Cons
More options for data and service placement.	More domains, more management, more complexity.
More control over compliance.	Dependencies on domain trusts and connections between domains and forests.
Less dependency on connections between regions.	More complicated and careful planning is required when bringing new organizations onboard (M&A).
More control over security standards (can apply standards to different tiers).	
Less service impact with regional outages.	

The restricted access forest model

In the restricted access forest model, a separate forest is created to isolate identities, and data must be separated from the other organization's data and identities. No trust is created between the two forests, so identities in one forest will not be able to access the resources in another. To access the resources in each forest, we need to have separate user accounts. The **restricted access forest** model provides data isolation:

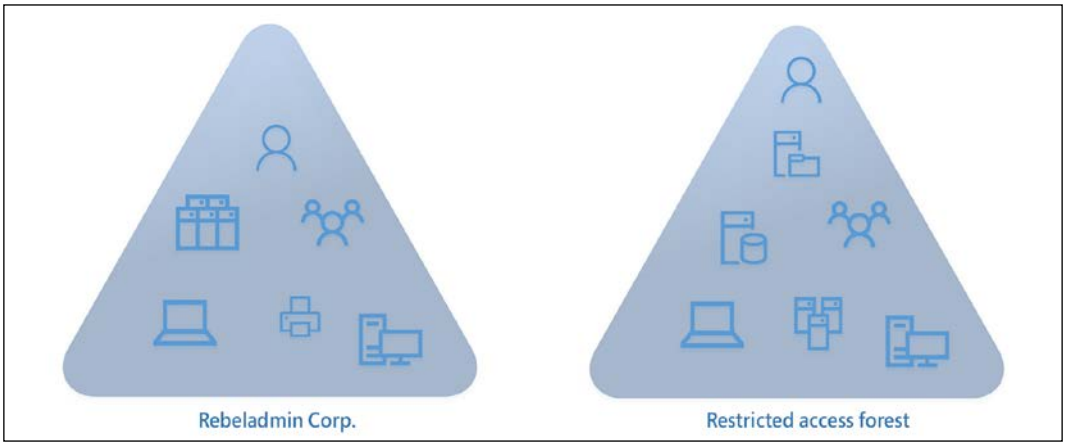


Figure 3.4: Restricted access forest model example

In the preceding example, **Rebeladmin Corp.** is involved in a corporate divestiture process. For some of the assets, data and identities need to be isolated completely within the business. In order to do that, the company introduced the restricted access forest. This model is usually used due to legal/compliance requirements, security requirements, or IT operation requirements.

Once the forest structure, the number of forests, and the design model have been finalized, the next step will be to design the domain structure.

Designing the domain structure

Every AD DS forest has at least one domain. When you set up your first domain forest, it will also become the default domain. There are a few reasons why you will need to consider having multiple domains in a forest:

- **Smaller administrative boundaries**: Active Directory is capable of managing nearly 2 billion objects. Having a large directory creates administrative nightmares. Imagine managing a large herd of sheep. As the herd grows, shepherds need to put in more and more effort to manage it. Predators will also take advantage of it, and, sometimes, shepherds may not notice missing sheep as they are too busy managing the herd. Instead of managing a large number of sheep together, isn't it easier if each shepherd manages smaller herds? Domains will help set smaller administrative boundaries and smaller management targets. This will help manage organization resources efficiently.
- **Replication**: Every domain in the Active Directory forest shares the same schema. It needs to be replicated to all the domain controllers. However, each domain has its own domain partition, which will only need to be replicated to the domain controllers inside the domain. This allows you to control the replication within the Active Directory forest. Rebeladmin Corp. has branches in different countries. These branches are connected together with leased lines. Each of these branches also has domain controllers set up. So, if it's a single-forest, single-domain setup, each and every domain controller will need to be replicated. If we create different domains to represent each branch office, it will eliminate unnecessary replication as the domain partition only needs to be replicated within domain boundaries.
- **Security**: In the previous section, we talked about data and service isolation based on forests. These are due to operational and legal requirements in the business. Domains help isolate resources and objects based on the security requirements within the forest. My-Learning Inc. is an IT training company. It has mainly two types of students. Some are academic students who are studying an HND program, and others are students who are taking professional exams. Both groups have separate labs, software, and resource access. Both groups have their own data, resources, and identity security requirements. Some of these requirements are only achievable via domain-wide security settings. Therefore, having two separate domains will allow them to apply different security standards without interaction.

There are a few models we can use to design the domain structure.

Single domain

A single domain model has a single-domain, single-forest structure. It is easier to administer and has a lower cost to implement. When we set up the Active Directory infrastructure for the first time, it will be based on this single domain model by default. The domain will become the root domain for the forest by default, and all the objects will be stored in there.

In this mode, all the directory data will need to be replicated to all the available domain controllers. It doesn't matter if it's in a different geographical location. The user can use any available domain controller to authenticate into any system or resources. All domain controllers in the domain can act as global catalog servers. The downside of this model is the administrative overhead and less controlled replication traffic. We can always start with a single domain model and move to other models later as required.

Regional domain

In the regional model, the AD DS forest will contain the forest root domain and the multiple domains that are connected via **wide area networks** (**WANs**). This is mainly applicable to branch offices and sub-companies located in different geographical locations:

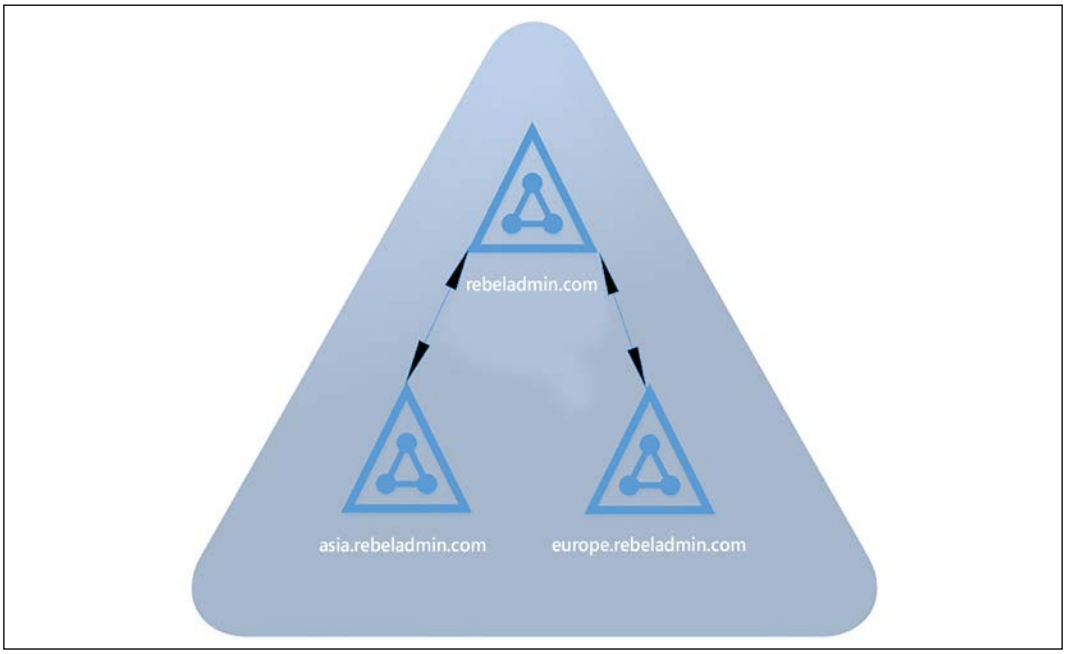


Figure 3.5: Regional domain model example

This model is complex to implement compared to the single domain model and will require additional hardware and resources. In this model, the domain partition data will only be replicated to the domain controllers inside the domain, and it will allow you to reduce the replication traffic flow between WAN links. You can find more information about AD replication in *Chapter 11*, *Active Directory Services - Part 01*. The regional domain model will also help you isolate the security requirements. Note that this will not provide data or service isolation. If it's a requirement, it needs to be done at the forest level.

The branch/site domain

If the company has multiple branches with completely independent IT operations, it is best to create a domain for each branch or site. Each of these domains can also have its own Active Directory forest. When your security boundary gets bigger, it also increases the risk. If we maintain one large Active Directory forest with inter-connected domains, we need to think how if one domain gets compromised, it will affect the other domains in the forest. If there is an attack, we need to think about how we can stop lateral movement. The investment needed to maintain a large directory is higher than maintaining disconnected directories. These branch domains can still connect with each other using domain or forest trusts when required.

The number of domains

After deciding on the domain structure model, the next step is to identify the number of domains that are required. The number of domains depends on the number of objects that will be managed, the number of geographical locations, administrative requirements, and the bandwidth of links.

In the following table, I have listed the number of users that can be maintained in a domain against the replication bandwidth:

Bandwidth	**If 1% of bandwidth is allowed for replication**	**If 10% of bandwidth is allowed for replication**
128 Kbps	25,000	100,000
256 Kbps	50,000	100,000
512 Kbps	80,000	100,000
1.5 Mbps	100,000	100,000

Bandwidth between locations is less of a problem these days. But administrative requirements and security requirements can be more relevant reasons to consider.

When the number of domains increases, the complexity, cost of implementation and maintenance, and management of diverse security settings increase as well. Therefore, decide on the number of domains in a meaningful manner. I have seen people create domains in places where they could achieve the same thing using an OU or some different group policies.

Deciding on domain names

Every domain in the forest needs to have a unique name. There are mainly two types of names: NetBIOS names are used for Windows (SMB)-based sharing and messaging, whereas DNS names can be resolved by the internet and different systems. When you're promoting the domain, it asks for the DNS name, as well as the NetBIOS name.

When considering the naming, keep the following points in mind:

- Don't use names that you do not legally own; if it's a fully qualified domain name, make sure that you have authority over it with the internet registrar.
- Don't use numbers and special characters that could easily confuse users.
- Don't use names associated with products and operating systems, for example, NT and Windows.
- Avoid using longer domain names or suffixes.
- Be careful with the spelling. I have seen people who deployed the entire domain structure and then realized they made a mistake with the spelling.

As a best practice, if the organization owns a business domain name, try to use it as the Active Directory domain as well. This simplifies Azure AD integration too. When planning names for the child domains, you can use a country name, continent, or branch name. As an example, Rebeladmin Inc. has a branch in the UK, so the child domain for that could be `UK.rebeladmin.com` or `Europe.rebeladmin.com`. The domain name should be clear enough to explain its purpose. As an example, if the subdomains are named `branch1.rebeladmin.com`, `branch2.rebeladmin.com`, and so on, it will not make sense at a glance to anyone.

Once the domain name has been assigned, it can't be changed without redeployment, completing a domain rename process, or by performing domain migration. But Active Directory can have multiple UPN suffixes.

The forest root domain

The first domain that's set up in the forest becomes the forest root domain. This root domain contains two important privileged groups, which are Enterprise Admins and Schema Admins.

Members of these security groups can add/remove domains and modify the Active Directory schema.

In the multiple domain model, there are two types that you can use to define the forest root domain:

- **Dedicated forest root domain**: A separate domain to operate as the forest root domain. It will not contain any regular user accounts, objects, or resources. It will only contain the service administrator accounts. All the other domains in the forest will be child domains for this root domain. In a single domain environment, domain administrators can add themselves to the Enterprise Admin or the Schema Admin group. But when you have a separate root domain, child domain administrators will not be able to add them to these privileged groups without doing that from the forest root domain level. The dedicated forest root domain shouldn't share a geographical naming convention, and it should stand with a separate name from the rest of the child domains. For example, `rebeladmin.com` could be a root domain name instead of `Europe.rebeladmin.com`.

If you are using a top-level domain (ex-rebeladmin.com) as your AD name, you may face issues if the company is also running a public website under the same name. In such a scenario, you need to adjust the internal DNS records; that is, if users want to browse the website from the local network.

- **Regional forest root domain**: If you're not going to use a separate forest domain, the regional domain can also be selected as the forest root domain. It will be the parent domain for all other regional domain controllers. For example, `HQ.rebeladmin.com` could be the regional root domain. This domain can contain regular user accounts, groups, and resources.

Now, we have all the required information to design the domain structure. The next step will be to determine the domain and forest functional levels.

Deciding on the domain and forest functional levels

Once the domain and forest designs are ready, the next step is to decide on the forest and domain functional levels. The forest and domain functional levels define the AD DS features that can be used in the identity infrastructure. You cannot have AD DS 2022 features if your organization level is running on the Windows Server 2012 domain and forest functional levels. When you add the domain controller to the existing forest or domain, it will automatically match the existing forest and domain functional level.

There are a couple of things you need to consider when you're deciding on the forest and domain functional levels:

- **Existing domain controllers**: It is always good to run the latest and greatest functional levels, but it isn't always practical. The lowest domain controller version in the domain decides the maximum forest and domain function you can have without an upgrade. As an example, in your domain, if you are running a Windows Server 2008 domain controller, the maximum forest and domain functional level you can have is Windows Server 2008. It will not prevent you from adding a domain controller with Windows Server 2022; however, until you decommission the Windows Server 2008 domain controller, it isn't possible to upgrade the forest and domain functional levels further.
- **Application requirements**: Sometimes, legacy applications support only certain domain and forest functional levels. This happened to me on several occasions when I was planning domain upgrades for customers. Most of the time, it happens when companies have custom-made applications. Therefore, check with your application vendors in terms of whether they're going to be compatible and supported before deciding on the forest and domain functional levels. This is very rare but still can happen with custom-made applications and legacy applications that are no longer supported.

Once you have defined the forest and domain functional levels, the lower domain controller version cannot be introduced to the system. If you are running the Windows Server 2016 domain and forest functional levels, you cannot introduce the Windows Server 2012 R2 domain controller to the same forest.

Before AD DS 2012 R2, if the domain and forest functional levels were raised, they couldn't be downgraded again. After AD DS 2012 R2, you can downgrade forest and domain functional levels if required.

In order to find the current domain and forest functional levels, you can run the following commands from any domain controller:

- To find the domain functional level, run the following command:

```
Get-ADDomain | fl Name,DomainMode
```

- To find the forest functional level, run the following command:

```
Get-ADForest | fl Name,ForestMode
```

Designing the OU structure

In Active Directory, there are different types of objects, such as user accounts, groups, and devices. It is important to manage them effectively. OUs can group objects that have similar administrative and security requirements within the domain. Organizational units are also used to delegate the administration of objects and apply group policies.

OU design changes are less complex compared to domain and forest level structure changes. When you move objects from one OU to another, they will inherit the security settings and group policies that are applied to the destination OU. Moving an object will not move any settings it has at the source OU level.

The domain administrators can delegate permission to users to become OU administrators. OU administrators can manage objects and manage policies within the OU. They can also create child OUs and delegate permissions to another user/ users to manage child OU objects. OU administrators will not have control over the directory services operations, and it is another way of managing privileged access within the directory. This is quite similar to the way NTFS permissions work.

There are two types of organization units:

- **Account OU**: An account OU contains the user, group, and computer objects. It is the forest owner's responsibility to create the OU structure and delegate the permissions.
- **Resource OU**: A resource OU contains the resources and user accounts that are used to manage resources. The forest owner must create the OU structure and delegate permissions as required.

It's important to follow up on some standards when you're defining the OU tree. These standards can be specific to the organization's requirements. In the following list, I have mentioned a few methods that can be used to organize an OU tree:

- **Based on the organization structure**: The OU tree can match the same organization structure; this will be easy to follow in most cases, but it will make the boundaries larger. In this method, OUs will be created based on departments and job roles.
- **Based on geographical locations**: This method can be used to build the OU structure if the organization has branch offices. It can be further broken down using departments or teams that exist in each branch office.

- **Based on departments**: This is the most commonly used method for creating the OU structure. It can be further structured based on geographical location. For example, the sales OU can have child OUs to represent branch offices such as Dallas, London, and Toronto.
- **Based on security requirements**: Group policies can be used to apply security policies and settings to objects in the OU. For example, tier-1 support engineers in the IT team will have a different set of security policies from tier-3 engineers. To accommodate that, we can create a tier-1 OU and a tier-3 OU. This method is suitable for small businesses.
- **Based on the resource type**: The OU tree can be structured based on server roles, applications, and device types.

It's possible to use a mixture of all of these methods as well. There is no limit to the number of child OUs you can create, but for administration and manageability, Microsoft recommends that you don't have more than 10 levels.

Once you have finalized the OU structure, make sure that you document it properly. Also, provide guidelines so that engineers can follow the method you used to structure it. When I've worked on projects, I've noticed that some OU structures don't make any sense at all as, over time, different engineers used their own methods to structure the OUs. Organizations should have a specific standard to follow.

Designing the physical topology of Active Directory

Dos	Don'ts
Run domain controllers in different virtualized clusters in different data centers in order to avoid a single point of failure.	Don't save the Active Directory database and log files on virtual IDE disks. For durability, save them on a VHD that's attached to a virtual SCSI controller.
Virtual hard disks (**VHDs**) security is important as copied VHDs can map to a computer and read the data inside it. If someone unauthorized gains access to `ntds.dit`, it will expose the identities. We can use encryption services to do server-side disk encryption with customer-managed keys.	
Disable time synchronization between the virtual domain controller and the host. This will allow domain controllers to sync time with PDC.	Don't use a copy of the already deployed domain controller's VHD to create additional domain controllers.

N/A	Don't use the Hyper-V export feature to export the virtual domain controller (for rollback or restore purposes).
N/A	Don't pause or stop domain controllers for long periods of time (longer than the tombstone's lifetime).
N/A	Don't take snapshots of virtual domain controllers.
N/A	Don't use a differencing disk VHD as it decreases performance.
N/A	Don't copy or clone Active Directory VHDs.

In the previous sections of this chapter, I explained how we can design the Active Directory logical topology. The next step is to design the physical topology of the Active Directory design.

Physical or virtual domain controllers

Most of the workloads we have in modern infrastructures are virtualized. Domain controllers can also be virtualized; however, depending on the virtualization vendor, the best practices to follow will be different. Therefore, if you plan to deploy virtual domain controllers, refer to your software vendor and find out what the recommendations for virtual domain controllers are. The guidelines in this section will focus on the Microsoft Hyper-V virtualization platform.

It isn't recommended that you use only virtual domain controllers. In fact, it is recommended that you balance between physical and virtual domain controllers for availability and integrity. This will mitigate the risk of losing all domain controllers due to a virtualization platform malfunction.

In a virtualized environment, make sure that you distribute the domain controllers among hosts to avoid a single point of failure.

In the following table, I have listed the dos and don'ts in terms of the virtualized domain controllers:

Dos	**Don'ts**
Run domain controllers in different virtualized clusters in different data centers in order to avoid a single point of failure.	Don't save the Active Directory database and log files on virtual IDE disks. For durability, save them on a VHD that's attached to a virtual SCSI controller.

Virtual hard disks (VHDs) security is important as copied VHDs can map to a computer and read the data inside it. If someone unauthorized gains access to ntds.dit, it will expose the identities. We can use encryption services to do server-side disk encryption with customer-managed keys.	
Disable time synchronization between the virtual domain controller and the host. This will allow domain controllers to sync time with PDC.	Don't use a copy of the already deployed domain controller's VHD to create additional domain controllers.
N/A	Don't use the Hyper-V export feature to export the virtual domain controller (for rollback or restore purposes).
N/A	Don't pause or stop domain controllers for long periods of time (longer than the tombstone's lifetime).
N/A	Don't take snapshots of virtual domain controllers.
N/A	Don't use a differencing disk VHD as it decreases performance.
N/A	Don't copy or clone Active Directory VHDs.

Domain controller placement

Domain controller placement in the infrastructure is dependent on a few things:

- **Network topology**: Organizations can have different buildings, branch offices, and data centers connected together. The services and resources that are hosted in those locations may require domain controller integration.
- Replication is key for domain controllers. The placement of the domain controllers in the network will depend on whether it's possible to achieve successful replication or not. Network segmentation can prevent relevant traffic from passing through networks, which can impact replication. It is important to adjust the network topology to support the Active Directory design you have in place.
- **Security**: Physical security is important for domain controllers as its holds the identity infrastructure footprint. In places where you cannot guarantee physical security in your network, it is recommended that you don't place the domain controller. In such scenarios, instead of the domain controller, it is possible to deploy an RODC.

- **Link reliability between sites**: As I mentioned previously, replication is key for the health domain controller infrastructure. If the connectivity between sites isn't stable, it isn't possible to place the domain controller and maintain healthy replication. In such scenarios, it's advisable that you use an RODC.
- **Active Directory sites**: We covered Active Directory sites in *Chapter 1, Active Directory Fundamentals*. This is important in terms of physical topology design. In later chapters, I will demonstrate how to set up site links and how to manage them.

Global catalog server placement

Global catalog servers are responsible for keeping a fully writable copy of objects in their own domain and a partial copy of all other domain objects in the forest. It facilitates querying objects in the entire forest. The global catalog service is part of the domain controller services, and it cannot be separated.

In a single-forest, single-domain environment, all the domain controllers can be global catalog servers as it won't be different from domain replication. But in a multi-domain environment, global catalog server placement involves planning as it increases the amount of data to be replicated, as well as the bandwidth.

There are certain things that you need to consider when you're placing a global catalog server:

- **The number of users**: It is recommended that you place a global catalog server on any site that has over 100 users. This will help maintain site availability in the event of WAN link failure.
- **WAN link reliability**: If you are struggling with link availability between sites, it's recommended that you place global catalog servers on a remote site, and enable universal membership caching.
- **Roaming users**: Roaming users are required to connect to the global catalog server when they log in for the first time from any location in the infrastructure. Therefore, if users are using roaming profiles on remote sites, it's important that you place the global catalog server.
- **Application requirements**: Applications such as Microsoft Exchange heavily depend on global catalog servers. If similar applications are hosted on remote sites, they will be required to have a global catalog server.

When the universal membership caching feature is enabled in the domain, any domain controller can process any login request locally without going through a global catalog server. This will provide a faster logon experience and reduced traffic when users use it over WAN. More information about enabling the UGMC feature is available in the following blog post: `https://bit.ly/3bOLMx6`.

Once the global catalog server placement has been determined, the Active Directory design phase is complete. Now that we have the logical and physical design for the Active Directory infrastructure ready, let's have a look at designing a hybrid identity.

Designing a hybrid identity

There are many reasons why organizations look to *extend* their On-prem AD to Azure AD. Let's look into some of those reasons:

- **Cloud application (SaaS) adoption**: Organizations use different types of applications for their operations (On-prem). Most standalone applications are easy to manage and maintain, but some applications have complexities. Some applications require lots of resources. SAP applications are a great example. SAP applications depend on a few components such as database servers, application servers, and front-end servers. If any of these components fail, the whole application fails too. Therefore, you need to plan for high availability on top of that. Now, more and more vendors are taking away this burden from customers and offer cloud versions of applications instead of On-prem ones. By doing this, organizations don't have to worry about scalability, availability, and maintenance. The application will be available for users whenever it's required. If an organization is moving to cloud applications, it also requires some sort of identity and access management. The majority of these SaaS solutions support Azure AD integration. By extending On-prem AD to Azure AD, organizations can maintain the unified login experience.
- **Authentication requirements**: On-prem AD uses Kerberos and NTLM for authentication. However, modern authentication requirements are more complex than that – especially when you consider web-based services.

  Unlike Windows AD, Azure AD is built for the cloud, and so it supports advanced authentication protocols such as SAML 2.0, OAuth 2.0, OpenID Connect, and WS-Federation. Using hybrid identity, we can have both types of authentications under one identity infrastructure.
- **Advanced identity protection**: Identities play a vital role when it comes to data security. Modern-day adversaries target identities as it opens up access to valuable data.

Microsoft is well aware of this and is continuously investing in improving identity and data protection. Conditional Access, Azure Information Protection, Azure RMS, Azure Identity Protection, and Azure Privileged Identity Management are some of the Azure services that can help with identity protection. These new features/services only work in cloud-only or hybrid infrastructures.

- **Moving workloads to Azure IaaS and Azure PaaS**: If an organization has already made decisions to move workloads to the cloud, there is no point in deploying additional domain controllers in Azure in order to extend the On-prem AD. It will only add additional dependencies and management overhead. Instead, the organization can choose a hybrid identity.
- **Unified access experience**: Some organizations use cloud applications (SaaS) as well as On-prem (web) applications for their operations. These systems may have different types of authentications and different types of portals to do authentication. But using Azure AD, Azure AD Application Proxy, and **single sign-on** (**SSO**), they can have a unified access experience across all web applications. If the organization is willing to use existing identities for authentication, we need a hybrid identity.

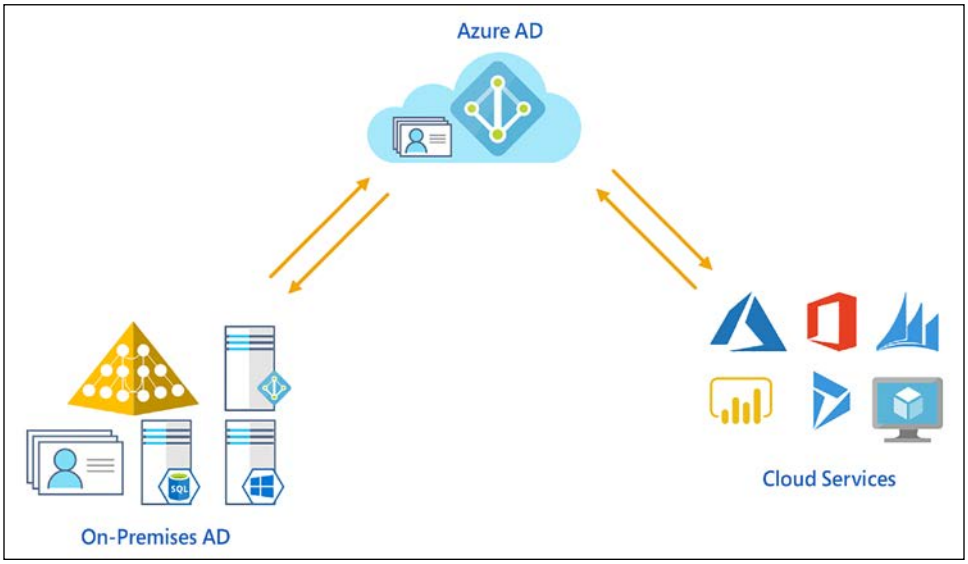


Figure 3.6: Unified access experience

Rebeladmin Corp. runs its applications and services from a data center located in their head office. It uses Microsoft Active Directory to manage its corporate identities and access. Recently, the company replaced some of its On-prem applications and services with cloud versions. Now, these applications and services also require some sort of identity and access management.

Due to this, they have decided to extend their on-prem AD to Azure Active Directory. The final solution includes the following:

- On-prem identities and password hashes are synced to Azure AD using Azure AD Connect. This allows users to authenticate with on-prem and cloud applications/services using the same username and passwords.
- Rebeladmin Corp. is using password writeback and the self-service password reset feature of Azure AD, which allows users to reset their password without the help of an IT helpdesk.
- Azure AD's seamless SSO feature allows users to authenticate into cloud and on-prem applications automatically via corporate devices.
- In Azure AD, objects are stored in a flat structure. Therefore, object management is still done using on-prem AD. Here, engineers can structure AD objects in a hierarchical order. They are still using group policies to manage user and computer settings in the on-prem infrastructure.
- All Windows 10 devices are directly joined to Azure AD, and down-level devices are registered with Azure AD using the hybrid Azure AD join method. The main difference between these two methods is Azure AD joined device authentication is only done through Azure AD and a hybrid Azure AD join device can authenticate through Azure AD or on-prem AD. Rebeladmin Corp. manages the state of this device using Microsoft Endpoint Manager.
- Rebeladmin Corp. published a few on-prem web-based applications to the internet using Azure AD's Application Proxy. In this way, users will not feel a difference when they authenticate into cloud or on-prem services.
- Azure MFA is also in place to provide an additional layer of security during the authentication process.
- Sign-in logs, risk events, and log analytics running in Azure help organizations identify identity infrastructure threats in advance.
- Rebeladmin Corp. is using Azure Identity Protection, Azure Information Protection, and Conditional Access in order to implement advanced identity and data protection for both cloud and on-prem resources.

The preceding scenario is a perfect example of a corporate hybrid identity setup. The solution that is in place not only extends the identity infrastructure, it also uses Azure AD-related services and features to improve identity and access management, identity protection, and data security across the board. I will explain these services later on in the book. However, every organization has a different budget, different business requirements, and different skills, all of which will affect the hybrid identity design.

Let's look into a few areas you need to consider when you're designing your hybrid identity.

Cloud approach

Businesses are moving to the cloud for many reasons. Based on intentions, we can group these businesses into four categories:

- **Cloud only**: This typically applies to new businesses. They aren't willing to invest in on-prem solutions at all. They are running their operations completely on the cloud. This model is ideal for businesses that are concerned about availability and scalability. Hybrid identity is irrelevant for these types of businesses as there is no on-prem footprint.
- **Future cloud only**: There are organizations that already have an on-prem infrastructure, but are willing to move completely to the cloud in the coming years. They have already decided there will be no more on-prem infrastructure refreshments and that all new investments are already being made in the cloud. For them, hybrid identity is an intermediate solution. on-prem compatibility is no longer a concern for them when they want to implement a new service or feature.
- **Permanent hybrid**: Due to legal requirements, compliance requirements, or business requirements, some organizations are not going to be cloud only but they are comfortable with using cloud services, which helps them improve their current operations. Most of these solutions will be hybrid compatible. They are going to continue investing in the improvement of the on-prem infrastructure.
- **Forced to the cloud**: On rare occasions, some organizations have to move to cloud services as they have no other option. This could be due to a vendor who no longer provides on-prem solutions and has moved completely to a SaaS solution. Alternatively, it could be due to a partner who moved their services to the cloud. For them, hybrid identity is just to get authentication to an app or a service using the same on-prem identities. There will be no additional investments to improve the identity infrastructure using Azure AD or associated services.

So, which category does your organization fall into? We must understand the organization's cloud approach correctly before we propose a design. Most of the time, designs get rejected not because of technical reasons but due to a mismatch of interest or budget.

Identifying business needs

If I go to the Auto Trader website and just search for a car, I'll receive more than 400,000 results. It would be impossible to go through all of those to find the car I have in mind. But if I, say, want a BMW 4 series, automatic, 2020 build, it is very easy to find the one I need. As long as we express our requirements correctly and as long as the supplier captures them correctly, we will get the solution we are looking for.

Business needs in organizations come in different forms. They can be in the form of operation improvements, strategy changes, or compliance/security requirements. In order to provide the appropriate solution, we need to know the ins and outs of the requirement. This is the most challenging phase of the design process for engineers. This always starts with some sort of communication between engineers and the business owner, director, manager, or department head. Some of them may not be technical enough to explain what exactly they need from a hybrid identity. Therefore, it is best to use generic questions to gather as much information as possible.

To design a hybrid identity, we need to gather information from the following areas:

- Cloud services (the ones we are going to use)
- Current on-prem infrastructure
- Authentication requirements
- Security requirements
- Monitoring/reporting/alerting requirements

In the following subsections, I have listed a set of generic questions and what we can provide based on the outcome:

- **Cloud services**:
    - Why is the organization looking to move to the cloud? (The answer will help determine the organization's cloud approach.)
    - What cloud services is the organization going to use? For example, SaaS, PaaS, or IaaS. (Based on the solution, the Azure AD implementation will change as well. As an example, if the organization is going to use PaaS, the relevant virtual networks and DNS should be associated with the Azure AD setup.)
    - How many users are going to use these services? (The answer will help determine the Azure AD license requirements.)
    - Who is going to manage these services? (The answer to this will help determine whether staff require additional training, professional support, and so on.)

    - How critical is this service going to be? (This helps us decide on the support options for the solution.)

- **Current on-prem infrastructure**:
    - Explain your current identity infrastructure (this is very important as this will help you understand the physical and logical design of the current identity infrastructure).
    - Are you experiencing any issues with user authentication? (Based on the answer, we can decide whether we need to perform on-prem AD health checks before hybrid identity implementation.)
    - What AD services are currently in use? For example, ADFS, RMS, and ADCS. (The answer will help you decide what Azure AD services could replace/improve the operations of these components. As an example, ADFS can be replaced with Azure Pass-through Authentication and seamless SSO services. AD RMS could potentially be replaced by Azure RMS.)
    - Does the organization have web services published externally? Are you happy to have the same login experience across the board? (If the answer is yes to both, we can use Azure AD Application Proxy to provide the same login experience for both cloud and on-prem web services.)
    - Does the on-prem device state need to be managed by Azure AD? (This will help determine what Azure AD join method to use and decide whether they require an MDM solution such as Intune.)

- **Authentication requirements**:
    - Who will use this cloud service? (Is it the same users who exist in the on-premises AD?)
    - Do any partner organizations wish to access cloud/on-prem services? (If the answer is yes, we can implement Azure AD B2B instead of AD FS.)
    - Do you wish to allow external users to use their existing social identities to authenticate to cloud services? (If the answer is yes, we can use Azure AD B2C for that.)
    - Do users require SSO? (This can be easily achieved using Azure Seamless SSO.)
    - What are the supported authentication technologies for applications? For example, NTLM, OAuth, SAML, Kerberos, and so on (Azure AD supports legacy authentication methods as well as advanced methods such as SAML, OAuth, and OpenID).

    - Does the business allow you to sync password hashes to the cloud? Does the password hash sync comply with company compliance and legal requirements? (If the answer is no, we need to either use AD FS or Pass-through Authentication.)

- **Security requirements**:
    - Is the organization happy to control access to resources based on device state, location, and sign-in risk? (If the answer is yes, we can implement Conditional Access to do this.)
    - Does the user require MFA? (This can be done using Azure MFA, for cloud users as well as on-prem users.)
    - Does the organization wish to identify risky activities related to identities? (Azure Identity Protection can detect potential vulnerabilities of your identity infrastructure.)
    - Does the data that's stored/used in the cloud and on-prem applications need protection? (We can use cloud app security and Azure Information Protection to protect sensitive data.)
    - Does the organization want to protect privileged accounts with advanced security measures? (We can provide just-in-time privilege access using Azure AD Privileged Identity Management.)
    - How we can transform from a perimeter-defense security approach to a zero-trust approach?
    - How we can prevent lateral movement in a hybrid environment?
- **Monitoring/reporting/alerting requirements**:
    - Does the organization wish to review sign-in logs, error logs, and audit logs from a centralized location? (We can collect and review logs from centralized locations by enabling log analytics for Azure AD.)
    - Does the organization wish to get insights into the overall identity infrastructure's security? (Azure AD's identity secure score, security overview, and Azure Security Center can provide lots of insight into the health of the current identity infrastructure and even recommend things we can do to improve it.)
    - How we can identify potential security risks? (We can use Microsoft Defender for identity to identify pass-the-hash, pass-the-ticket, golden ticket, and silver ticket attacks. We can also identify lateral movement attempts and other identity-related security risks.)

Apart from the above technical evaluation, we also need to consider the business needs, which could affect the design:

- What sort of investment can the business afford on the company's hybrid journey? (This will decide what licenses and features the business can afford.)
- What is the business' cloud strategy? (Does the company have plans to move to the cloud completely? What sort of cloud services is the company looking to use?)
- Does the company expect any M&A? (This will help to design a system to support future merges or acquisitions.)
- Will the hybrid approach have an impact on current compliance? (This will help us to understand what needs to be considered in the design to make sure it matches the business compliance requirements.)

Synchronization

Synchronization controls how your identities appear in the cloud. In a typical AD environment, engineers do password changes, name changes, group membership changes, and add/remove custom attributes. In a hybrid environment, the cloud identity should represent the same characteristics as an on-prem identity. This is why synchronization is crucial. Azure AD Connect is a Microsoft tool that was designed to sync on-prem identities to the cloud.

Azure AD Connect has five main features:

- **Synchronization services**: This service checks whether Azure AD has the same identities and attributes as on-prem AD. If it doesn't match, it will replicate relevant objects and changes to Azure AD (we can decide what data it should consider for the comparison).
- **Federation service**: Azure AD Connect can be used to provide a hybrid identity via an on-prem AD FS farm. This is mainly used when organizations don't want to sync password hashes to Azure AD.
- **Password hash synchronization**: Azure AD Connect can sync user password hashes from on-prem AD to Azure AD.
- **Pass-through authentication**: This feature allows users to authenticate to Azure AD using the same password without a password hash sync or federated environment. More information about this feature can be found in *Chapter 18, Hybrid Identity*.

- **Monitoring**: Azure AD Connect monitors the health of Azure AD Sync. These stats can be viewed using the Azure portal.

Let's look at a few things you need to consider when you're deciding on synchronization:

- **On-prem AD topology**: Azure AD connect support two types of configurations:
    - **Single AD forest – single Azure AD**: This is the most commonly used deployment topology. When a user has a single AD forest, it can be synced to one Azure AD tenant. Even if it has multiple domains, it can still be used with one AD tenant. The Azure AD Connect express setup only supports this topology. However, at any given time, only one of the Azure AD Connect servers can sync data to the Azure AD tenant. For high availability, staging server support is available.
    - **Multiple AD forest – single Azure AD**: Some organizations have multiple AD forests for various reasons. Azure AD has support for syncing identities from all the forests into one Azure AD tenant. Each AD forest can have multiple domains as well. The AD Connect server should be able to reach all the forests. The Azure AD Connect server can be placed in a perimeter network and then allowed access to different forests from there. A rule of thumb in this model is to represent a user only once in Azure AD. If a user exists in multiple forests, it can be handled in two ways:
        - We can set it to match the user's identity using the mail attribute. If MS Exchange is available in one or more forests, it may also have an on-prem GALsync solution. GALsync is a solution that is used to share Exchange mail objects between multiple forests. This will allow you to represent each user object as a contact in other forests. If a user has a mailbox in one forest, it will be joined with the contacts in the other forests.
        - If users are in an account-resource forest topology that has an extended AD schema with Exchange and Lync, they will be matched using ObjectSID and ExchangeMasterAccountSID.
    - **Password hash sync or federated**: If the organization is allowed to use a password hash sync to the cloud, it is the easiest way to get the identities to sync to the cloud. If it isn't, there are two options.

      If the organization is already using federation, we can use the same method to authenticate to Azure AD. To do this, we need to maintain a highly available AD FS environment. Pass-through Authentication, on the other hand, provides similar functionality, but it is all based on agents.

You don't have to maintain different server farms for it. You will learn how to implement this feature in Chapter 18: Hybrid Identity .

- **Directory extensions**: There are AD integrated applications that require AD schema extensions in order to work. Normally, this initial schema modification happens during the installation process. Also, the required engineers can add custom attributes to the AD schema and use them in their own applications (this is further explained in *Chapter 7*, *Managing Active Directory Objects*). Azure AD Connect can sync these custom attributes to Azure AD. This allows businesses to build their own applications in the cloud and use values from these custom attributes. This requires additional configuration in Azure AD Connect. Therefore, it is important to gather this information before you finalize the design.
- **Password writeback**: In most cases, organizations like to keep control of their passwords for the on-prem AD. So, if a user needs to reset their password, this needs to be done in the on-prem AD and then synced to Azure AD (if password hash sync in use). If required, we also can allow users to reset their passwords in Azure AD and write them back to the on-prem AD. This can also be done via the Azure AD Connect service.

Shared responsibility

Security in public cloud services such as Azure is not only the CSP's responsibility, it is also the responsibility of the customer. The shared responsibility model helps us to understand who protects what in the cloud environment. For PaaS and SaaS, IAM is a shared responsibility between the customer and the CSP. This requires a plan which includes activities such as managing enterprise identities, implementing PIM controls, implementing password-less authentication, implementing Defender for Office 365, and so on. The CSP provides various security products and services that can help to improve the security of PaaS and SaaS solutions. It is the customer's responsibility to decide which solutions to choose and invest in accordingly. Also if the customer is in a hybrid environment, it is the customer's responsibility to make sure that if there is a breach, it is not spread into cloud services. In the *Solorigate* attack, once attackers forced an initial breach, they laterally moved and gained control of ADFS servers.

Then they forged SAML tokens to authenticate into cloud services. In this scenario, it is the customer's responsibility to implement relevant security control in the on-premises environment to protect digital identities.

This could have been prevented by enforcing the zero-trust model, implementing the enterprise access model for PAM, and implementing **Privileged Access Workstations** (**PAWs**). Also, the customer could have used services for the CSP such as Defender for Identity (with support for ADFS servers), and Defender for Endpoint to identify potential risks. As we can see, when it comes to identity both parties should work side by side.

Cost

Last but not least, cost also has an impact on the design of the hybrid identity. AD DS services don't cost you extra as they come with the Windows Server operating system. Azure AD is a managed service and it comes at a cost. There is a free version of Azure AD, but that has very limited features. Therefore, when you are proposing the design, you need to consider licensing costs as well. Most of the identity protection and data protection features of Azure AD are only available under Azure AD P1 and P2. If the organization wants to drop features because of the cost, make sure that they understand the damage it can do. More information about Azure AD licenses and features are available at `https://bit.ly/3nYhxtn`. You can also use the Azure price calculator to find out more about costs: `https://bit.ly/31yZEKh`.

In this section, we looked at things we need to consider when we're designing a hybrid identity. In *Chapter 16*, *Active Directory Security Best Practices*, and *Chapter 18: Hybrid Identity*, I will be demonstrating how to implement the services/features that were discussed here.

Summary

Design, implementation, and maintenance are key stages of any successful service deployment. In this chapter, we learned how to design the Active Directory infrastructure according to industry standards and best practices. The Active Directory infrastructure has two types of components: logical and physical. In this chapter, we learned about the design and placement of both types. As part of the design exercise, we also learned how to gather business data, how to identify risks, and how to do sizing.

We also looked into the design process of a hybrid identity. Here, we learned why a hybrid identity is important and what we need to consider during the design phase. We also learned how we can gather the required information from businesses using questionnaires.

In the next chapter, we are going to look into the DNS, which is the naming system for infrastructures.

4

Active Directory Domain Name System

We can't talk about **Active Directory Domain Services** (**AD DS**) without mentioning the **Domain Name System** (**DNS**). Since Windows Server 2003, DNS has become the primary name resolution service. Before that, Windows was using NetBIOS and the **Windows Internet Name Service** (**WINS**) for name resolution.

WINS and DNS are both TCP/IP network name resolution services. There are legacy systems that still use WINS instead of DNS.

DNS helps to locate resources on the internet and intranet. It can be a computer, server, service, or application. DNS can run as an independent server role on the intranet, perimeter network, or public network. There are different vendors who provide DNS solutions other than Microsoft; Linux/Unix **Berkeley Internet Name Domain** (**BIND**) is a good example of that. There are mainly two categories of DNS infrastructure. One category is organizations that host their own DNS servers to facilitate name resolution requirements for their corporate infrastructures. Another category is businesses that sell DNS as a service, such as Azure DNS, **Dynamic DNS** (**DynDNS**), and Amazon Route 53.

In this chapter, our main focus will be to understand how AD-integrated DNS works in the infrastructure. Throughout the chapter, you will learn about the following topics:

- What is DNS?
- Hierarchical naming structures

- How DNS works
- DNS infrastructure design
- DNS essentials
- Conditional forwarders
- DNS policies
- DNS server operation modes
- Zone transfers
- DNS delegation
- DNS service providers

If you are working with AD, you may already have a basic idea of how DNS works in an AD environment. Before we dive into DNS advanced topics, let's go ahead and refresh our knowledge of DNS basics first.

What is DNS?

On mobile phones, we have phone books. If we need to save someone's phone number, how we do that? Do we just enter the number and save it? No. We save the number with the person's name or something we can remember, so the next time we open the contact list, we can easily find it. The same applies when you are dealing with IP addresses. I remember a few of the most commonly used IP addresses in my clients' infrastructure, but I do not remember most others. I remember lots of servers by their hostnames rather than their IP addresses. This is because hostnames are more user-friendly and are easier to remember than IP addresses. This is exactly what DNS does: it maps IP addresses to domain names or common terms that are user-friendly.

As I stated, there can be no functioning AD domain infrastructure without DNS. There are two main reasons why AD DS needs DNS:

- **Maintaining hierarchical infrastructure design**: In the previous chapters, I talked about designing the AD infrastructure. I mentioned implementing multiple forests, domains, and child domains. We use domain namespaces to separate them from each other and build the AD hierarchy. The only way you can reflect that logical structure infrastructure is by using DNS.
- **Locating domain controllers**: Devices in infrastructure need to communicate with AD domain controllers for authentication. If it's a remote site, it needs to locate its closest domain controller for authentication. This process is done using DNS **service** (**SRV**) records. Also, if an application or service needs to locate a host or resources, DNS will help resolve that.

Before DNS, systems were using **LAN Manager Hosts** (**LMHOSTS**) and hosts files to map IP addresses to hostnames. This method is still used in small networks. The LMHOSTS file helps find NetBIOS names in TCP/IP networks. The hosts file helps to find domain names in TCP/IP networks. This is also used to override the DNS entries as in the name resolution process, the hosts file still gets priority.

The LMHOSTS and hosts files are located at `C:\Windows\System32\drivers\etc`:

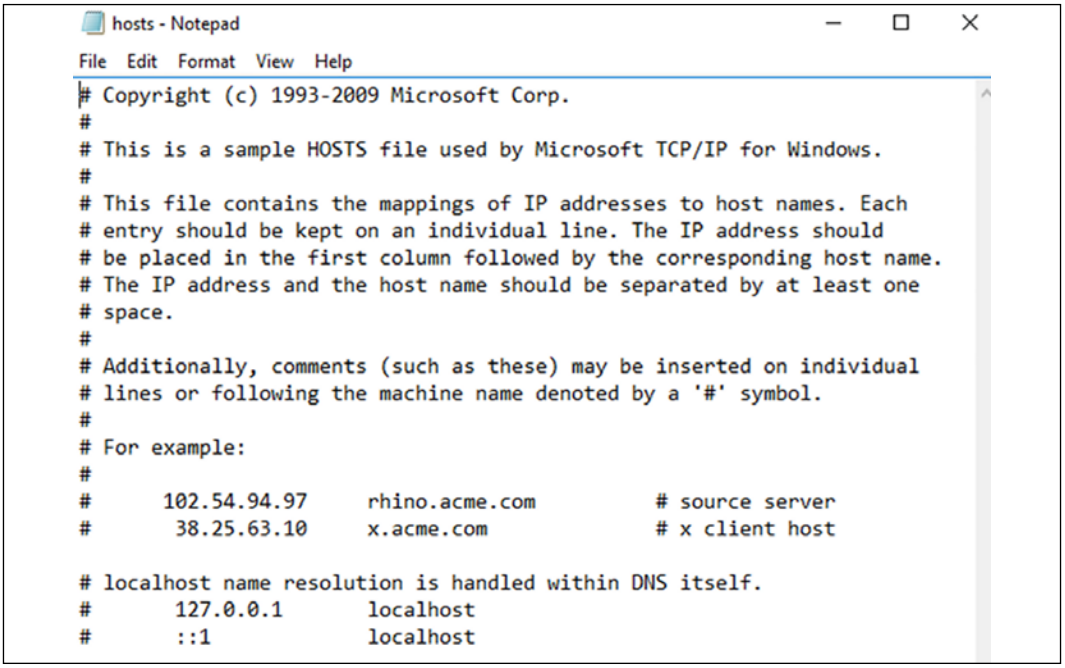
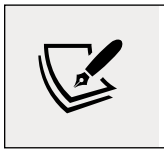


Figure 4.1: The hosts file

DNS was invented to support email communication in the **Advanced Research Projects Agency Network** (**ARPANET**). In the past, people were using the LMHOSTS and hosts file, and as networks grew, it wasn't a practice to maintain large hosts files. The first conversation to start a better, centralized name resolution system started with RFC 606 in December 1973. It took almost a decade with several RFCs to decide on the technology outline for modern DNS, and the final RFCs were released in November 1983 (RFC 881, 882, 883).

DNS maintains a database that contains various DNS data types (A, MX, SRV, and AAAA). This database can be distributed among multiple servers. If required we also can maintain a read-only copy of the database where we cannot guarantee infrastructure security.



In Windows Server, DNS data is saved under `C:\Windows\System32\dns`. It contains DNS zone files and Root Hints configuration files (cache.dns).

Hierarchical naming structures

In *Chapter 1*, *Active Directory Fundamentals*, we looked into domain trees and explored how they can be used to organize the domain structure in the hierarchical method. DNS allows us to translate this logical structure into the domain namespace. Similar to a tree, it starts from the root and is spread into different layers, such as branches and leaves. In the domain tree, the root is represented by a dot (`.`). A typical tree branch contains many leaves. In the domain tree, a branch represents a collection of named resources, and a leaf in a branch represents a single named entry. In a tree, branches and leaves depend on each other. Branches and leaves are part of one system until everything is attached together. When we describe a leaf or a branch, we explain it with the relationship to the tree. For example, if I need to show someone a leaf of an apple tree, I will call it an apple leaf. Then, the person knows it's a part of an apple tree.

In the following diagram, **Level 1** represents the **Top-Level Domains** (**TLDs**). These are managed by the internet name registration authority according to international standards:

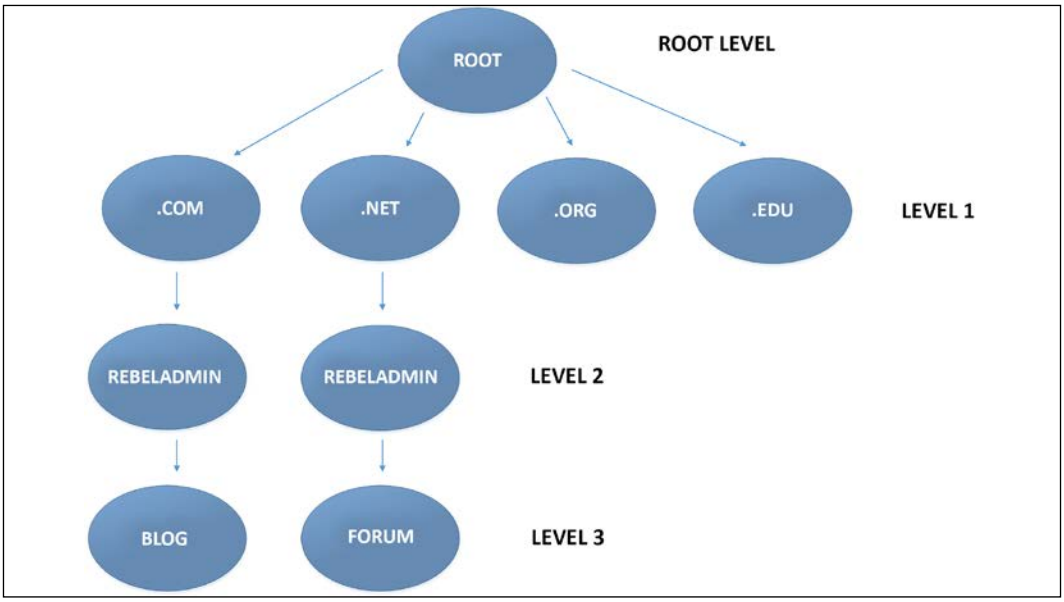


Figure 4.2: Domain naming hierarchy

Top-Level Domain managers (TLD managers)

Internet Assigned Numbers Authority (**IANA**) has administrative responsibility of TLDs. TLDs are categorized into two different classes called **country-code TLDs** (**ccTLDs**) and **generic TLDs** (**gTLDs**). The administrative responsibility of ccTLDs is delegated to individual country managers. TLD managers are responsible for the delegated domain, and have a duty to serve the community. TLD managers are doing a public service on behalf of the global internet community. The list of current TLD managers is available on `https://bit.ly/3raaQaK`. Also, more information about internet DNS structure, delegation, and TLD managers is available on `https://go.icann.org/3FJVCxv`.

The following table describes common TLDs:

TLD name	Description
`.com`	This is the most commonly used TLD and it is mainly used to register businesses that are focused on profits. This is also used to register sites such as personal websites, blogs, and community websites. It is open for any person to register.
`.org`	This is mainly used by non-profit organizations and communities. It is open for any person to register.
`.net`	This is used to represent distributed computer networks. It is open to the public and anyone can register a name under it.
`.edu`	This is a limited registration for educational institutions.
`.gov`	This is limited to government institutions.
`.ca`, `.co.uk`	These are used to represent countries. The registration of a domain under these depend on the rules and regulations of their country's domain name registrar authority.

The complete list of TLDs can be viewed at `https://bit.ly/3oPNaWI`.

Level 2 in the hierarchy is controlled by the organization. In my example, I need two domain names called `rebeladmin.com` and `rebeladmin.net` for my organization. I can go to a domain name registrar (for example, GoDaddy or Dotster) and register the names.

Once I register it, no one else can have the same name without my authority, unless the subscription expires. As the domain owner, I can create any sub-domain (child domain) under `rebeladmin.com` and `rebeladmin.net`. In DNS, there are some records that need to be published on the internet. For example, if I need a website hosted under my domain that people can access via the internet, I need to map my web server's public IP address to the `www.rebeladmin.com` DNS record.

In order to do that, I need an **authoritative server**. This DNS server will hold a copy of all the DNS records related to `https://bit.ly/3nI6pSF`. If another DNS server (recursive DNS server) queries DNS records related to `https://bit.ly/3cG0JCi`, this authoritative server will be able to answer. **Authoritative servers** need to be highly available as well. When you register the domain name, the domain registrar will use their own DNS servers as authoritative DNS servers by default. But if we need to, we can have our own DNS servers and point to them using **nameserver** (**NS**) records.

In **Level 3**, domain owners have complete authority to create any sublevel DNS namespace. In my example, they are `blog.rebeladmin.com` and `forum.rebeladmin.net`. Each dot (`.`) represents each level of the domain tree:

Figure 4.3: Domain name example

The leftmost part represents the lowest level in the domain tree and the rightmost part shows the domain root.

If you own a domain name, you can use the same name to represent your AD domain name, but there are a few things you need to consider doing:

- **Manually set up DNS records to match public DNS records**: As an example, I am going to use `rebeladmin.com` as my AD domain name. I am using domain registrar DNS servers to set up the public DNS records. I have a website, `www.rebeladmin.com`, and it points to the `38.117.80.2` public IP address. I also use AD-integrated DNS servers to maintain local infrastructure DNS records. But when I go to access the `www.rebeladmin.com` website from a domain PC, first I need to make sure it is not trying to resolve the name using a local DNS server. If a DNS record is needed to resolve to external IP addresses, we need to create the relevant records manually under a local DNS server.
- **Modify DNS records to use the lowest routing path**: In my previous example, the web server's public address is `38.117.80.2`. This web server can also be accessed via its local IP address, `192.168.0.100`. If I traceroute to `www.rebeladmin.com` with the `38.117.80.2` DNS record, it takes nearly 10 network hops to reach the web server. But if I use the `192.168.0.100` local IP address, it gets resolved with one network hop. Therefore, if your domain name has public and private DNS entries, make sure that you make adjustments and help users use the lowest routing path in order to improve performance and accessibility.

Maintaining the same domain namespace on two infrastructures is called a split-brain DNS structure. This will require manual record adjustments in both infrastructures in order to maintain service availability and integrity. The recommended setup for a similar infrastructure is a whole-brain DNS structure. Linking both DNS namespaces into one using a standard DNS name resolution mechanism will reduce manual user interaction.

How DNS works

A few days ago, I posted a birthday card to my mother who lives in Sri Lanka. I posted it from the local post office in Kingston upon Thames, England. Once I put it inside the post box, the delivery process started, and now it was the postal service's responsibility to deliver it to the correct person. So, when the local post office worker picked up my letter, did they know my parents' exact house location? No, they didn't. But at the end of my address, it said the country was Sri Lanka. They then knew that if this letter goes to Sri Lanka, then the postal service there will be able to deliver it. So, the next stop of the mail was Sri Lanka. Once the card reached the main postal sorting facility in Sri Lanka, would the worker who picked up the letter know the exact address location? Maybe not, but if they didn't, they could look for the city that it should be delivered to. Then, the post office in that city would know what to do. Once the letter reached the city's mail sorting facility, the worker would know which local post office this letter should be delivered to.

Once it reached the local post office in my parents' city, the postal carrier there would definitely know where my parents' house is located. In the end, even though my local post office in Kingston did not know where my parents live, the letter got delivered. How did they do it? Even though they did not know the end delivery location, they knew someone who could figure it out at each stage. This is exactly how DNS resolves addresses too. The DNS server in your infrastructure isn't aware of billions of websites on the internet and their IP addresses. When you type a domain name and press *Enter*, your DNS server will try to resolve it, but if it doesn't know how, it will work with other DNS servers that may know about it and work with them to find the correct destination.

In the following example, my friend William is trying to access the `www.rebeladmin.com` website from his laptop. This device is a domain-joined device and the user logs in to the device using the domain username and password.

In order to get the IP address of the `rebeladmin.com` web server, the DNS client in William's PC needs to perform a DNS query from its DNS server.

A DNS query can happen between the DNS client and the DNS server or between DNS servers.

There are two types of DNS queries:

- **Recursive**: A recursive query is usually sent by DNS clients. Once the query is processed, it expects a success or failure response from the DNS server. It is the DNS server's responsibility to work with other DNS servers in order to provide an answer if it cannot process the query by itself.
- **Iterative**: Once the DNS server receives an iterative query, it will respond with the best answer it has by looking at its DNS zones and caching. If it cannot resolve the query, it will respond with a negative answer without taking any help from any other DNS server.

In the following example, a user in an AD environment is trying to access the `https://bit.ly/3DMX5Ct` website. This website is hosted in the external network. Let's go ahead and see how this DNS query works:

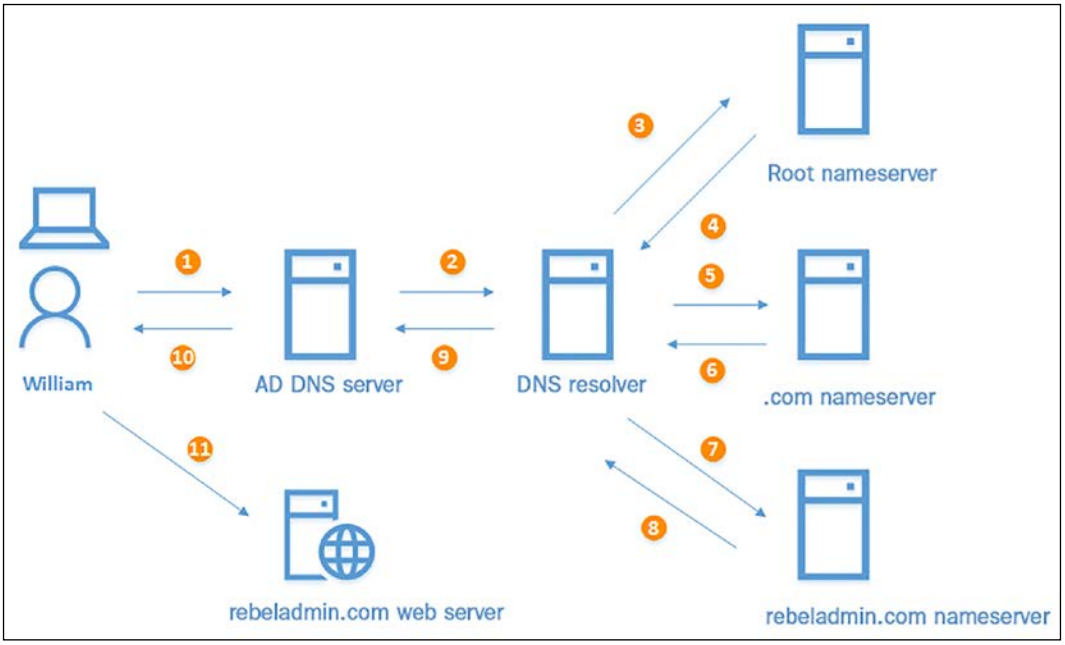


Figure 4.4: DNS query example

The following steps explain the preceding diagram:

1. William's laptop sends the recursive DNS query to the AD DNS server in order to resolve the IP address for the `www.rebeladmin.com` website. This DNS server information is defined in the IP configuration of the laptop.
2. The AD-integrated DNS server tries to resolve the name to an IP address, but it cannot. It doesn't have a relevant DNS record configured for it. However, it knows a DNS server that can help resolve this. This is defined as a DNS forwarder in the DNS server configuration. Normally, it will be the **Internet Service Provider's** (**ISP's**) DNS server or a public DNS server, such as Google's.

   In order to check the DNS forwarder address on your DNS server, you can use the following PowerShell command:

   ```
   Get-DnsServerForwarder
   ```

   The following screenshot shows the output for the preceding command:

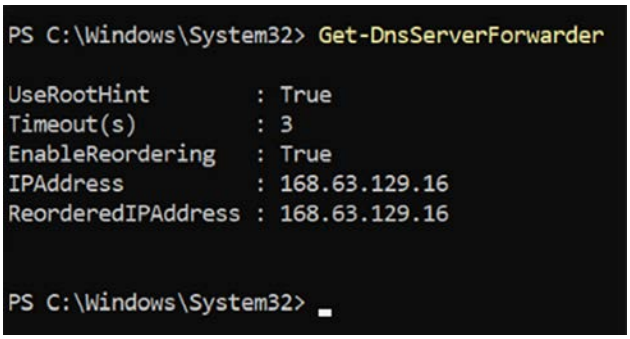
   

   Figure 4.5: DNS forwarders

3. If the DNS resolver does not have a record in its cache or local zone, then it will send an iterative query to the root NS. There are 13 root servers clustered and operating from different geographical locations. They are controlled by IANA.

Microsoft DNS uses an option called Root Hints. By default, this contains all 13 root servers. If you do not have the forwarder set up in your DNS server, it will use these root servers to resolve DNS queries:

```
PS C:\Windows\System32> Get-DnsServerRootHint

NameServer          IPAddress
----------          ---------
j.root-servers.net. 2001:503:c27::2:30
b.root-servers.net. 2001:500:200::b
f.root-servers.net. 2001:500:2f::f
m.root-servers.net. 2001:dc3::35
a.root-servers.net. 2001:503:ba3e::2:30
h.root-servers.net. 2001:500:1::53
d.root-servers.net. 2001:500:2d::d
c.root-servers.net. 2001:500:2::c
k.root-servers.net. 2001:7fd::1
e.root-servers.net. 2001:500:a8::e
g.root-servers.net. 2001:500:12::d0d
i.root-servers.net. 2001:7fe::53
l.root-servers.net. 2001:500:9f::42
```

Figure 4.6: Root DNS servers

4. Since it's an iterative query, the root server will look into it and respond saying, "I do not know about `rebeladmin.com`, but I know someone who knows about `.com`," and it attaches the information about the `.com` TLD NSes to the response.
5. Based on the response from the root server, the DNS resolver sends an iterative query to the `.com` TLD NSes.
6. The TLD NSes look into the query and respond by saying, "I do not know about the IP address of `rebeladmin.com` but I know NS information for `rebeladmin.com`; go and check with them – they may be able to help you."
7. Now, the DNS resolver knows about the `rebeladmin.com` NS and sends an iterative query to it by asking for the IP address for `www.rebeladmin.com` (a record).
8. The question finally came to the right person who can answer it. The `rebeladmin.com` NS responds with the IP address for `www.rebeladmin.com`.
9. Now, the DNS resolver knows the answer, and it responds to the AD DNS server.
10. Then, the DNS client on William's laptop gets the response for its recursive query with the IP address for `www.rebeladmin.com`.

11. William's laptop connects to the `rebeladmin.com` web server and views the website.
12. William's laptop DNS client will also cache the DNS query result so it can use it next time without going through the whole process.

This is how the DNS request will be processed from top to bottom in an infrastructure. This will not be exactly the same on each and every request, as DNS servers will cache the data. If DNS servers can find the answer from their cache, the process will be quicker.

DNS infrastructure design

In the first chapter, I have mentioned how AD domain and forest represent the logical structure of AD setup. We also need to design DNS infrastructure to support the AD logical structure.

AD DS must require integration with DNS. Otherwise, the clients will not be able to locate domain controllers. The DNS infrastructure design mainly has two models:

1. The organization already has existing DNS infrastructure and they'd like to keep it. If that is the case, we need to integrate the existing DNS infrastructure with AD namespace. This involves deploying new DNS servers and DNS delegation.
2. The organization doesn't have DNS infrastructure at all. In such a situation, it's easier to implement new DNS infrastructure along with the new AD setup process. It simplifies the maintenance and administration process.

From these two models, integration with existing DNS infrastructure can be challenging. Let's go ahead and explore more about this model.

Integrate AD DS with existing DNS infrastructure

When we integrate AD DS with existing DNS, we need to consider the following best practices:

1. It is required to install a DNS role in each domain controller. This way the domain controller does not have to depend on another server to resolve DNS queries. Also, we do not need to move zones or servers.
2. Configure each regional domain controller to host the DNS zone related to their own domain. This is also to reduce the dependencies.

3. Replicate the zone containing AD forest-wide locator records to all the DNS servers. This helps replication partners to find each other and also to find global catalog servers.

Configure the forest root domain controller to host AD forest DNS zone. Instead of depending on an existing DNS namespace, we also can implement a new DNS namespace just for AD DS. This is called disjoint naming space configuration.

Disjoint naming space

In this method domain members can end up with two DNS names. As an example, Rebeladmin Inc. had an existing DNS set up with `hq.rebeladmin.com`. The new AD DS setup has the DNS namespace `ad.rebeladmin.com` DNS suffix. ComputerA already has a DNS record called `ComputerA.hq.rebeladmin.com`. When ComputerA is added to the domain, the system will register the DNS record in the AD-integrated DNS zone as `ComputerA.ad.rebeladmin.com`. We can use both DNS names for ComputerA. This is called disjoint naming space. This is very complex to maintain and manage. Before we consider using disjoint naming space, we need to consider the following:

1. Even though Windows operating systems support disjoint namespace, we need to confirm if the applications can support disjoint namespace configuration.
2. The disjoint namespace suffix should not match another forest or domain name that required a "trust." This will not work as the routing fails.
3. We need to use group policies or DHCP service parameters to set the DNS suffix search order to optimize the name resolution.
4. Applications (especially custom-made) must be tested for compatibility issues. Use a lab environment for testing and also if possible confirm with the vendor before the disjoint namespace implementation.

If an organization does not have a DNS namespace, we can create a DNS namespace as part of the AD DS configuration process. This is very straightforward and easy to maintain.

Deploying AD-integrated new DNS infrastructure

By using the AD DS Installation Wizard, we can create new AD DS-integrated DNS infrastructure. As part of the process, the system creates AD-integrated DNS zones to represent the DNS namespace.

There are a few advantages of using AD-integrated DNS zones:

- They use the same AD replication topology for zone replication. We do not need to configure and maintain a different replication topology.
- Any authoritative domain controller can update DNS records and it will be replicated to other domain controllers (multi-master).
- By using secure dynamics updates, we can control who can update existing DNS records.
- Once the initial setup process is completed, we also can create our own AD-integrated DNS zones to have the same above benefits.

DNS essentials

In a Windows Server environment, the DNS service can be run as an individual service or as an AD-integrated service. Either way, core DNS components, technology, and terms will be the same for both scenarios.

DNS records

The DNS database holds various types of resource records. In an AD-integrated DNS setup, most of these records will be created automatically when adding resources to the domain, changing settings in resources, or promoting/demoting domain controllers (SRV records, A records, and AAAA records). However, in an infrastructure, some resource records may still need to be created manually in DNS servers (static).

Start of authority record

Each DNS zone must have a **start of authority** (**SOA**) record, and it is created when a zone is created for the first time. This record provides lots of general information for the DNS zones, such as:

- **Primary server**: The best DNS source for the zone.
- **Responsible person**: The email address of the zone administrator.
- **Serial number**: A number that is used to track the zone changes. If there is a secondary DNS zone on another server, when it checks the updates with the master zone, it will compare this serial number. If the secondary zone has a lower serial number than the master zone, it will update its copy.
- **Refresh interval**: This value is defined when the secondary server should check for zone updates with the master server.

- **Retry interval**: This value defines how long the secondary server should wait to try for updates if the first request is unsuccessful.
- **Expires after**: If the secondary server can't refresh the zone before the value defined here, it will no longer consider the secondary server as self-authoritative.
- **Minimum (default) TTL**: If the **time-to-live** (**TTL**) value is not defined in the resource records, it will use the default TTL value defined in here. This TTL value is attached to the response of DNS-resolved queries.

The following PowerShell command can be used to view the properties of an SOA record:

```
Get-DnsServerResourceRecord -ZoneName "REBELADMIN.COM" -RRType "SOA" |
Select-Object -ExpandProperty RecordData
```

In the preceding command, `REBELADMIN.COM` can be replaced with any zone. `Select-Object -ExpandProperty RecordData` is used to expand the output.

A and AAAA records

Host records are used to map a **fully qualified domain name** (**FQDN**) to an IP address. These records are used for IPv4, and AAAA records are used for IPv6. In AD and integrated DNS, every device will have an A or AAAA record when it is added to the domain. If the IP addresses are changed, the system will automatically update its DNS record, too.

FQDN contains the hostname and domain name. As an example if we consider the `DC01.rebeladmin.com` FQDN, DC01 is the hostname of the object and `rebeladmin.com` is the domain name.

To add an `A` record, run the following command:

```
Add-DnsServerResourceRecordA -Name "blog" -ZoneName "REBELADMIN.COM"
-IPv4Address "192.168.0.200"
```

To remove an `A` record, run the following command:

```
Remove-DnsServerResourceRecord -ZoneName "REBELADMIN.COM" -RRType "A"
-Name "blog"
```

To list `A` records in a zone, run the following command:

```
Get-DnsServerResourceRecord -ZoneName "REBELADMIN.COM" -RRType "A"
```

In the preceding commands, `REBELADMIN.COM` can be replaced with any zone name.

NS records

NS records are used to list authoritative DNS servers for the zone. In an AD-integrated DNS setup, all the domain controllers with the DNS role installed will be added to the NS records in the zone file. Having multiple name servers will add redundancy to the DNS setup.

The following command can list the NS for a zone. `REBELADMIN.COM` can be replaced with any zone name:

```
Get-DnsServerResourceRecord -ZoneName "REBELADMIN.COM" -RRType "NS"
```

Mail exchanger records

Mail exchanger (**MX**) records specify the MX server for a domain. The MX can be any email server (including Microsoft Exchange, Exim, or Office 365). A domain can have multiple MX records and the mail server priority number will be used to maintain the priority order. The lowest value will get the highest priority.

Canonical name records

Canonical name (**CNAME**) records are aliases for FQDN. They work similarly to nicknames. For example, I have an `A` record set up for `blog.rebeladmin.com`; there is a blog running under it. Some time ago, I used `my.rebeladmin.com` for the same blog. If users still use `my.rebeladmin.com`, I need them to still see my blog at `blog.rebeladmin.com`. To do that, I have to use CNAME records.

Pointer records

Pointer (**PTR**) records are used to map IP addresses to FQDN. Some call it reverse DNS records as well. The reverse lookup zone will not be created when the DNS is set up with AD; instead, it will need to be created manually. If you have multiple address spaces, you will need to create separate reverse lookup zones to represent each address space.

SRV records

SRV records are used to specify the location of a service inside an infrastructure. For example, if you have a web server in the infrastructure, by using an SRV record, you can specify the protocol, service, and domain name and then define the service location.

In an AD environment, SRV records are important as they help to locate the nearest domain controllers. In the previous chapters, I explained AD sites: when a user logs in, the system needs to point the user to the site's local domain controller instead of the domain controller in the hub. This is done via SRV records.

In an SRV record, the following information can be specified:

- **Service**: This will define the service that belongs to the SRV record.
- **Protocol**: This will define the protocol it will use. It can be either **Transmission Control Protocol** (**TCP**) or the **User Datagram Protocol** (**UDP**).
- **Priority**: This will define the service priority (only if the service supports this function).
- **Weight**: This will help define the priority of the same type records.
- **Port number**: This will define the service port number.
- **Host offering this service**: This will define the server offering this particular service. It needs to use FQDN.

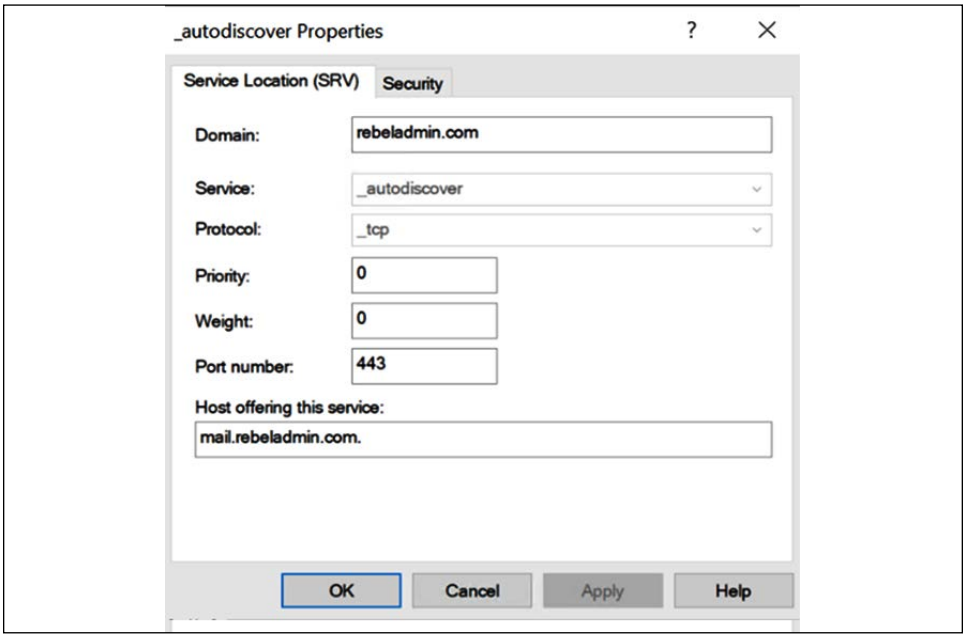


Figure 4.7: Sample SRV record

The AD-integrated DNS environment has a set of default SRV records created.

SRV records can be listed using the following command:

```
Get-DnsServerResourceRecord -ZoneName "REBELADMIN.COM" -RRType "SRV"
```

Detailed output can be viewed using the following command:

```
Get-DnsServerResourceRecord -ZoneName "REBELADMIN.COM" -RRType "SRV" |
Select-Object -ExpandProperty RecordData
```

In the preceding commands, `REBELADMIN.COM` can be replaced with any zone name.

Zones

The Microsoft DNS server supports four types of zones explained in the following sections. Each of these zones has different responsibilities and characteristics within the DNS namespace.

Primary zone

A primary zone is a read/write container that contains the DNS records for a domain. When you create the first domain controller with integrated DNS in a domain, it will create a primary zone by default. It holds a master copy of the zone and stores it in AD DS. Primary zones will be the source for secondary zones. Users are allowed to create primary zones and integrate them with AD DS, even though they are not related to the AD DS domain name.

There are two types of primary zone:

- Standard primary zones
- AD-integrated primary zones

Standard primary zones are mostly used in non-AD environments to manage DNS. As an example, if you are hosting a public-facing website, you can also install a DNS role and manage the DNS for it by yourself. But a DNS server doesn't necessarily have to be a domain controller or a member server of an AD environment. A standard primary zone will save its data in a data file located in the `c:\windows\system32\DNS` folder. The first server to host the standard primary zone becomes the master server. The DNS records under the primary zone can only be updated via the master server.

An AD-integrated primary zone stores data in AD. It uses the multi-master replication method. Therefore, we can update DNS records from any available domain controller and it will be replicated to all other domain controllers. AD-integrated primary zones also have faster replication, as they are part of AD replication topology.

In an AD-integrated DNS setup, a primary zone can be created using the following command:

```
Add-DnsServerPrimaryZone -Name "rebeladmin.net" -ReplicationScope
"Forest" -PassThru
```

This will create an AD-integrated primary DNS zone for the `rebeladmin.net` domain. Its replication scope is set to the forest, which means it will replicate to any DNS servers running on domain controllers in the `rebeladmin.net` forest. If you use the wizard to create the primary zone, the replication scope is set to the `rebeladmin.net` domain by default, which refers to any DNS servers running on domain controllers in the `rebeladmin.net` domain.

Primary zones are the only DNS zones that can be edited. Standard primary zones should have a backup zone for redundancy purposes, and this is called a secondary zone. Let's go ahead and see what a secondary zone is and how it works.

Secondary zone

A secondary zone keeps a read-only copy of a primary zone. It needs to refresh the zone data by contacting the primary zone hosted on another server. Reliable network connectivity and sufficient zone transfer permissions are a must to maintaining a healthy secondary zone. Secondary zones cannot be stored in AD DS.

I have an AD-integrated primary zone called `rebeladmin.net`. I have a standalone DNS server, and for application requirements, I need to set up a secondary zone in it.

Before I set up the secondary zone, I need to adjust the permissions for zone transfer. By default, zone transfer is not allowed in AD DS-integrated zones:

```
Set-DnsServerPrimaryZone -Name "rebeladmin.net" -SecureSecondaries
TransferToSecureServers -SecondaryServers 192.168.0.106
```

In the preceding command, `rebeladmin.net` is my zone and `TransferToSecureServers` defines that the transfer will be allowed only for the listed secondary server, `192.168.0.106`.

If needed, configuration can be modified with `-TransferAnyServer` to allow transference to any server and `-TransferToZoneNameServer` to allow transference only to NS:

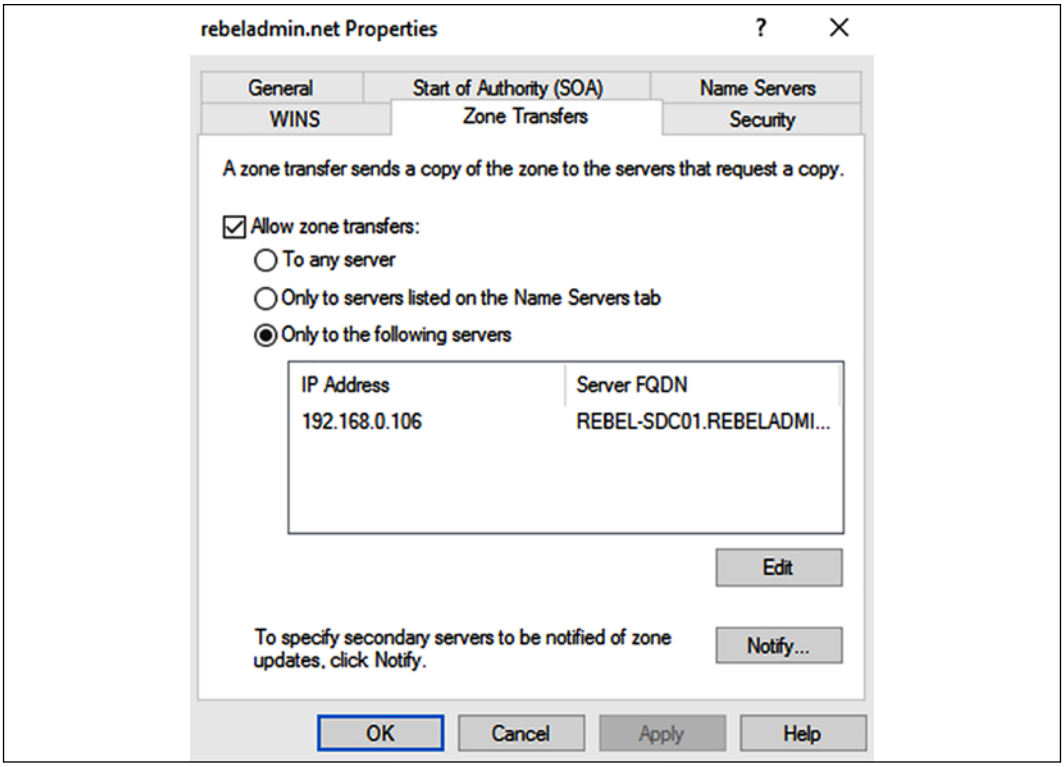


Figure 4.8: Zone transfer settings

Now, I can set up a secondary zone from the `192.168.0.106` server.

In the following command, `-MasterServers` defines the IP address of the master server. The `-ZoneFile` parameter is there, but only for file-backed DNS servers:

```
Add-DnsServerSecondaryZone -Name "rebeladmin.net" -ZoneFile
"rebeladmin.net.dns" -MasterServers 192.168.0.105
```

Stub zones

A stub zone is a read-only copy of a master zone but contains only SOA and NS records. It is not an authoritative DNS server for that particular zone. Stub zones are not a replacement for secondary zones, and they cannot be used for load sharing or redundancy purposes. Some think stub zones and conditional forwarders do the same thing, but they are completely different.

A conditional forwarder has a list of DNS servers that can help your DNS server to resolve DNS queries for a specific domain. Once your DNS server receives a query for that domain, it forwards the query to the servers list in the conditional forwarder. This may or may not include all the authoritative DNS servers for the domain. A stub zone is aware of all the authoritative DNS servers for its domain. When your DNS server receives a query for a stub zone, it goes ahead and queries directly from the servers listed on the zone and retrieves the result.

Reverse lookup zones

Reverse lookup zones hold PTR records. A reverse lookup zone can also be a primary zone, a secondary zone, or a stub zone. Reverse lookup zones will need to match the organization network segment. Even if it is AD DS-integrated, the system will not create a default reverse lookup zone. It needs to be created manually.

As an example, I need to create a reverse lookup zone for my `10.10.10.0/24` network. I can use the following command to do that:

```
Add-DnsServerPrimaryZone -NetworkID "10.10.10.0/24" -ReplicationScope
"Domain"
```

In the preceding command, `-ReplicationScope` is set to `Domain`. This means that it will be replicated to all domain controllers with integrated DNS in the domain.

We have now gone through all four types of DNS zones. We learned the different characteristics of each zone and also looked into the roles of each zone.

Conditional forwarders

In DNS servers, we use DNS "forwarders" to forward DNS queries to external DNS servers when it can't resolve them internally. Usually it will be the ISP's public DNS servers. We also can use public DNS servers from a third party such as Google as a forwarder:

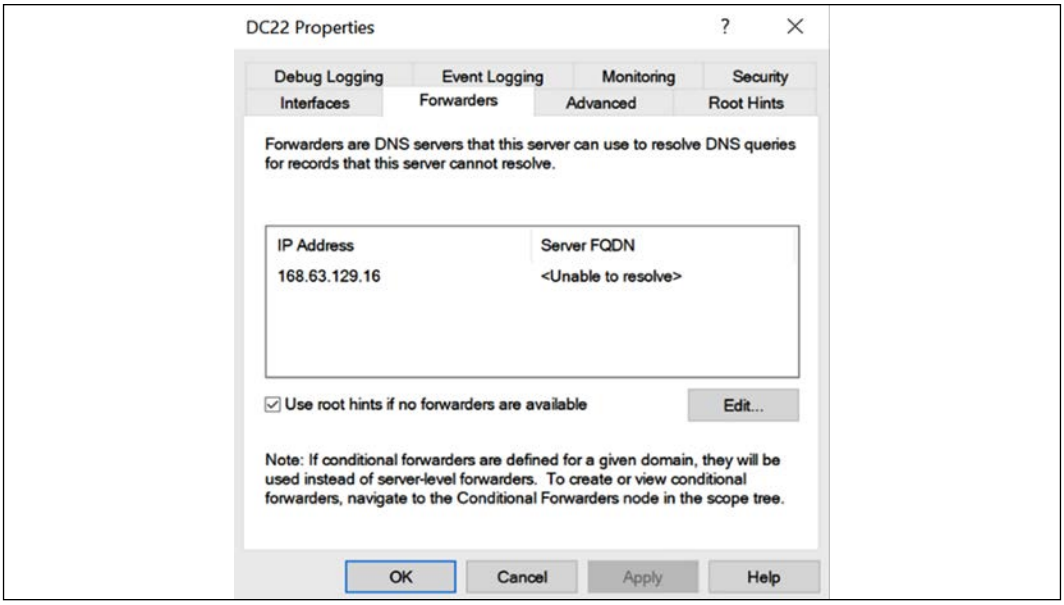


Figure 4.9: DNS forwarders

If the forwarders are not responding, the DNS server will use "Root Hints" to resolve the query.

Forwarders and Root Hints are used to resolve external DNS queries in general. But if we need to point DNS queries for a specific domain to a specific DNS server/s we can do that using "conditional forwarders." As an example, there is a partner company with the `rebeladmin.net` domain name. Their DNS server is `10.0.0.5`. These two organizations are connected together via a VPN connection. So if any user in the `rebeladmin.com` domain tries to resolve the hostname in `rebeladmin.net`, it should forward to `10.0.0.5`. To do that we can create a conditional forwarder by using

```
Add-DnsServerConditionalForwarderZone -Name "rebeladmin.net"
-ReplicationScope "Forest" -MasterServers 10.0.0.5
```

In the above, I am creating an AD-integrated conditional forwarder for the domain `rebeladmin.net`. Any DNS queries for `rebeladmin.net` will be resolved using the `10.0.0.5` DNS server. This configuration will be replicated to all the domain controllers in the forest.

Conditional forwarders are mainly used when an organization creates AD "trust" with another AD forest or domain. More information about AD trusts will be explained in *Chapter 11, Active Directory Services- Part 1*.

DNS queries via conditional forwarders are faster and more secure. When we resolve DNS queries using DNS forwarders, they use iterative queries back and forth to find the answers. But conditional forwarders use recursive queries, which make name resolution faster.

DNS policies

DNS policies were first introduced with Windows Server 2016 and are also available on Windows Server 2022. DNS policies mainly have the following capabilities:

1. **Geo-location based traffic routing –** Let's assume Rebeladmin Inc. uses two web servers to host its website `rebeladmin.com`. One of the web servers is located in Canada and the other one is located in the UK. Most of the website visitors are coming from these two regions and by maintaining the local datacenter, Rebeladmin Inc. is expecting to improve the user experience. By using a DNS policy, Rebeladmin Inc. can point USA/Canada clients to the Canada web server and Europe users to the UK web server.

We also can do this using Azure Traffic Manager. More details about its configuration is available on `https://bit.ly/30QoPr8`.

2. **Application load balancing –** The `rebeladmin.com` website has four web servers in four different datacenters in Canada. Each server has similar hardware capacity and by using DNS policies I can route 25% of traffic to each datacenter. Also let's assume `rebeladmin.com` has a web server running in the UK datacenter. But this server doesn't have better processing power. By using DNS policies I can route 50% of European traffic to the UK web server and the rest of the traffic to the Toronto web server.
3. **Time-based DNS response –** `rebeladmin.com` has two web servers; one is running in Canada and the other one is running from the UK. 70% of the site's users are from the USA and the busiest hours are between 7pm and 10pm. Recently USA users have been complaining about site performance. During these peak hours, the UK web server usage is very low.

Therefore by using DNS policies we can route 40% of USA user traffic at peak hours to the UK web server to improve the user experience.

4. **Split-brain DNS** – When users in the Rebeladmin Inc. office access `rebeladmin.com`, I need them to go to the internal IP address of the web server and if anyone else from external networks accesses `rebeladmin.com` it should go to the public IP address of the web server. This is called a split-brain DNS scenario and we can now manage this configuration by using DNS policies.
5. **DNS query filtering** – We can create DNS policies to filter queries based on the client subnet, server interface IP address, transport protocol, internet protocol, FQDN, query type, and time of day. As an example, we can create a DNS policy to block queries from the `40.114.1.0/24` range.

Let's go ahead and see the DNS policy in action. In my demo setup, I have a computer with IP address `10.0.0.6`. The DNS server is running on `10.0.0.4` and there is an A record in place for `portal.rebeladmin.com` that resolves to `10.0.0.7`. When I run `nslookup` from `10.0.0.6`, I receive the expected response:

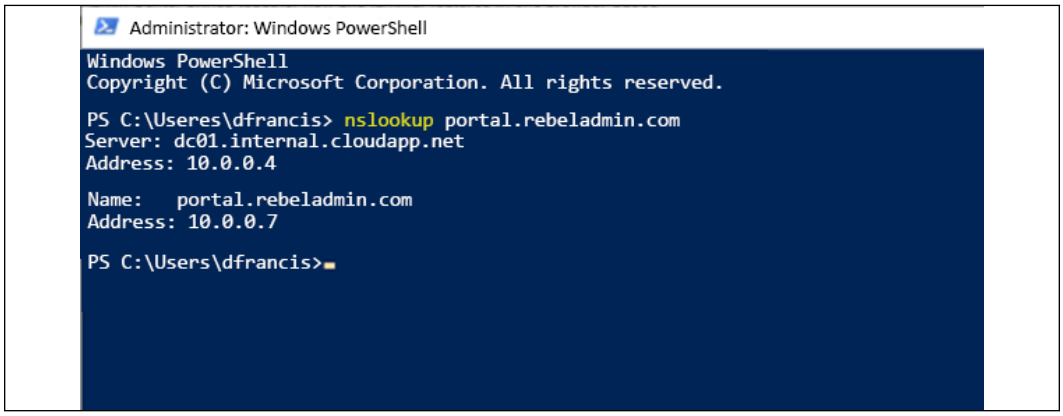


Figure 4.10: DNS query

As the next step, I am going to create a DNS policy to block DNS queries from `10.0.0.6`. Before I do that, first I need to create a client subnet. The client subnet represents the source of the query.

```
Add-DnsServerClientSubnet -Name "blockA" -IPv4Subnet 10.0.0.6/32
```

In the preceding command, I named the subnet "blockA." Next, let's go ahead and create a policy.

```
Add-DnsServerQueryResolutionPolicy -Name "blockApolicy" -Action IGNORE
-ClientSubnet  "EQ,blockA"
```

In the above, I am creating a policy called "blockApolicy" to block DNS queries from IP address `10.0.0.6`.

After the policy was in place, I went ahead and tried the `nslookup` query again from `10.0.0.6`:

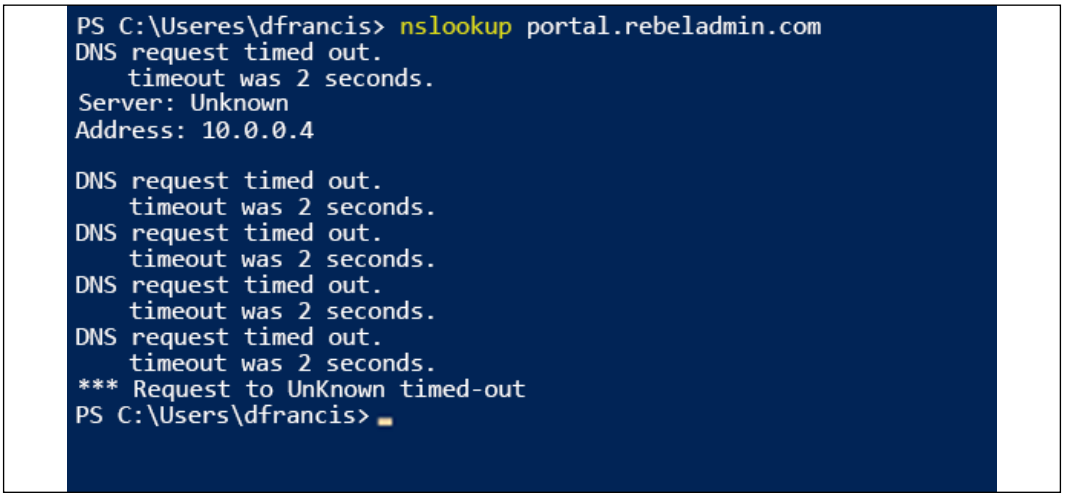


Figure 4.11: Failed DNS query

As we can see, `10.0.0.6` didn't receive any response.

As we can see DNS policies can be used to improve the quality of the DNS infrastructure and also it helps organizations to accomplish their complex name resolution requirements.

Secure DNS client over HTTPS (DoH)

DNS queries between DNS server and DNS client are normally in plain text format. But starting from Windows Server 2022, DNS queries can pass through secure HTTP (HTTPS) connections. This prevents someone from modifying/accessing DNS data in transit. Please note this setting is only for DNS clients. Windows DNS Server does not support DoH queries. Therefore, you should not enable DoH in domain-joined computers. If there is DNS traffic between a domain-joined computer and the AD domain server, we need to consider securing IPSec connections. At the moment, Google and Cloudflare DNS servers support DoH queries.

DNS server operation modes

There are three types of DNS server operation modes. These modes are not something we can choose during the setup process. They are listed based on their characteristics:

- **Dynamic**: AD DS directory-integrated DNS uses Dynamic DNS by default. Dynamic DNS allows hosts and users to register, update, and remove DNS records from DNS servers. Let's assume we have an AD environment with 200 computers. It uses **Dynamic Host Configuration Protocol** (**DHCP**) to maintain the IP assignment; so every three days, each device will renew its IP allocation. Some may have the same IP address, but some may receive a new one. But if the system uses static DNS every three days, administrators will need to update the DNS list to match IP allocations. Also, AD will not be able to find the devices to establish authentication or handle resource access requests. However, thanks to Dynamic DNS, this is no longer manual work, and it allows the environment to maintain up-to-date DNS information without user interaction.
- **Read/write**: This is applicable when DNS zones run without AD DS integration. For example, one of the Rebeladmin Corp. clients wants to host their own web server. Therefore, as a service provider, we need to provide a solution, which DNS design is part of. The client likes to keep the cost to a minimum, and since it's a testing environment, they aren't worried about high availability. For their web server DNS requirements, we can set up a standalone DNS server in the same web server and use it as an authoritative DNS server. Records there are not going to change often, so there is no need for Dynamic DNS. If records need to be updated, an authorized user can update them manually.
- **Read-only**: If the DNS server only keeps a read-only copy of a master zone, it operates in read-only mode. Some DNS servers keep only secondary zones for security, load balancing, or disaster recovery purposes. This can typically be seen in web server farms. Read-only DNS servers will check with master DNS servers for DNS updates periodically.

With Windows Server 2008, Microsoft introduced **read-only domain controllers** (**RODCs**). RODCs can be used in infrastructures where physical security and connectivity cannot be guaranteed. RODCs run AD DS-integrated primary DNS zones in read-only mode.

These operation modes can be used in infrastructures in order to meet their DNS requirements. It is possible to mix DNS servers with different operation modes. But it's important to clearly understand the capability of each operation mode for DNS troubleshooting.

Zone transfers

Healthy DNS replication is a key requirement for service and infrastructure integrity. In the previous section, I explained the different zones. I also mentioned how to set the zone transfer permissions. Now, it is time to look into DNS replications.

There are two types of zone file replications:

- **Asynchronous Full Transfer Zone** (**AXFR**): When setting up a new secondary zone, the system will replicate a full copy of the zone file from the master server. It is not just for the secondary zone; it's applicable to other zones, too. In the event of DNS replication issues, the administrator may need to request a full zone transfer (aka complete zone transfer) from its master server from time to time.
- **Incremental Zone Transfer** (**IXFR**): After the initial full zone transfer, the system will only replicate the records that have been modified. It reduces the replication traffic as well as providing faster replication.

When there is a change in the master DNS zone, the system will send a notification to secondary servers about the change. Then, the secondary servers will request a zone update. If secondary servers lose connection to the primary server, or after the service is restarted, the system will still query (based on SOA refresh intervals) from master DNS servers for zone updates.

DNS delegation

In the previous chapter, when we were looking at the AD DS design, I explained child domain controllers and how they can be used to organize the company's AD hierarchy. When you create a child domain in a forest, it also creates its own DNS namespace. In a DNS infrastructure, sometimes, it is required that you divide the DNS namespace to create additional zones for better management. DNS delegation allows organizations to achieve this without the need to change the domain structure.

In AD-integrated DNS, you are allowed to create any DNS zone, and it will be replicated to other domain controllers. But there are situations where this can lead to administrative overhead. Rebeladmin Corp. uses `rebeladmin.com` as its AD DS domain name. They have a software development department that develops web-based applications.

In order to test their applications, they have to use a web URL. Every time they need to test something, they open a support ticket and the IT team creates A records for it in the DNS server. Of late, these requests have become too frequent, and sometimes due to the workload, the IT team faces delays in processing these requests. This starts to affect software release deadlines. What can we do in order to overcome this situation and provide a better solution? If there was a way to allow the software development team to create A records when they are required, they would not need to wait for the IT team. But at the same time, it shouldn't be complex, as it will still create additional administration tasks for the IT team. With the help of the DNS delegation feature, we can deploy a DNS server and create a DNS zone called `dev.rebeladmin.com`. Then, we can allow the software development team to manage the DNS record under it. They can create A records for apps such as `app1.dev.rebeladmin.com` and `app2.dev.rebeladmin.com`. Also, all the users under `rebeladmin.com` will still be able to access these URLs without additional changes, as the `rebeladmin.com` DNS zone knows which DNS server has authority over the `dev.rebeladmin.com` DNS entries.

DNS delegation can also be used to divide the DNS workloads into additional zones to improve the performance and create a fault-tolerant setup.

Before we start the delegation process, we need a second DNS server with an active primary DNS zone.

In my example, I have a new DNS server called `REBEL-SDC01.rebeladmin.com`. I have installed the DNS role in it and created a primary DNS zone called `dev.rebeladmin.com`:

```
Add-DnsServerPrimaryZone -Name "dev.rebeladmin.com" -ZoneFile "dev.
rebeladmin.com.dns"
```

I have also created an A record in the zone called `app1`:

```
Add-DnsServerResourceRecordA -Name "app1" -ZoneName "dev.rebeladmin.
com" -AllowUpdateAny -IPv4Address "192.168.0.110"
```

To set up the DNS delegation, I log in to the `REBEL-PDC-01.rebeladmin.com` domain controller and run the following command:

```
Add-DnsServerZoneDelegation -Name "rebeladmin.com" -ChildZoneName "dev"
-NameServer "REBEL-SDC-01.rebeladmin.com" -IPAddress 192.168.0.110
```

In the preceding command, `ChildZoneName` defines the zone name in the other DNS server.

Now, when I go to DNS Manager and expand the tree, I can see the delegated zone entry for `dev.rebeladmin.com`:

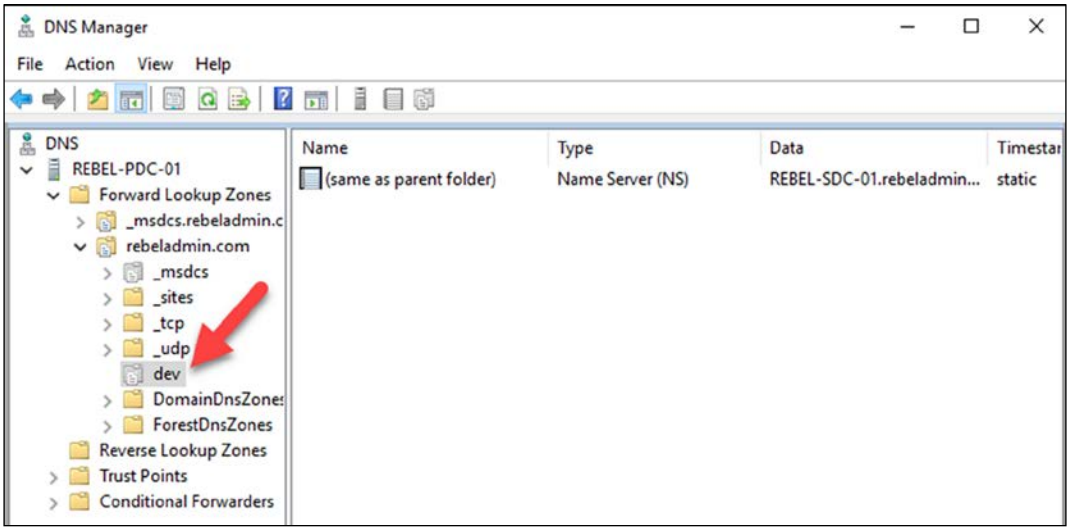


Figure 4.12: Delegated zone

When I ping `app1.dev.rebeladmin.com` from a PC in `rebeladmin.com`, I can successfully reach it:

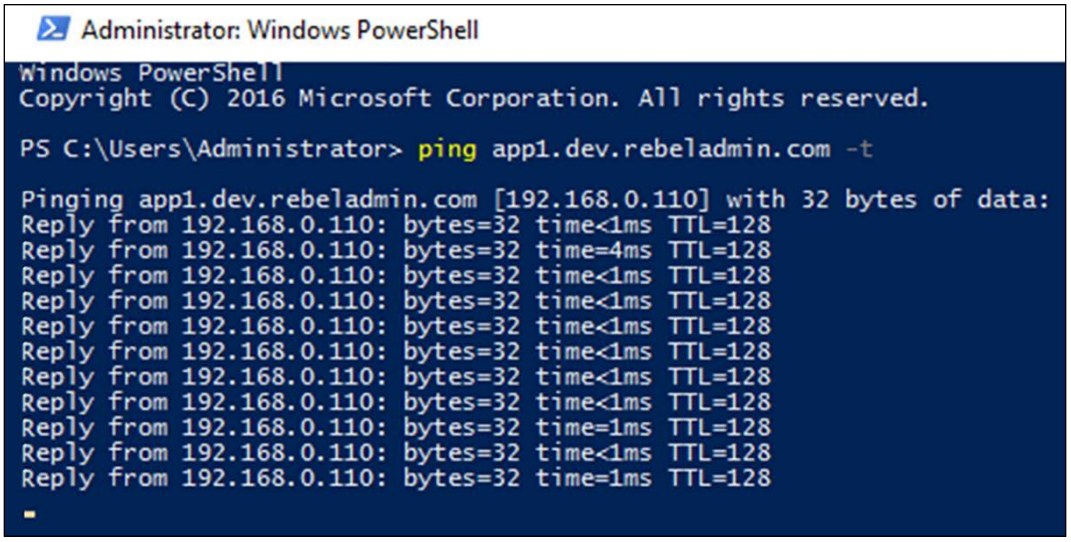


Figure 4.13: Resolve names from the delegated zone

DNS service providers

In this chapter, we mainly talked about how DNS works in AD environments. However, this is not the only way to use DNS. If we need to manage DNS for external URLs such as websites, we need to configure DNS servers in a different way. This type of DNS server mainly works with public IP addresses. In order to set up an external-facing DNS server, first we need to register NS records. Then, via a domain registrar, we need to point the domain DNS to NSes. After that, we can set up the required DNS records for the domain using our own DNS servers. As with any other server role, DNS also requires maintenance from time to time. Also, we need to maintain the high availability of the DNS servers. Instead of all this additional maintenance overhead, we can also use a DNS service provider to do the same thing. Azure DNS, GoDaddy, Dotster, 1&1, and DynDNS are some of the most well-known DNS service providers. All we need is a valid subscription and within minutes, we can start adding the relevant DNS records. This doesn't require firewall changes, servers, or any other resources. DNS record management is mainly done via a simple web interface. We also do not need to worry about backup, replication, or high availability as they are all handled by the service providers.

Summary

In an infrastructure, we can't talk about AD DS without mentioning DNS. In this chapter, we covered the basics of DNS and you learned why it's important. Then, we moved on to the hierarchical naming structure and saw how it helps translate an organization's logical structure to the domain namespace. We also learned exactly how DNS works behind the scenes. Then, we looked at DNS records and DNS zones. This also included some explanation of how to create different zones. We also learned about DNS policies, which were first introduced with Windows Server 2016. At the end of the chapter, you learned about different DNS operation modes and replications. I hope this information helps you understand the importance of DNS in an infrastructure and how to use it properly.

In the next chapter, we will look at the AD FSMO roles and explore how to place them correctly in an infrastructure.

5
Placing Operations Master Roles

In organizations, we use roles and responsibilities to maintain different levels of accountability. With roles and responsibilities in place, everyone knows what they are supposed to do in their job. This applies to applications/services as well. Within application/services we have different roles. These roles are also associated with different sets of privileges. While one role has read-only permissions, another has permission to do system-wide configuration changes. This helps to maintain the integrity of the application/services.

Active Directory is built upon a multi-master database model. This means that any writable domain controller in the domain can change the Active Directory configuration and it will replicate to all other domain controllers. But there are some operations that need to be controlled in a sensible manner in order to maintain the integrity of **Active Directory Domain Services** (**AD DS**). These operations are better managed in single-master mode rather than multi-master mode. These special roles are called **Flexible Single Master Operation** (**FSMO**) roles. These roles can run from one domain controller or be distributed among multiple domain controllers (according to the guidelines). But each role can appear only once in a domain or forest. This makes the operations master role holder important in an AD DS infrastructure. If the domain controller that holds the operations master role fails and can't recover, another domain controller will have to forcefully regain control over its operations master roles.

In this chapter, we are going to look into the following topics:

- FSMO roles and duties

- FSMO role placement
- Moving FSMO roles
- Seizing FSMO roles

Each FSMO role has its own part to play in the Active Directory environment. Before we look into FSMO role placement, let's go ahead and look into the responsibilities of each role.

FSMO roles

There are five FSMO roles in the Active Directory infrastructure. Each of the roles is responsible for performing specific Active Directory tasks that other domain controllers in the infrastructure are not permitted to perform. These five FSMO roles are divided into two categories based on their operation boundaries:

Forest level	**Domain level**
Schema operations master	The **primary domain controller** (**PDC**) emulator operations master
Domain-naming operations master	The **relative identifier** (**RID**) operations master
N/A	The infrastructure operations master

When we create the first Active Directory forest and the first Active Directory domain, all these FSMO roles will be installed in the domain's first domain controller (obviously; there's no other place). The majority of the "time," engineers leave the default configuration as it is, even though they keep adding domain controllers. Keeping FSMO roles in one domain controller is not a fault, but if you want to get the most out of it, role placement change is inevitable.

However, there are many different things that can have an impact on FSMO role placements, such as the size of the organization, the network topology, and infrastructure resources.

Schema operations master

This role boundary is the forest. This means that an Active Directory forest can have only one schema master. The owner of this role is the only domain controller in the forest who can update the Active Directory schema.

In order to make schema changes in the forest, it also needs to have a user account that is a member of the Schema Admins group. Once the schema changes are done from the schema master role owner, those changes will be replicated to other domain controllers in the forest.

In an Active Directory forest, the schema master role owner can be found using the following command:

```
Get-ADForest | select SchemaMaster
```

When you add a new version of Active Directory to the domain for the first time, it will need a schema modification. If you run the Active Directory configuration wizard with a user account that has Domain Admins permissions, it will fail. You need an account with Schema Admins privileges.

Domain-naming operations master

The domain-naming operations master role holder is responsible for adding and removing domain controllers to and from the Active Directory forest. In the Active Directory forest, the domain-naming operations master role owner can be found using the following command:

```
Get-ADForest | select DomainNamingMaster
```

When you add or remove a domain controller, the system will contact the domain-naming operations master role holder via the **Remote Procedure Call** (**RPC**) connection, and if it fails, it will not allow you to add or remove the domain controller from the forest. This is a forest-wide role, and only one domain-naming operations master role holder can exist in one forest.

PDC emulator operations master

The **PDC** operations master role is a domain-wide setting, which means each domain in the forest will have a PDC operations master role holder. One of the most common Active Directory-related interview questions is: *Which FSMO role is responsible for time synchronization?* The answer is *PDC!* In an Active Directory environment, it allows a maximum of a 5-minute time difference (time skew) between the server and client to maintain successful authentication. If it's more than 5 minutes, devices will not be able to be added to the domain, users will not be able to authenticate, and the Active Directory-integrated application will start throwing authentication-related errors.

It is important that domain controllers, computers, and servers in the Active Directory domain controller agree on one clock:

Figure 5.1: Time sources

Computers in a domain will sync their time with the domain controller they are authenticated with. Then, all of the domain controllers will sync their time with the **domain PDC** role holder. All the **domain PDC** role holders will sync the time with the forest **root domain PDC** role holder. Finally, the **root domain PDC** role holder will sync the time with an **external time source**.

The time accuracy of PDC is really important as the majority of resources in infrastructure will use PDC as the time source. We can use an external reliable time source for PDC to maintain the time accuracy. It is always recommended to use regulated sources such as NIST for the task. NIST has several stratum-1 network time servers (directly link to UTC) that we can use as time sources. The complete list of NIST time servers is available on `https://bit.ly/3HLOQsG`.

Also, before we configure the time source, there are certain TCP/UDP ports that need to be open in the edge firewall. More information about this is available on `https://bit.ly/3xf1ZFW`.

Once firewall rules are in place, we can configure the PDC by using the following steps:

1. Log in to PDC as a domain administrator.
2. Launch Command Prompt as an administrator.
3. Run `w32tm.exe /config /syncfromflags:manual /manualpeerlist:132.163.97.1,0x8 /reliable:yes /update`.

   In the preceding command, `132.163.97.1` is one of the NIST time servers; you can change this and choose any external source.
4. Then, to update the config, run `w32tm.exe /config /update`:

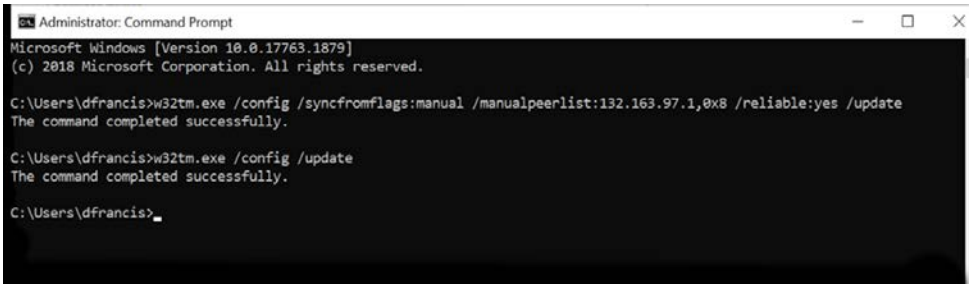


Figure 5.2: Configuring the external time source

Apart from time synchronization, the PDC role holder is also responsible for maintaining password change replications.

Also, in the event of authentication failures, PDC is responsible for locking down the account. All the passwords changed in other domain controllers will be reported back to the PDC role holder. If any authentication failure occurs in a domain controller before it passes the authentication failure message to the user, it will check the password saved in the PDC, as that will prevent errors due to password replication issues.

In the Active Directory domain, the PDC role owner can be found using the following command:

```
Get-ADDomain | select PDCEmulator
```

The PDC is also responsible for managing the **Group Policy Object** (**GPO**) edit. Every time a GPO is viewed or updated, it will be done from the copy stored in the PDC's `SYSVOL` folder. Due to the importance of the role, it is recommended to choose the most reliable domain controller to hold the PDC role.

RID operations master role

The **RID** master role is a domain-level role, and each domain in the forest can have RID role owners. It is responsible for maintaining a pool of RIDs that will be used when creating objects in the domain. Each and every object in a domain has a unique **security identifier** (**SID**). The RID value is used in the process of SID value creation. The SID is a unique value to represent an object in Active Directory. The RID is the incremental portion of the SID value. Once the RID value is being used to generate a SID, it will not be used again. Even after deleting an object from Active Directory, we will not able to reclaim the RID value back.

This ensures the uniqueness of the SID value. The RID role owner maintains a pool of RIDs. When the domain has multiple domain controllers, it will assign a block of 500 RID values for each domain controller. When pool usage is more than 50%, domain controllers will request another block of RIDs for the RID role owner.

In the Active Directory domain, the RID role owner can be found using the following command:

```
Get-ADDomain | select RIDMaster
```

In the event of a RID role owner failure, its impact will be almost unnoticeable until all domain controllers run out of allocated RID values. It will also not allow you to move objects between domains or add new domain controllers.

Infrastructure operations master

This role is also a domain-level role, and it is responsible for replicating SID and **distinguished name** (**DN**) value changes to cross-domains. SID and DN values get changed based on their location in the forest. If objects are moved between domains, their new values need to be updated in groups and **Active Control Lists** (**ACLs**). This is taken care of by the infrastructure operations master. This will ensure that the moved objects have access to their resources without interruptions.

In the Active Directory domain, the infrastructure operations master role owner can be found using the following command:

```
Get-ADDomain | select InfrastructureMaster
```

The infrastructure operations master role owner checks its database periodically for foreign group members (from other domains), and once it finds those objects, it checks its SID and DN values with the global catalog server. If the value in the global catalog is different from the local value, it will replace its value with the global catalog server's value. Then, it will replicate it to other domain controllers in the domain.

By design, the global catalog server holds a partial copy of every object in the forest. It does not have the need to keep a reference of cross-domain objects. If the infrastructure master is in place in a global catalog server, it will not know about any cross-domain objects. Therefore, the infrastructure operations master role owner should not be a global catalog server. However, this is not applicable when all the domain controllers are global catalogs in a domain because, that way, all the domain controllers will have up-to-date information about objects.

FSMO role placement

The first domain controller in the first Active Directory forest will hold all five FSMO roles. All these roles are critical in the Active Directory infrastructure. Some of them are heavily used and some roles are only used on specific occasions. Some role owners will not be able to afford any downtime, and some roles will still be able to have downtime. So, based on features, impact, and responsibilities, these can be placed on different domain controllers. However, FSMO role placement is heavily dependent on Active Directory design.

Active Directory's logical and physical topology

If it's a single forest-single domain environment, it is not wrong to keep all the FSMO roles in one domain controller. According to the best practices, the infrastructure operations master role should not be held by a global catalog server. But, this is only if the environment has multiple domains. In a single forest-single domain environment, on most occasions, all servers are global catalog servers as well.

In the following example, in the `rebeladmin.com` single forest-single domain environment, there are three domain controllers in the infrastructure:

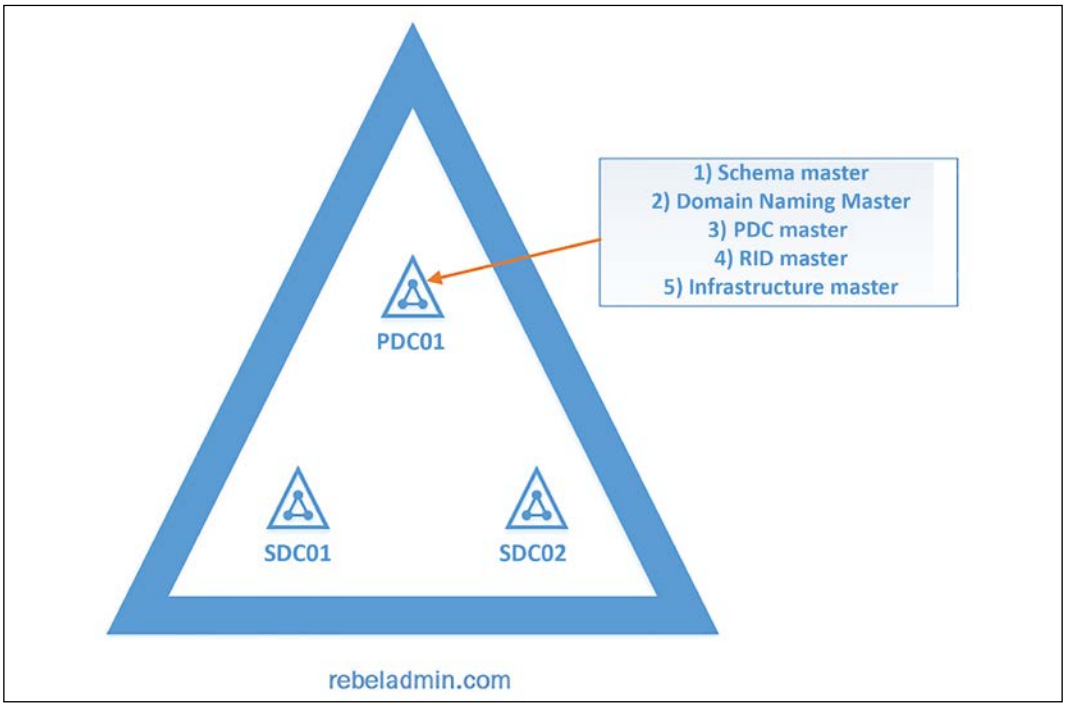


Figure 5.3: FSMO role placement example 1

PDC01 is recognized as the most powerful (capacity) and most reliable domain controller. Therefore, **PDC01** holds all five FSMO roles. **SDC01** and **SDC02** are also two global catalog servers. In this case, just moving the infrastructure operations master role to one of the domain controllers will not have a significant impact.

In the event of a **PDC01** failure, secondary domain controllers will be able to claim ownership of FSMO roles (this process is called **seizing FSMO roles** and will be described later in the chapter).

In a multiple-domain environment, this will change. Forest-wide roles and domain-wide roles will need to be placed properly in order to maintain high availability.

In the following example, Rebeladmin Corp. has three domains. The `rebeladmin.net` domain is the forest root domain. The `rebeladmin.com` domain is used at its headquarters in the USA and `rebeladmin.ca` is used at its Canadian branch:

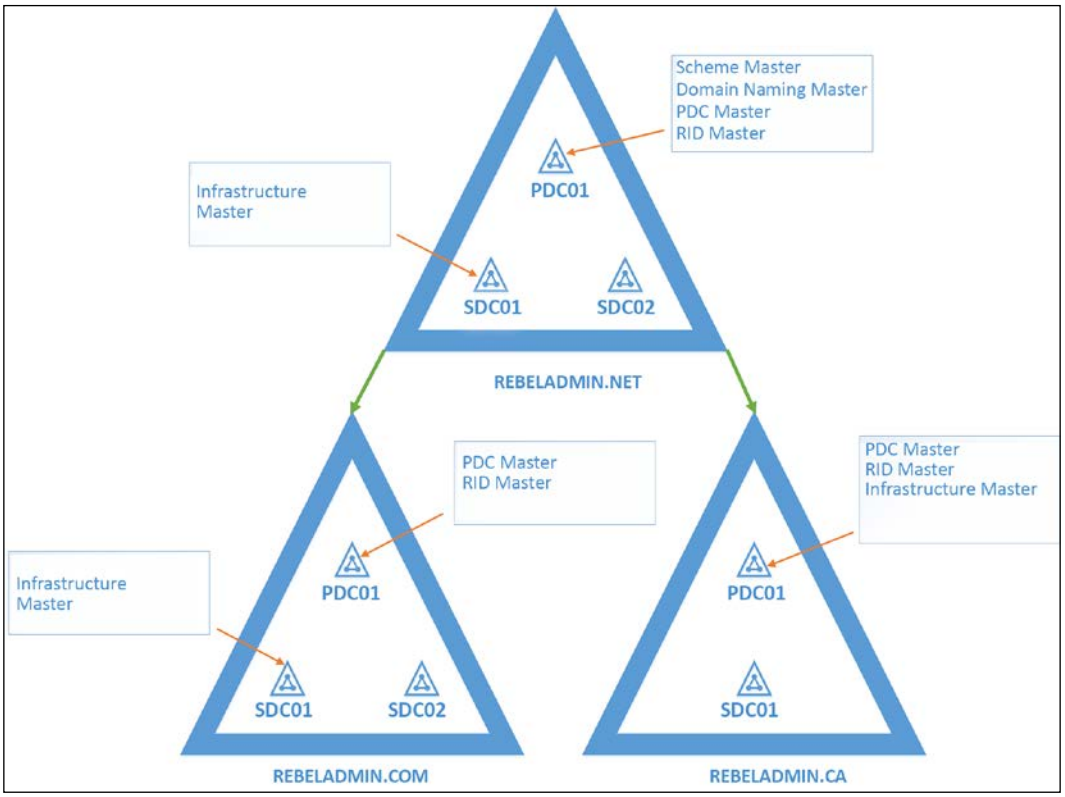


Figure 5.4: FSMO role placement example 2

All three domains share one forest. Therefore, forest-wide roles, which are the **Schema master** and the **Domain-naming master**, will be placed in the forest root domain, `rebeladmin.net`. It has three domain controllers. **PDC01** is identified as the most reliable domain controller.

The **Schema master** and the **Domain-naming master** roles use relatively lower amounts of processing power as forest-wide changes will not happen frequently. But high availability is a must, as other domain activities will depend on it.

PDC is the most consumed FSMO role, as it will replicate password changes, control time synchronization, and manage GPO edits. So, it's important for PDC to run on the domain controller that has the most processing power. Domain controllers depend on the **RID master role**. Therefore, reliable communication between domain controllers and the RID master role holder is crucial.

It is advised that you keep PDC and RID together in one domain controller in order to avoid network latency-related issues. So, in the end, we can place four FSMO roles in **PDC01**. In a multi-domain environment, cross-domain referencing is important. If the user account in the forest root domain changes its name, it will immediately be replicated to all the domain controllers in the forest root domain. But, if the user is part of a group in the other domains, they also need to know about these new values. Therefore, **SDC01** is made as a non-global catalog server and it holds the **Infrastructure master** role. **SDC02** is kept as a backup domain controller; if any of the FSMO role holders are dead, it can use it to claim role ownership. In some infrastructures, the forest root domain is not used for active operations. In such situations, keeping multiple domain controllers is management overhead.

The `rebeladmin.ca` domain is used in a regional office infrastructure. It has less than 25 users, and most of them are salespeople. Therefore, keeping multiple domain controllers cannot be justified based on capacity or reliability facts. The setup is limited to two domain controllers, and **PDC01** hosts all three domain-wide **FSMO** roles. **SDC01** is kept as a backup domain controller, to be used in a **Disaster Recovery** (**DR**) scenario.

Connectivity

Healthy replication between domain controllers is a must for the Active Directory infrastructure. FSMO role holders are designated to do specific tasks in the infrastructure. Other domain controllers, devices, and resources should have a reliable communication channel with FSMO role holders in order to get these specific tasks done when required.

In Active Directory infrastructures, there can be regional offices and remote sites that are connected using WAN links. Most of the time, these WAN links have limited bandwidth. These remote sites can host domain controllers, too. If replication traffic between sites is not handled in an optimized way, it can turn into a bottleneck. Rebeladmin Corp. is a managed services provider and it has two offices. The headquarters is located in Toronto and the operation center is based in Seattle, USA. It is connected via a 512 KB WAN link. In the Toronto office, there are 20 users and in the Seattle office, there are 500 users. It runs on a single-domain Active Directory infrastructure.

As I mentioned earlier, among all these FSMO roles, the PDC is the highly used FSMO role. Devices and users keep communicating with the PDC more frequently than other FSMO role holders. In this scenario, if we place the PDC in the Toronto office, 500 users and associated devices will need to go through the WAN link in order to communicate with the PDC. But if we place it in the Seattle site, then the traffic that will pass through the WAN link will be lower. In a regional office scenario, make sure you always place the PDC near the site that hosts the greatest number of users, devices, and resources.

The network topology design for inter-site connectivity also has an impact on the FSMO role placement. In the following example, the Active Directory setup has three Active Directory sites with a single domain infrastructure. **Site Canada** connects to **Site USA** and **Site USA** connects to **Site Europe**. But **Site Canada** does not have a direct connection with **Site Europe**:

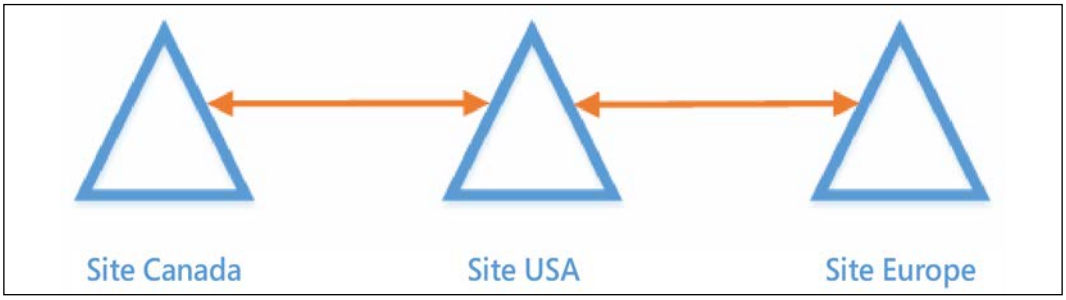


Figure 5.5: Inter-site connectivity

Now, if the FSMO roles are placed in **Site Canada**, **Site Europe** will have issues communicating with it. **Site Europe** will not be able to perform any FSMO-related tasks. According to the network topology, the best option will be to place the FSMO roles in **Site USA**, as both sites have a direct connection to it.

The number of domain controllers

The number of domain controllers that can be deployed on an Active Directory infrastructure depends on the budget and available resources (computer resources, space, power, and connectivity).

Based on the number of domain controllers, engineers will need to decide to either run all FSMO roles together or distribute them among domain controllers. It is recommended that you have at least two domain controllers on a site. One will be holding the FSMO roles, and the other one will be kept as a standby domain controller. In the event of a primary server failure, FSMO roles can be hosted on the standby domain controller.

Capacity

The capacity of the Active Directory domain controllers also affects the FSMO role placement. When the number of users, devices, and other resources increases, it also increases FSMO role-related activities. If it's a multi-site environment, it's recommended that you place FSMO roles on sites that run with a high capacity of Active Directory objects in order to lower the impact of replication and latency issues.

FSMO role holders are also involved in typical Active Directory tasks, such as user authentication. In large Active Directory environments (10,000+ objects), it is important to prioritize FSMO-related tasks over regular Active Directory tasks. This mainly impacts the PDC emulator, as it is the most active FSMO role. It is possible to prioritize domain controller operations by editing the *weight* value of the DNS SRV record. The default value is `100`, and reducing it will reduce the number of user authentication requests. The recommended value is `50`. The domain controller with the highest weight value has more authentication referrals than other domain controllers. As an example, let's assume we have two domain controllers in place and one has a weight of 100 and the other one has a weight of 150. The domain controller with the highest weight value (150) will have more authentication referrals.

This can be done by adding a new registry key under `HKLM\SYSTEM\CurrentControlSet\Services\Netlogon\Parameters`. The key should have a `DWORD` value with the entry name `LdapSrvWeight`.

Best practices

In previous sections, I have explained things we need to consider for FSMO role placement. Here, I'd like to summarize the points we discussed so far:

- Domain controllers in the forest should be able to reach FSMO role holders without any network layer connection barriers. If domain controllers are in a segmented network, make sure traffic is routed correctly.
- We can distribute FSMO roles to multiple servers; however, more servers means more management overhead. Unless it is a real requirement, try to keep FSMO roles to a fewer number of computers.
- Place the PDC role in the most reliable and powerful domain controller. Avoid installing applications and other Windows Server roles in PDC to reduce unnecessary resource usage.
- Keep the RID master and PDC roles in the same domain controller (same domain). Communication between these roles is crucial and keeping this in the same domain controller ensures reliable connectivity. Resource usage of the RID master role is small and it will not make a significant impact.

- Place the schema master role and domain-naming master role in the PDC of the forest root domain. In a mature Active Directory forest, we will rarely change the schema or add/remove domain controllers. But when it comes to that, it needs to be done in a controlled manner. When these critical changes happen in a domain, both of these role holders need to be available.
- If possible, keep the infrastructure master role in a non-global catalog server. This is because a global catalog server keeps a partial copy of every Active Directory object in the forest. The infrastructure master role will not update any object as it will not have any information about objects it doesn't hold. However, if the Active Directory recycling bin feature is enabled, every domain controller is responsible for updating cross-domain object references. In that case, it doesn't matter where the infrastructure role is placed.

Moving FSMO roles

In the Active Directory setup, sometimes, FSMO roles will need to be moved from one domain controller to another. Here, I have listed a few scenarios where it will be necessary to consider FSMO roles transfers:

- **Active Directory upgrades**: When there is the requirement for an Active Directory upgrade, first we need to introduce the new domain controllers to the existing setup and then move the FSMO roles. After that, the domain controllers that run older versions can be decommissioned.
- **Active Directory logical and physical topology**: When installing the first domain controller in the infrastructure, it will automatically hold all five FSMO roles. But, based on the Active Directory topology design, the roles can be transferred to ideal locations, as discussed in the previous section. This can be based on the initial design or an extended design.
- **Performance and reliability issues**: FSMO role owners are responsible for specific tasks in an Active Directory infrastructure. Each role can appear in only one domain controller in a domain. Some of these roles are focused on more processing power, and other roles are more concerned about the uptime. Therefore, in general, FSMO roles should be running on the most reliable domain controllers in the infrastructure. If the allocated resources are not enough for the FSMO role operations or if the servers have reliability issues, it will be necessary to move on to another host. This happens mainly when these roles are running on physical servers. If it's a virtual server, it will just be a matter of increasing the allocated resources. Some businesses also have infrastructure refreshment plans, which will kick off every 3 or 5 years. In such situations, the FSMO roles will need to move into the new hardware.

Let's look at how we can transfer the FSMO roles.

Before we start, we need to check the current FSMO role holder. This can be done by running the following command:

```
netdom query fsmo
```

In the infrastructure, there is a new domain controller added with the name `REBEL-SDC02`, and I'd like to move the domain-wide FSMO roles, which are the PDC, RID, and infrastructure roles, to the new server:

```
Move-ADDirectoryServerOperationMasterRole -Identity REBEL-SDC02
-OperationMasterRole PDCEmulator, RIDMaster, InfrastructureMaster
```

FSMO role transfer commands need to be run with the required privileges. If you need to move domain-wide roles, the minimum you need to have are Domain Admins privileges. If they are forest-wide roles, they need to be Enterprise Admins privileges. To move the schema master role, Schema Admins privileges are the minimum requirement.

Once the move is completed, we can check the role owners again:

```
PS C:\Users\dfrancis.REBELADMIN> netdom query fsmo
Schema master               REBEL-PDC-01.rebeladmin.com
Domain naming master        REBEL-PDC-01.rebeladmin.com
PDC                         REBEL-SDC02.rebeladmin.com
RID pool manager            REBEL-SDC02.rebeladmin.com
Infrastructure master       REBEL-SDC02.rebeladmin.com
The command completed successfully.

PS C:\Users\dfrancis.REBELADMIN>
```

Figure 5.6: Migrating the selected number of FSMO roles

If we need to move all five FSMO roles to a new host, we can use the following command:

```
Move-ADDirectoryServerOperationMasterRole -Identity REBEL-SDC02
-OperationMasterRole SchemaMaster, DomainNamingMaster, PDCEmulator,
RIDMaster, InfrastructureMaster
```

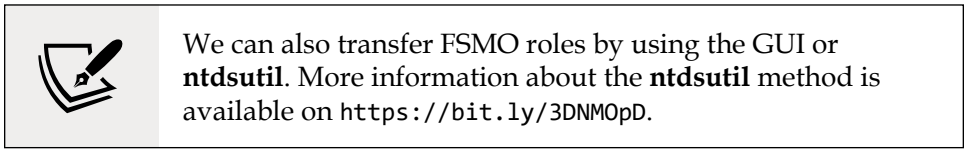
The following screenshot shows the output for the `netdom query fsmo` command:

```
PS C:\Users\dfrancis.REBELADMIN> netdom query fsmo
Schema master               REBEL-SDC02.rebeladmin.com
Domain naming master        REBEL-SDC02.rebeladmin.com
PDC                         REBEL-SDC02.rebeladmin.com
RID pool manager            REBEL-SDC02.rebeladmin.com
Infrastructure master       REBEL-SDC02.rebeladmin.com
The command completed successfully.
```

Figure 5.7: Migrating FSMO roles

If we need to move a single FSMO role, the `Move-ADDirectoryServerOperationMaster Role` command can be used with the individual role.

Once the transfer is completed, the system will create an event in the event viewer under the directory service log with the event ID `1458`.



We can also transfer FSMO roles by using the GUI or **ntdsutil**. More information about the **ntdsutil** method is available on `https://bit.ly/3DNMOpD`.

Seizing FSMO roles

In the previous section, I explained how to transfer FSMO roles from one domain controller to another. But there are certain situations where we will not be able to transfer the FSMO roles, such as the following:

- **Hardware failures**: If the domain controller that holds the FSMO roles failed due to hardware issues and there is no other way to bring it back online using a backup/DR solution, we need to use the seize method to recover FSMO roles
- **System operation issues**: If the domain controller has issues, such as operating system corruption, viruses, malware, or file corruption, it may not be allowed to transfer the FSMO role to another domain controller, which will also lead to an FSMO role seize
- **Forcefully removed domain controller**: If the FSMO role holder is forcefully decommissioned using the `/forceremoval` command, we need to use the seize method from any other available domain controller to recover FSMO roles

The FSMO role seize process should be used only in a disaster where you cannot recover the FSMO role holder. Some of the FSMO roles (RID, domain-naming master, and schema master) can still afford a few hours of downtime with minimum business impact. Therefore, we do not use the seize option as the first option if the FSMO role holder can still be recovered or fixed.

Once the seize process is completed, the old FSMO role holder should not be brought online again. It is recommended that you format and remove it from the network. At any given time, it is not possible to have the same FSMO role appear in two servers on the same domain.

In the following example, there are two domain controllers in the infrastructure. `REBEL-SDC02` is the FSMO role holder and `REBEL-PDC-01` is the additional domain controller. Due to hardware failure, I cannot bring `REBEL-SDC02` online and I need to seize the FSMO roles:

```
PS C:\Users\dfrancis>
PS C:\Users\dfrancis>
PS C:\Users\dfrancis> netdom query fsmo
Schema master               REBEL-SDC02.rebeladmin.com
Domain naming master        REBEL-SDC02.rebeladmin.com
PDC                         REBEL-SDC02.rebeladmin.com
RID pool manager            REBEL-SDC02.rebeladmin.com
Infrastructure master       REBEL-SDC02.rebeladmin.com
The command completed successfully.

PS C:\Users\dfrancis> ping REBEL-SDC02 -t

Pinging rebel-sdc02.rebeladmin.com [10.2.0.5] with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 10.2.0.5:
    Packets: Sent = 5, Received = 0, Lost = 5 (100% loss),
Control-C
```

Figure 5.8: Domain controller ping test

In order to seize the roles, the following command can be used:

```
Move-ADDirectoryServerOperationMasterRole -Identity REBEL-PDC-01
-OperationMasterRole SchemaMaster, DomainNamingMaster, PDCEmulator,
RIDMaster, InfrastructureMaster -Force
```

This command will take a few minutes to complete in the background, as it will try to connect to the original FSMO role holder first.

The only change in the command from the FSMO role transfer is the `-Force` parameter at the end. Otherwise, it's the exact same command. You also can seize the individual role using `Move-ADDirectoryServerOperationMasterRole -Identity REBEL-PDC-01 -OperationMasterRole <FSMO Role> -Force`.

`<FSMO Role>` can be replaced with the actual FSMO role value.

Once the command is completed, we can check the status of the new FSMO role holder:

```
PS C:\Program Files\PowerShell\7> netdom query fsmo
Schema master               REBEL-PDC-01.rebeladmin.com
Domain naming master        REBEL-PDC-01.rebeladmin.com
PDC                         REBEL-PDC-01.rebeladmin.com
RID pool manager            REBEL-PDC-01.rebeladmin.com
Infrastructure master       REBEL-PDC-01.rebeladmin.com
The command completed successfully.

PS C:\Program Files\PowerShell\7>
```

Figure 5.9: Seizing FSMO roles

As we can see, `REBEL-PDC-01` becomes the new FSMO role holder.

Summary

This is the end of another Active Directory infrastructure design chapter that was focused on FSMO role placements. FSMO roles are designated to do specific tasks in an Active Directory infrastructure in order to maintain integrity. In this chapter, you learned about FSMO roles and their responsibilities. Then, we moved on to FSMO role placement in the infrastructure, where you learned about techniques and best practices that need to be followed in order to maintain the best performance and availability. After that, we looked at how to transfer the FSMO roles from one domain controller to another using PowerShell, followed by a guide for seizing FSMO roles in the event of a disaster where you cannot recover the original FSMO role holder.

In the next chapter, we will look at actual Active Directory deployment scenarios and explore how to migrate from older versions of Active Directory to AD DS 2022.

6

Migrating to Active Directory 2022

In previous chapters, we looked at **Active Directory** (**AD**) components and learned how to design an AD infrastructure using them. Now, it's time to look at installing **Active Directory Domain Services** (**AD DS**). It would be perfect if we could design and implement an AD infrastructure from scratch, but in reality, the majority of organizations already have an AD infrastructure. Therefore, most of the time, as engineers, we will be looking into AD migrations rather than completely new designs. Apart from migrations, we may also have to work on extending the current AD design to meet new business requirements (for example, creating a new domain, introducing a new AD site, Azure AD integration, and so on) or to correct existing design issues (for example, changing the domain name, dealing with mergers and acquisitions). In all of these scenarios, we may have to add new domain controllers and these installation steps are pretty much the same. Apart from different AD DS deployment scenarios, we will also cover the following topics in this chapter:

- AD DS installation prerequisites
- AD DS health check
- How to plan an AD migration
- How to migrate to AD DS 2022 from Windows Server 2008 R2
- How to confirm a successful installation and migration

AD DS role installation is a very straightforward process. But there are certain important prerequisites we need to consider before proceeding with AD DS role installation and configuration.

AD DS installation prerequisites

Before we look at installing AD DS, there are certain prerequisites that need to be fulfilled. Without these, even if we have a good design, we will not have a healthy AD DS environment.

Hardware requirements

In modern infrastructures, most workloads run on virtualized platforms. Some still think it is best to keep at least one physical domain controller in AD infrastructure but this is not true. In the early days of virtualization, I would somewhat agree but now technology has moved on. We can keep all domain controllers as virtualized domain controllers. However, if required, the following are the minimum hardware requirements for AD DS 2022:

- 1.4 GHz 64-bit processor
- 2 GB RAM
- A storage adapter that supports the PCI Express architecture (Windows Server 2022 does not support IDE/ATA/PATA/EIDE for boot, data, or page drives)
- 32 GB of free space
- 1 x network adapter
- DVD drive or support for a network USB boot

Virtualized environment requirements

Today, organizations have several options to choose from when it comes to virtualization. They can either build their own private cloud using solutions such as Hyper-V or VMware or use public cloud providers such as Microsoft Azure or AWS. If required, they also can maintain a "hybrid cloud" or "multi-cloud" setup where they can use the best of both (private and public cloud). AD DS 2022 supports both of the above scenarios.

If it's a private cloud, it will allow you to have more control over resource allocation, but in the public cloud, this will depend on the subscription and the amount the organization is capable of spending.

The minimum requirements for a virtualized environment are as follows:

- 1.4 GHz 64-bit processor
- 2 GB RAM
- Virtual SCSI storage controller
- 32 GB of free space
- 1 x virtual network adapter

Depending on your virtualization service provider, there will be specific guidelines for virtualized domain controllers. Always follow these recommendations. This applies to the public cloud as well. Let's go ahead and see what we need to consider when deploying a domain controller in Azure.

Best practices for installing a domain controller in Microsoft Azure

Here are some of the considerations we need to make when we install a domain controller:

1. We can install an AD DS role in a VM running in Azure. It is a fully supported deployment. However, there are certain things we need to consider before we deploy virtualized domain controllers in Azure. If you are using Azure Firewall or a native **Network Security Group** (**NSG**) make sure you have allowed relevant ports that are used by AD DS. The port requirements are available in the following document: `https://bit.ly/3HIU60i`.
2. You can select any VM size that suits your operation and budget requirements.
3. Use a separate data disk for an NTDS database, SYSVOL folders, and log files.
4. Set **Host Cache Preference** of the data disk to **None** to disable write-through caching, which can cause a conflict with AD DS operations.

5. Assign a static private IP address for the domain controller. The static IP configuration needs to be done at the VM configuration level and not at the operating system level. Once installation is completed, update the virtual network DNS settings with the domain controller's IP address. This will allow you to add VMs to the domain that use the same virtual network:

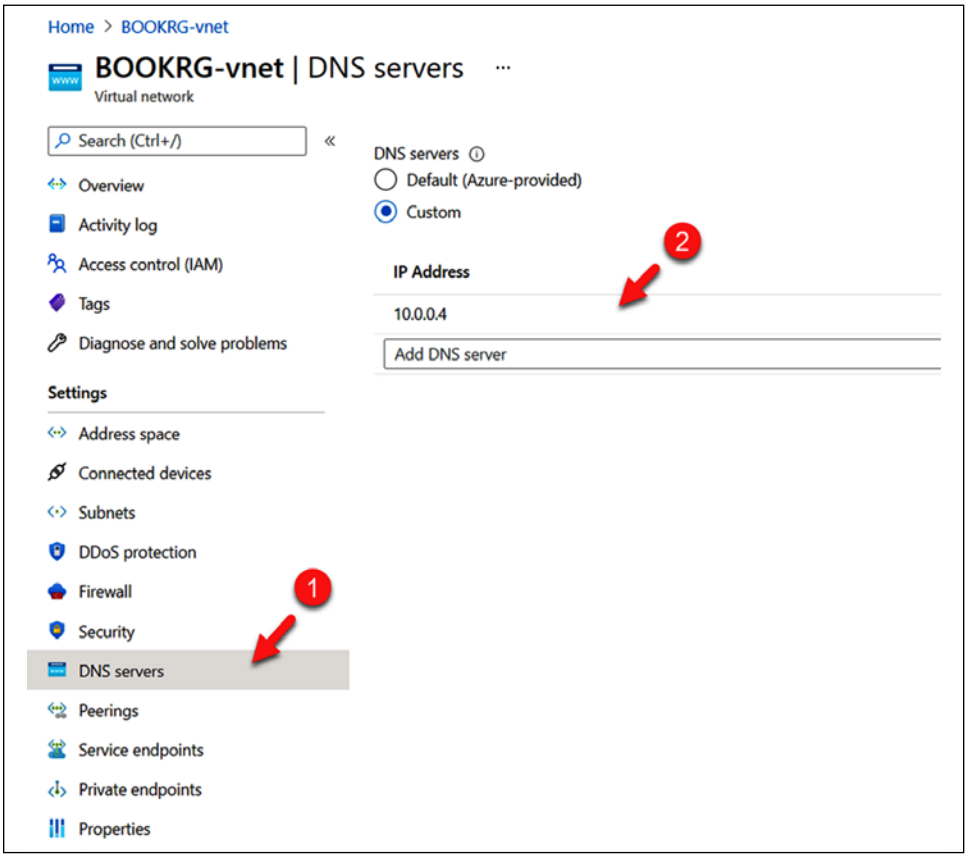


Figure 6.1: Azure Virtual Network DNS servers settings

6. DO NOT assign public IP addresses to a domain controller as it is a security risk.
7. Azure Availability Zones offer high availability for data and applications. In an Azure region, there can be one or more data centers. An Azure Availability Zone is made out of one or more data centers in the same Azure region that have independent power, hardware, networking, and cooling. All zone-redundant services will replicate data and applications across Availability Zones for high resilience. Each Azure region contains a minimum of three Azure Availability Zones.

8. Azure also has availability sets, which you can use to replicate a VM to different hardware in the same Availability Zones. Both of these solutions provide high availability to Azure VMs. It is recommended to run at least two domain controllers and add them to an Availability Zone or availability set.
9. An AD site can be a physical location or a separate network. AD sites help computers to find the closest domain controller. More information about this can be found in *Chapter 11, Active Directory Services - Part 01*. It is recommended to create a separate AD site to represent Azure domain controllers and subnets.
10. Microsoft also recommends not placing **Flexible Single Master Operation** (**FSMO**) roles in Azure-based domain controllers. However, I couldn't find any strong reason for this recommendation unless you are in a hybrid environment and the majority of the workloads are still running On-prem. Also, if the link between On-prem and Azure is not stable, it is also a good reason not to transfer FSMO roles to Azure-based domain controllers.
11. Do not shut down a domain controller running in Azure by using the Azure portal. Always shut down/restart at the guest operating system level. This will prevent issues such as **Relative ID** (**RID**) pool exhaustion.

Additional requirements

Apart from the physical or virtual resource requirements, there are a few other things to consider before installing a domain controller:

- **Operating system version and installation mode**: Windows Server 2022 has standard and data center versions. AD DS roles are available under both versions, but it is important to decide on and arrange the required licenses in advance. Windows Server 2022 supports two installation modes. A server with desktop experience is the standard installation method. Server roles and operations can also be managed by using GUIs or commands. The Server Core method also supports AD DS. Server Core doesn't have a GUI, and it reduces the operating system footprint. It also reduces the attack surface of the identity infrastructure.
- **Design document**: Documentation is crucial in any system implementation. Before starting the installation process, produce a document that includes the AD physical and logical topology, risks, features, and so on.

It is recommended to get the documentation approved by authorized people before deployment. This helps everyone agree on one design and refer to it when required. It also creates a starting point for AD infrastructure changes.

- **Domain and forest names**: During the AD DS installation, we need to specify the domain name and the forest name. In an organization, it's important to agree about these names with management before starting the installation process. This information can be added to your AD design document and submitted for approval. Back in 2015, I wrote an article on my blog about the AD domain rename process. Surprisingly it is still one of the top posts on my blog. The most common reasons for domain name changes are to correct some legacy naming errors or due to M&A activities. Either way, domain renaming should be avoided whenever possible. If domain renaming is required, the recommendation is to use the migration method. Some organizations use `.local` (**non-routable domain**) for their domain names. If we extend the on-prem AD infrastructure with a non-routable domain to Azure AD, we have to add additional **User Principle Names** (**UPNs**) with a routable domain name and force users to use it. Therefore, it's good practice to use routable domain names in the first place.
- **Dedicated IP address**: Domain controllers should use static IP addresses. Before installation begins, assign static IP addresses to domain controllers and test their connectivity. AD domain controller IP addresses can be changed later if required, but it is recommended to avoid that as much as possible.
- **Monitoring**: Once AD DS is installed, we need to monitor system performance, replication health, and the integrity component and services to identify potential service impacts and bottlenecks. Microsoft **System Center Operations Manager** (**SCOM**) and **Azure Sentinel** are the recommended monitoring tools since they include modules that have been specially designed to identify both service-level and security-level issues.
- **Backup/disaster recovery**: The high availability of AD infrastructure is a must for organizational operations as many services, applications, and other business components depend on it. Therefore, we need to plan how to keep the AD infrastructure functioning in a disaster, with minimal operations impact. There are different technologies and services that can be used to back up AD domain controllers, and some of these will be evaluated in *Chapter 11*, *Active Directory Services – Part 01*. After deciding on the solution, we also need to plan for periodic disaster recovery tests in order to verify the solution's validity. The general practice is to run at least two domain controllers in one physical site to maintain high availability. It is very rare that someone needs to restore a complete domain controller from a backup.

- **Virus protection in domain controllers**: As with any other system, domain controllers can also get infected by malicious code. There is debate about whether an AD domain controller should have antivirus software installed or not, but in the Microsoft documentation, I have never found anything saying it shouldn't have antivirus software. Always refer to your antivirus solution provider and check whether the solution supports protecting AD domain controllers. Depend on the vendor, we may have to exclude certain files and directories from scanning.

AD DS installation methods

There are two methods we can use to install AD domain controllers:

- **Using the Windows GUI**: After Microsoft introduced Server Manager with Windows Server 2008, the installation process of AD DS was simplified. In order to install AD DS using the Windows GUI, we need to install the AD DS role using Server Manager. Once this has been completed, we can run the AD DS configuration wizard:

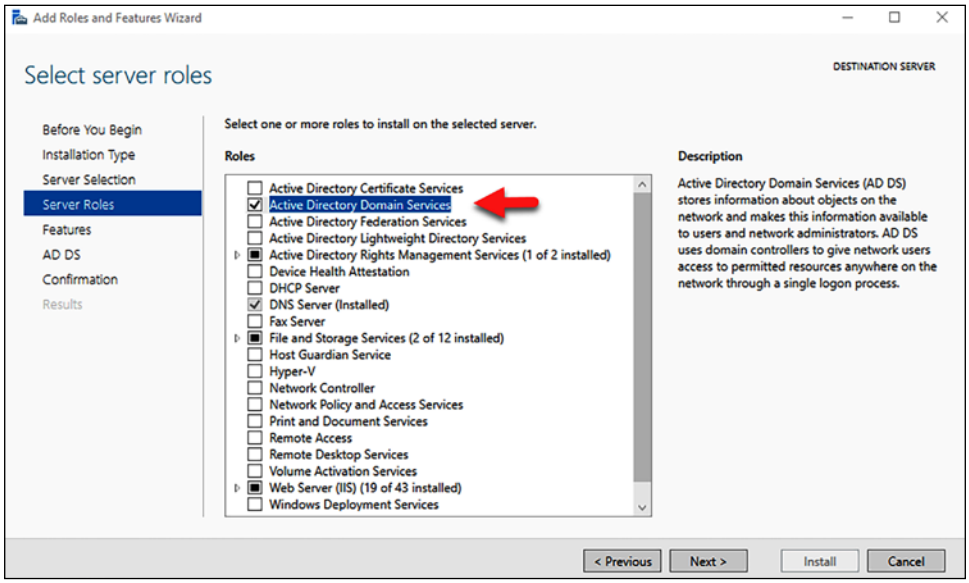


Figure 6.2: AD DS server role

The following screenshot shows the AD DS configuration wizard:

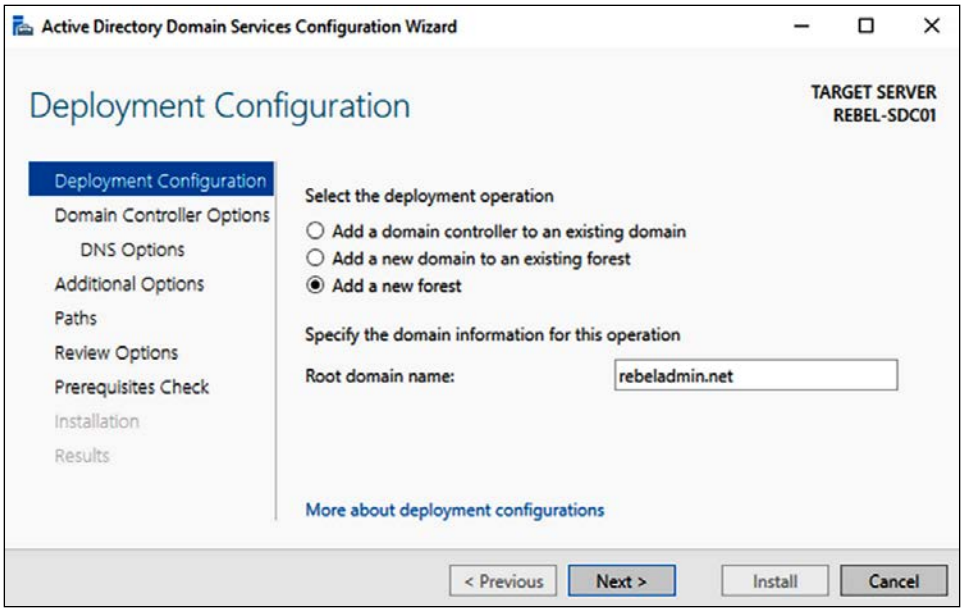


Figure 6.3: New AD forest root domain name

- **Using PowerShell**: Before Windows Server 2012, AD DS could be configured using *DCPromo unattended* files. The DCPromo tool was used to configure AD DS, and, using a text file, it was possible to pass the configuration values that were required. It removed user interaction for the AD DS configuration. With Windows Server 2012, DCPromo was replaced with PowerShell. Now, we can use a PowerShell script to install and configure AD DS. In this chapter, we will be using PowerShell for deployments.

AD DS deployment scenarios

In this section, we are going to look into different installation scenarios for AD DS.

Setting up a new forest root domain

For the first scenario, I am going to demonstrate how to set up a new AD forest. This will be the first domain controller of a new AD infrastructure. You can use the following checklist to make sure you have done your homework before clicking on the installation button.

AD DS installation checklist for the first domain controller

The following checklist can be used for a fresh AD DS installation:

1. Produce an AD design document
2. Prepare the physical/virtual resources for the domain controller
3. Install Windows Server 2022 Standard/Datacenter
4. Patch your servers with the latest Windows updates
5. Assign a dedicated IP address to the domain controller
6. Install an AD DS role
7. Configure AD DS according to the design
8. Review the logs to verify the health of the AD DS installation and configuration
9. Configure service and performance monitoring
10. Configure AD DS backup/disaster recovery
11. Produce system documentation

The preceding checklist covers the major things we need to consider during fresh AD DS deployment. However, based on business requirements and policies, you may have to add additional steps to the list.

Design topology

As we can see from the following diagram, in this scenario, `rebeladmin.com` will be the forest root domain:

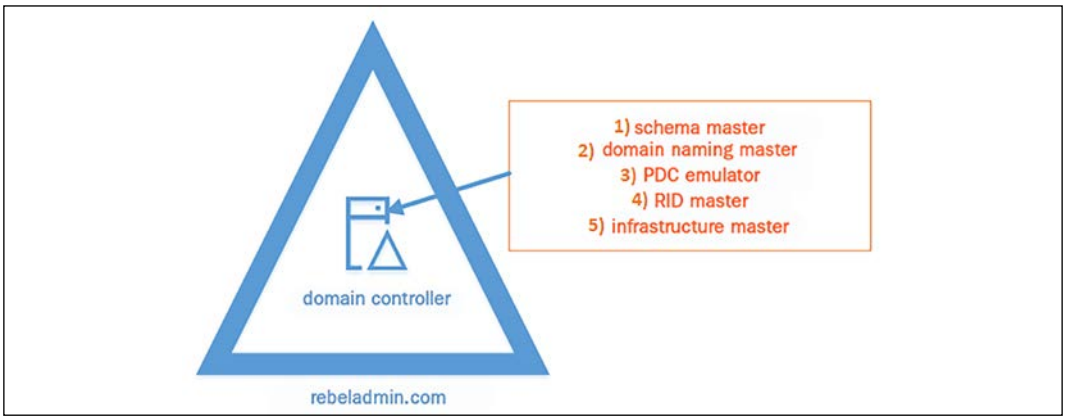


Figure 6.4: Design topology – fresh AD DS installation

The first domain controller that's installed on the forest will hold all five FSMO roles. In the previous chapter, we learned about FSMO role placement, and how, once additional domain controllers are added to the domain, these roles can be migrated to the best location.

Installation steps

Here, I will be demonstrating how to install the first domain controller in the forest. These demonstration steps are based on Windows Server 2022:

1. Log in to the server as a member of the local administrators group.
2. From here, verify the static IP address allocation by using `ipconfig /all`.
3. Launch the PowerShell console as an administrator.
4. Before the configuration process, we need to install the AD DS role in the given server. In order to do that, we can use the following command:

```
Install-WindowsFeature -Name AD-Domain-Services
 -IncludeManagementTools
```

   We don't need to reboot to complete the role service installations.

5. Now that we have the AD DS role installed, the next step is to proceed with the configuration:

```
Install-ADDSForest
 -DomainName "rebeladmin.com"
 -CreateDnsDelegation:$false
 -DatabasePath "C:\Windows\NTDS"
 -DomainMode "7"
 -DomainNetbiosName "REBELADMIN"
 -ForestMode "7"
 -InstallDns:$true
 -LogPath "C:\Windows\NTDS"
 -NoRebootOnCompletion:$True
 -SysvolPath "C:\Windows\SYSVOL"
 -Force:$true
```

There are no line breaks for the preceding command; I have listed it like this to allow you to see the parameters clearly. In the preceding command, the `-DomainName` and `-DomainNetbiosName` values can be replaced with the domain and NetBIOS names for your environment.

6. The following table explains the PowerShell commands and what they do:

Cmdlet	**Description**
`Install-WindowsFeature`	This cmdlet allows us to install Windows roles, role services, or Windows features in a local server or remote server. It is similar to using Windows Server Manager to install them.
`Install-ADDSForest`	This cmdlet allows us to set up a new AD forest.

7. The following table explains the arguments for the commands and what they do:

Argument	**Description**
`-IncludeManagementTools`	This installs the management tools for the selected role service.
`-DomainName`	This parameter defines the FQDN for the AD domain.
`-CreateDnsDelegation`	Using this parameter, we can define whether we will create a DNS delegation that references AD's integrated DNS.
`-DatabasePath`	This parameter defines the folder path for storing the AD database file (`ntds.dit`).
`-DomainMode`	This parameter will specify the AD domain's functional level. In the previous example, I used mode `7`, which is Windows Server 2016.
`-DomainNetbiosName`	This defines the NetBIOS name for the forest root domain.
`-ForestMode`	This parameter will specify the AD forest's functional level. In the previous example, I used mode `7`, which is Windows Server 2016.
`-InstallDns`	Using this, you can specify whether a DNS role needs to be installed with the AD domain controller. For a new forest, it is required that you set it to `$true`.
`-LogPath`	A log path can be used to specify the location that you save domain log files to.
`-SysvolPath`	This is used to define the `SYSVOL` folder path. The default location for it will be `C:\Windows`.
`-NoRebootOnCompletion`	By default, the system restarts the server after domain controller configuration. Using this command can prevent an automatic system restart.
`-Force`	This parameter will force a command to execute by ignoring the given warning. It is typical for the system to pass warnings about best practices and recommendations.

8. Once executed, the command will prompt you for the **SafeModeAdministratorPassword**. This is used in **Directory Services Restore Mode** (**DSRM**). Make sure that you use a complex password (according to Windows' password complexity recommendations). Failure to do so will stop the configuration.
9. Once the configuration is complete, reboot the domain controller and log back in as the Domain Admin.
10. Let's do a further check to confirm the successful installation of the services:

```
Get-Service adws,kdc,netlogon,dns
```

The preceding command will list the status of the AD-related services running on the domain controller:

Figure 6.5: Status of AD-related services

11. The following command will list all the configuration details of the domain controller:

```
Get-ADDomainController
```

The following screenshot shows the output for the preceding command:

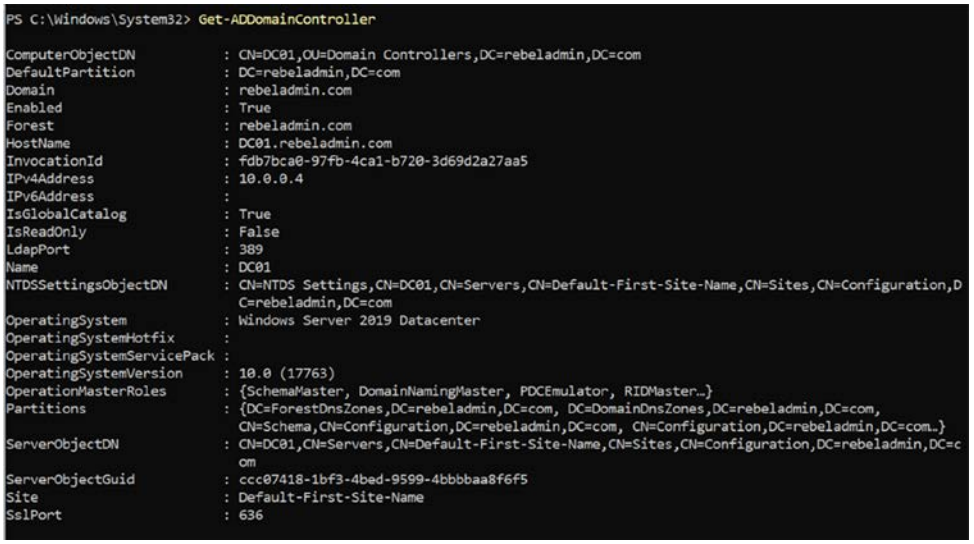


Figure 6.6: AD domain controller details

12. The following command will list the details of the AD domain:

```
Get-ADDomain rebeladmin.com
```

13. In the same way, `Get-ADForest rebeladmin.com` will list the AD forest details.
14. The following command will show whether the domain controller shares the `SYSVOL` folder:

```
Get-smbshare SYSVOL
```

As we can see, AD DS components are installed and configured successfully. Now that we have a new root domain installed, we will extend this further and set up an additional domain controller.

Setting up an additional domain controller

In this scenario, we are going to look into installing an additional domain controller in an existing AD domain. Before we look into this, there are a few prerequisites that need to be looked at:

- **Existing setup**: Before we promote the physical server or VM that's going to be used as the additional domain controller, it needs to be added to the existing AD domain. If the additional domain controller is going to be a DNS server, set its own IP address as the primary DNS server in the NIC settings and the existing domain controller IP address as the secondary DNS server. During the installation process, it needs to have connectivity to existing domain controllers (via LAN or WAN) in order to replicate the AD data.

By default, during the installation process, the system will try to replicate the AD partitions from any available domain controller. If this is via a slow WAN link, it is going to affect the installation process. Therefore, in such a scenario, AD can be installed using installation media. This is similar to an AD backup from an existing domain controller. Then, the system will use local media to replicate the initial AD data. This will be explained in detail in *Chapter 11, Active Directory Services – Part 01*.

  In order to add an additional domain controller, we need to know about certain information regarding the existing AD infrastructure, such as the following:
    - The AD domain name
    - The AD site
    - Whether the additional domain controller needs to be a global catalog server, DNS server, or **read-only domain controller** (**RODC**)
    - The initial AD data sync source (a specific AD domain controller or media installation)
- **Schema preparation and domain preparation**: Let's assume that we have an AD infrastructure based on Windows Server 2012 R2. Here, we need to add a domain controller, but we need it to use Windows Server 2022. Each version of AD has a different schema. By default, the AD DS 2012 R2 schema will not recognize the domain controller running on Windows Server 2022.

- Before the actual configuration begins, the existing schema needs to be modified to support this new requirement. This is done by using the `adprep.exe` file, which comes with the *operating system source files*. Before Windows Server 2012, this file had to be copied to the schema master, and we needed to run `/domainprep` and `/forestprep` to prepare the domain and forest. However, at the time of writing, it comes as part of the AD DS configuration, and it will run these commands in the background. In order to do that, the configuration process of the domain should be run as a Schema Admin or Enterprise Admin.

AD DS installation checklist for an additional domain controller

The following checklist can be used if you wish to install an additional domain controller:

1. Prepare the physical/virtual resources for the domain controller
2. Install Windows Server 2022 Standard/Datacenter
3. Patch the servers with the latest Windows updates
4. Assign a dedicated IP address to the domain controller
5. Add the domain controller to the existing AD domain as a domain member
6. Find information about existing domain sites and the initial replication method
7. Log in to the server with a privileged account (such as Schema Admin or Enterprise Admin)
8. Install the AD DS role
9. Configure the AD DS role
10. Review the logs to verify healthy AD DS installation and configuration
11. Configure service and performance monitoring
12. Configure AD DS backup/DR

The above checklist is simply an example. Depending on your environment, you may have to add additional tasks to the list. However, it is good to at least create a checklist before the task so we will not miss the steps.

Design topology

As per the following diagram, we are going to add an **additional domain controller** to the `rebeladmin.com` domain:

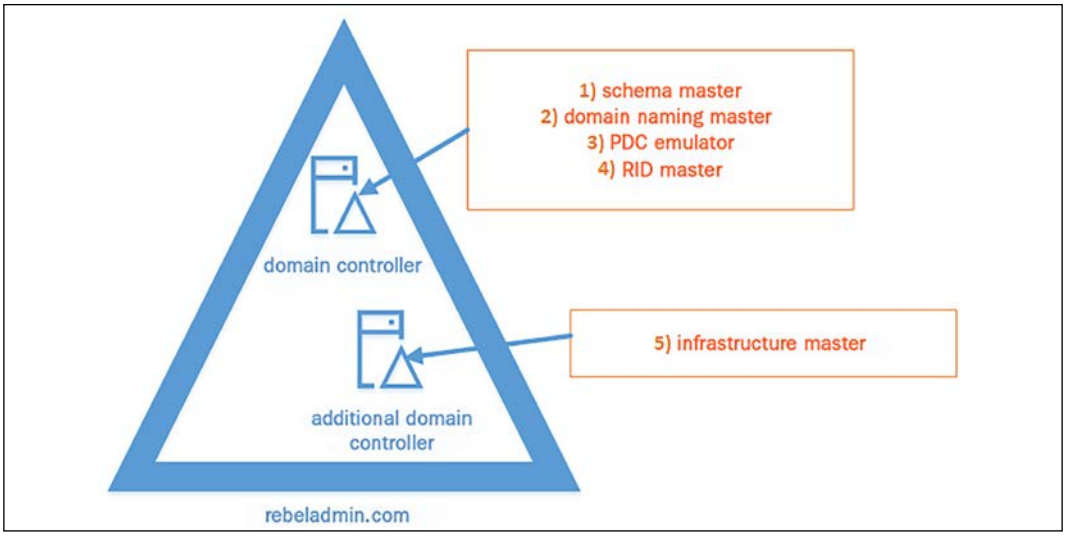


Figure 6.7: Design topology – additional domain controller

This domain controller isn't going to be a global catalog server, and so we will move the infrastructure master FSMO role over to a new domain controller at the end of the configuration.

Installation steps

The following steps demonstrate how to add an additional domain controller to an existing domain:

1. Log in to the server as a member of the Schema Admin or Enterprise Admin group.
2. Here, verify the static IP address allocation by using `ipconfig /all`.
3. Launch the PowerShell console as an administrator.
4. Install the AD DS role service:

```
Install-WindowsFeature -Name AD-Domain-Services
 -IncludeManagementTools
```

5. After successful installation of the role service, the next step is to configure the domain controller:

```
Install-ADDSDomainController
 -CreateDnsDelegation:$false
 -NoGlobalCatalog:$true
 -InstallDns:$true
 -DomainName "rebeladmin.com"
 -SiteName "Default-First-Site-Name"
 -ReplicationSourceDC "REBEL-SDC01.rebeladmin.com"
 -DatabasePath "C:\Windows\NTDS"
 -LogPath "C:\Windows\NTDS"
 -NoRebootOnCompletion:$true
 -SysvolPath "C:\Windows\SYSVOL"
 -Force:$true
```

There are no line breaks for the preceding command. The following table contains the parameters that were used here:

Argument	Description
`Install-ADDSDomainController`	This cmdlet will install the domain controller in the AD infrastructure.
`-NoGlobalCatalog`	If you don't want to create the domain controller as a global catalog server, this parameter can be used. By default, the system will enable the global catalog feature.
`-SiteName`	This parameter can be used to define the AD site name. The default value is `Default-First-Site-Name`.
`-DomainName`	This parameter defines the FQDN for the AD domain.
`-ReplicationSourceDC`	You can use this parameter to define the AD replication source. By default, it uses any available domain controller, though you can specify one if you wish.

6. Once executed, the command will ask for the **SafeModeAdministratorPassword**. Use a complex password to proceed. This will be used for DSRM.
7. After the configuration has completed, restart the system and log back in as an administrator to check the AD DS status.
8. The following command will confirm the status of the AD DS service:

```
Get-Service adws,kdc,netlogon,dns
```

9. The following command will list the domain controllers, along with their IP addresses and the sites they belong to:

```
Get-ADDomainController -Filter * |  Format-Table Name,
 IPv4Address, Site
```

10. The following command will list the global catalog servers that are available in the domain and confirm that this new domain controller server isn't a global catalog server:

```
Get-ADDomainController -Discover -Service "GlobalCatalog"
```

11. As per our plan, the next step is to move the infrastructure master role to the new additional domain controller:

```
Move-ADDirectoryServerOperationMasterRole
 -Identity REBEL-SDC-02
 -OperationMasterRole InfrastructureMaster
```

12. By using the `Get-ADDomain | Select-Object InfrastructureMaster` command, we can confirm this change.

This is the end of this scenario.

How to plan AD migrations

AD migration from an older version to a newer one is a common requirement for any AD infrastructure. As time goes by, operating systems go out of support. Even if an organization is not looking to implement new AD features, sometimes they have to migrate to a newer version if the operating system is out of support. In a typical AD migration process, a new AD DS version will be installed on a new server. Then, the FSMO roles will migrate to the new domain controllers.

Once this is completed, the older version of AD DS will be decommissioned. Afterward, the domain and forest functional levels will be raised to match the new AD DS version. Even though each AD DS version has core functions that are the same, newer versions always have new features and enhancements that apply to the domain or forest level.

There can be many reasons why an organization may consider an AD migration. I have listed a few reasons as follows:

- **To implement new features in the identity infrastructure**: Every new version of AD comes with new features and enhancements. Some of these changes are game-changers. As an example from AD DS 2016, privilege management is a turning point for identity infrastructures. In order to implement these new features, companies look for AD DS migrations.

At the time of writing, there is a new Windows Server version on the horizon. I am a geek and I always prefer to run the latest and the greatest. At the same time, I have seen that some organizations just like to run the latest but don't worry much about implementing any new features. Just migrating to a new version isn't going to give you any benefits if it's not used properly. There's no point buying a Ferrari just to drop your kids at school. Therefore, "migration" is a good time to evaluate an existing setup and see what we can do further to improve with the help of the new AD version.

- **To address support issues and compliance issues**: Back in 2015, Microsoft ended its support for Windows Server 2003. At that time, organizations that were running AD DS 2003 had no choice but to migrate to a new version. Some businesses needed to comply in order to run their operations. Businesses in the financial sector are a good example of this. These compliances have standards related to IT systems. As an example, businesses that were subject to **Payment Card Industry** (**PCI**) compliance had rules stating that they couldn't use end-of-life operating systems for operations. These types of business requirements force organizations to migrate from one AD DS version to another.
- **To fix existing issues**: The good health of AD infrastructure is key for an organization's operations and security. Like any other system, it is possible for the AD infrastructure to have issues. These could be due to bad design, configuration issues, system corruption, and so on. Fixes for these problems may lead to migrations as well. When I get my car serviced, sometimes they tell me that some parts need to be replaced. But before these parts are replaced, the mechanic normally gives me a few options. Some parts are of the same make and model, while others are for a new version. Therefore, the mechanic makes sure to explain the advantages of the new model and what it can provide me, rather than just fixing the problem. Most of the time, I end up using the new model, as it doesn't just fix the existing issue, it fixes it in a better way. So, if existing AD infrastructure issues can be fixed in a *better way* by migrating to a new version, don't hesitate to go for it. But at the same time, there are some basic AD health requirements that need to be fulfilled before performing the migration. This will be covered later in this chapter.
- **Operation requirements**: In business, there can be different operational requirements that can lead to AD migrations. New application implementations are a good example of this. Some applications only support certain AD DS schema versions. In such situations, businesses have no other option but to upgrade. There is another scenario that applies to organizations that run AD forests with multiple domain trees. As an example, Rebeladmin Corp. has three domain trees with one forest. Each domain tree represents a separate subsidiary with its own IT department.

- One company has business requirements to upgrade its domain and forest functional levels to a newer version. However, in order to do that, the other two domains also need to upgrade their AD DS versions. Although this isn't an operational requirement for two of the companies, in order to support the forest-level upgrade, there is no option but to upgrade their AD DS version.

Migration life cycle

Migrating FSMO roles to a new server and upgrading forest and domain functional levels doesn't take more than a few minutes but when it comes to migration there are few other things we need to consider. Therefore, I have summarized the AD DS migration process with the following life cycle. I called it a life cycle because organizations will keep doing AD DS migration at least to get rid of outdated operating systems:

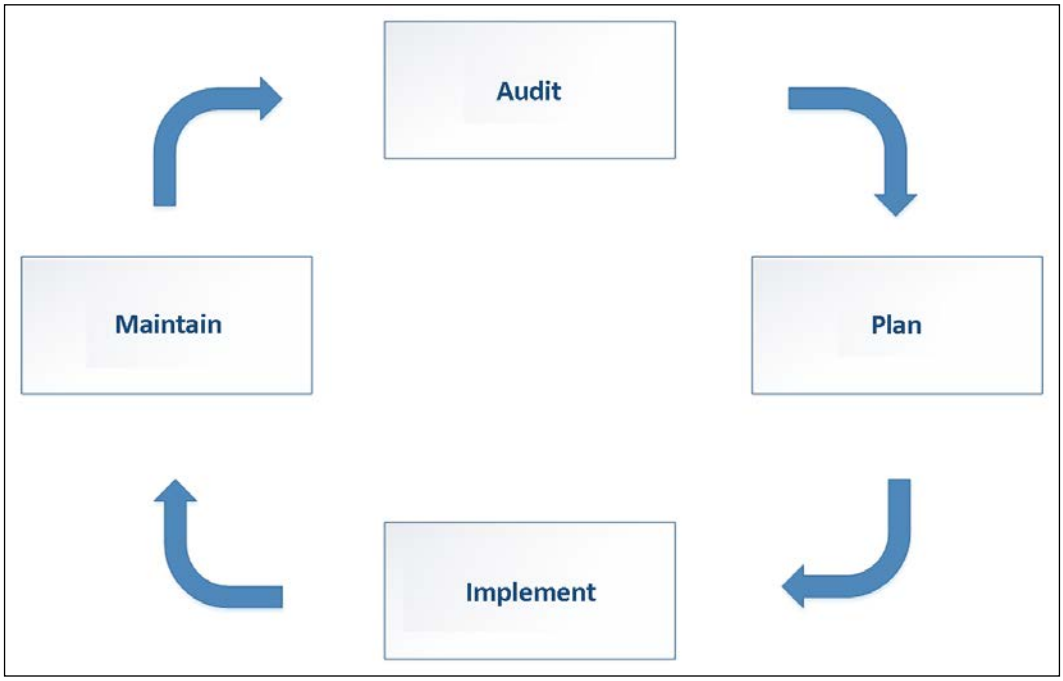


Figure 6.8: AD migration life cycle

Following the above steps will ensure that every aspect of the migration process is covered.

Auditing

In the migration process, auditing and planning are the most important stages. When you have proper auditing and a plan, the implementation process is quite easy. It minimizes the post-implementation issues. In the audit stage, we should review the current AD infrastructure and get a clear understanding of the logical and physical topology, as well as its health status. This also helps us to identify potential risks involved in the migration process.

AD logical and physical topology

A few years ago, I needed to add an additional room to my house, so I went to meet an architect and explained my requirements. When I explained it to them, I mentioned the structure of the house, where it's located, why I needed this extension, and where I thought it was best suited. Even though I explained the structure, the architect wanted to review the existing house plan. My verbal explanation about the house wasn't enough for him to create a new plan. This was because my explanation was still missing some information, such as length, width, door and window locations, and more. But the existing plan had all this information recorded, which helped a professional architect to come up with a new plan to address my requirements.

Similarly, in AD migration projects, one of my initial requests to customers is to provide an AD topology diagram. However, most of the time, I only get basic information back, such as the number of domain controllers, the number of sites, and so on. This isn't enough to get an understanding of the domain trees, replication topology, site links, and FSMO role placements. In most cases, I have to create an AD topology diagram from scratch as part of the exercise. Microsoft Active Directory Topology Diagrammer is a tool that I have used on many occasions to generate a topology diagram but unfortunately, this tool is no longer available. There is no replacement for this tool either from Microsoft's end.

Apart from the topology diagram, network diagrams also help engineers understand the physical connection between AD components. The topology diagram should include data about the AD physical structure, such as site links, the number of domain controllers, and the number of sites, but it may not give much of an insight into how traffic flows and what kind of connections and bandwidth each site has. Usually, this important information will be available on network diagrams.

Some organizations always keep these diagrams up to date with all the required changes, but this isn't the case for the majority. Lots of companies either don't have any of these diagrams or, even if they have them, they may not be up to date. Depending on the project scope, it may not be your responsibility to create a topology diagram and network diagram, but you need to collect the data that is vital for the AD DS migration project at the very least. If you are dealing with conflicting data, you can use other techniques, such as interviews and questionnaires with engineers, managers, and team leaders, to clear up doubts.

The following are the types of data we need to gather during this exercise:

- AD logical topology
- AD physical topology
- The organization's network topology
- Connection between AD components
- Bandwidth between AD sites

After we gather the above information, we will know where domain controllers are located and how they are communicating with each other. That helps us to run an AD health check and verify the status of the current deployment.

AD health check

My first car was a Honda Civic (I loved that little beast). After I bought the car, I saw little rust spots on the bonnet. I took it to a paint shop and the engineer who checked it said that it needed a fresh coat of paint. I agreed and got it done. After a few months, I started to see some bubbles again on the bonnet. I took it back to the shop since I had a 6-month warranty for the job they had done. They said they would redo the paint job. But guess what – after a few months, the same issue occurred. I didn't want to waste any more of my time, and so I took it to another place that specialized in paint jobs. When I explained the issues I was having, the engineers there performed some tests and said that they needed to remove all the paint and apply anti-rust first, which they did. After that, there were no more bubbles on the bonnet. Just applying a new coat of paint didn't fix the issue. It was only a waste of time and money.

If the AD infrastructure has got existing issues related to its core operations (such as replications, DNS, and site links), then they need to be identified and fixed before migration. There is no specific, predefined sequence for AD health checks. You can have your own checklist. The following are some key areas that need to be covered in any AD health check:

- **Replication health**: A healthy replication is critical for any AD infrastructure. All domain controllers in the infrastructure need to be aware of every change to the AD database. There are tools and techniques we can use to identify the replication issues between AD domain controllers. `Repadmin.exe` is a Microsoft-built tool that can be used to diagnose AD replication issues. Since Windows Server 2008, it has come built into the operating system, and it can be used if the AD DS role is installed. This tool needs to be run as an Enterprise Admin. If it runs as a Domain Admin, it can only be used to review domain-level replications:

```
Repadmin /showrepl
```

The preceding command will display the status of the last inbound replication of the AD partition. This will only list the replication status of the domain controller this command executes from.

If you need to check the replication status of a specific domain controller, you can use a command similar to the following. `REBEL-SDC-03` can be replaced with the name of the domain controller:

```
Repadmin /showrepl REBEL-SDC-03
```

The `/replicate` parameter can be used to trigger a replication between the domain controllers so that you can see the results in real time:

```
Repadmin /replicate REBEL-SDC-03.rebeladmin.com
 REBEL-PDC-01.rebeladmin.com DC=rebeladmin,DC=com
```

The preceding command will initiate replication of the `rebeladmin` naming context from `REBEL-PDC-01` to `REBEL-SDC-03`.

The following commands will initiate the full replication of all the changes from `REBEL-PDC-01` to `REBEL-SDC-03`:

```
Repadmin /replicate REBEL-SDC-03.rebeladmin.com
 REBEL-PDC-01.rebeladmin.com DC=rebeladmin,DC=com /full
```

The `/replsummary` parameter can be used so that you can view the summary of the replication status of all the domain controllers:

```
Repadmin /replsummary
```

The preceding command will provide a summary of all the domain controllers in the infrastructure.

The following command will only list the domain controllers that have replication issues with other domain controllers:

```
Repadmin /replsummary /errorsonly
```

Windows Server 2022 domain controllers only can add to an AD environment that uses DFSR for SYSVOL replication. If FRS is still in use, before we add the domain controller, we need to migrate SYSVOL replication from FRS to DFSR.

We can check if SYSVOL replication uses DFSR by using:

```
dfsrmig /getmigrationstate
```

If the above command returns the state as `eliminated`, it means DFSR is already in place.

- **Event Viewer**: Event Viewer can also be used to evaluate the replication health of the AD environment. There are certain event IDs you can use to filter this data. You can find these events under **Event Viewer | Application and Service Logs | Directory Services**.

  Here, I have listed some key event IDs that will show replication problems:

Event ID	Cause
1925	The attempt to establish a replication link for a writable directory partition failed. This can be caused by network issues, domain controller failures, or DNS issues.
1988	The local domain controller has attempted to replicate an object from a source domain controller that isn't present on the local domain controller because it may have been deleted and already garbage-collected. Replication will not proceed for this directory partition with this partner until the situation is resolved. This happens when a domain controller is down for a long time (more than the tombstone's lifetime) before being brought back online. After this, however, it could have non-existing objects (lingering objects). They need to be cleaned to initiate replication again.
2087	AD DS could not resolve the DNS hostname of the source domain controller to an IP address, and replication failed. This will show up in the destination domain controller when it cannot resolve the DNS name for its source domain controller. If DNS lookup fails in the first place, it will also try FQDN and NetBIOS to resolve the name. It will prevent replication until it's been resolved.
2088	AD DS could not resolve the DNS hostname of the source domain controller to an IP address, but replication succeeded. In this situation, the destination domain controller failed to resolve the source name using DNS lookup, but it was able to connect to it using the FQDN or NetBIOS name.
1311	The replication configuration information in AD DS doesn't accurately reflect the physical topology of the network. This usually occurs due to the misconfiguration of AD site links. It may have the wrong subnets assigned to it.

Once an object is deleted from the directory, it will not be deleted from the AD database right away. It will be removed by the garbage collector once it passes the tombstone lifetime value. The default value is 180 days.

- **Domain controller health**: In the previous section, we learned how to evaluate the replication's health. The next step is to check the health of the domain controllers. It is recommended only to install AD DS roles and features in the domain controller. But many use roles such as DHCP or NPS on the domain controller. As part of the health check, we need to check what the additional roles installed in the domain controller are. This can be done using:

```
Get-WindowsFeature -ComputerName DC01 | Where Installed
```

In the above command, the `-ComputerName` value should be replaced by the domain controller hostname.

We should also avoid installing additional software in domain controllers (unless it is for backup or security). We can find additional software installed in a domain controller by using:

```
Get-ItemProperty HKLM:\Software\Microsoft\Windows\
CurrentVersion\Uninstall\* | select DisplayName
```

```
PS C:\Windows\System32> Get-ItemProperty HKLM:\Software\Microsoft\Windows\CurrentVersion\Uninstall\* | select DisplayNam
e

DisplayName
-----------


PuTTY release 0.74 (64-bit)
PowerShell 7-x64
```

Figure 6.9: List of software installed in a domain controller

The `Dcdiag.exe` tool can be used to run predefined tests to evaluate the health of the domain controllers:

```
Dcdiag /e
```

The preceding command will test the domain controllers in the forest.

```
Dcdiag /s:REBEL-SDC-03
```

The preceding command will run the test on the `REBEL-SDC-03` domain controller.

Instead of running all the tests, the following command will only run a replication test on `REBEL-SDC-03`:

```
Dcdiag /test:replications /s:REBEL-SDC-03
```

The following command can be used to check AD services on the local domain controller:

```
Dcdiag /test:Services
```

- **DNS health**: We can't talk about AD health without talking about a healthy DNS infrastructure. AD heavily depends on DNS functionalities.

  To start with, I prefer to review the DNS server-related events in the domain controllers. We can access the DNS logs by going to **Event Viewer** | **Application and Service Logs** | **DNS Server**.

  The `Dcdiag` utility can also be used to test DNS health:

```
Dcdiag /test:DNS /DNSBasic
```

  The preceding command will run a basic DNS check to ensure that the DNS services are running, the resource records are registered, and the DNS zones are presented.

  The following command will test whether the DNS forwarders are functioning properly:

```
Dcdiag /test:DNS /DnsForwarders
```

  The following command will test the registration of DC locator records:

```
Dcdiag /test:DNS /DnsRecordRegistration
```

  We also can run a complete health report by using:

```
dcdiag /v /c /e /s:DC01 | Out-File C:\healthreport.txt
```

  In the preceding command, `/v` means verbose and it will print extended information. The `/c` argument means the system will run all tests except DCPromo and RegisterInDNS. With the `/e` argument tests will run on all servers in the enterprise. The `/s` argument is used to define the name of the domain controller to run the commands against. The output of the complete command will be written to the `C:\healthreport.txt` file. This report includes rich information about the health of the AD infrastructure.

In this section, I have mentioned the most common things we need to check during an AD health check. However, based on the findings from our testing, we will need to perform an additional test. The whole point of an AD health check is to verify the state of the AD environment and make sure it is ready for an upgrade.

SCOM and Azure Sentinel

SCOM has the Active Directory Management Pack and DNS management packs that monitor the application-level health events.

Azure Sentinel can collect events from domain controllers and that information can be used to evaluate the health and security of the AD environment. We can use KQL queries with Sentinel to find specific security and health events.

Here, I have summarized a list of things I usually try to cover during an AD health check:

- Review the connection status between domain controllers
- If the organization has a monitoring system, review the reports and latest events about domain controllers, AD DS roles, replication health, and DNS services
- Review the latest backup reports
- Review DNS issues and events
- Review the AD domain controller's health
- Test AD replications
- Review AD logs to find any recurring issues
- Review the existing domain controller's performance
- Review bandwidth utilization between site links

After the health check, the next step of the process will be to verify the application dependencies.

Application auditing

If organizations are running AD DS, it's obvious they also have AD-integrated applications. Some of them may use it just for LDAP authentication, while some may use advanced integration with a modified AD schema. With AD migration, some of these applications may require modifications or upgrades to match the new AD DS version.

Therefore, before the implementation process, it is important to recognize these AD-integrated applications and evaluate the risks:

- **LDAP connection string modifications**: In order to use **single sign-on** (**SSO**) with applications, AD may use LDAP connections to the domain controllers. Sometimes, applications use hardcoded hostnames or the IP addresses of domain controllers in connection strings. If domain migration involves IP address changes and hostname changes, alternate settings will need to be planned.

- **Schema version changes**: Some legacy applications only support certain versions of the AD schema. This is mostly applicable to custom-made AD-integrated applications. This is very rare, but I have seen them in my AD migration projects. Therefore, if it's not a well-known application, check with the vendor whether it supports the new AD DS schema version.
- **Application migrations**: Some organizations have legacy application versions that are no longer supported or developed by their vendors. Once, I was working on an AD DS 2003 to AD DS 2012 R2 migration project. The organization had a legacy application that ran on the Windows Server 2000 system. AD DS 2012 R2 doesn't support Windows Server 2000 member servers. The vendor who created the application was no longer in business. As a result, we had to migrate users to a similar type of application that supported the new operating system before we started with the actual AD migrations.
- **Server roles/applications installed on domain controllers**: In the majority of cases, once the FSMO roles are migrated to new domain controllers, the old domain controllers will be decommissioned. Even though Microsoft recommends not to install applications or other server roles in domain controllers, people still do it. Some of the common roles that are installed in domain controllers are DHCP and NPS.

If existing domain controllers are subject to decommissioning, these applications and server roles need to migrate to new servers. Some of these roles or application versions may not be on the market anymore. For example, the Windows **Internet Authentication Service** (**IAS**) in Windows Server 2003 was replaced by the **Network Policy Server** (**NPS**) in Windows Server 2008. If you cannot use the same version in migration, you will need to plan for upgrades or replace it with an equivalent application.

Planning

After a successful auditing phase, we have a lot of data and insight into the existing AD infrastructure. In the planning process, I usually reevaluate the collected data and make a blueprint to follow in the implementation process. This can be presented as a document so that each party that's involved in the project is aware of it.

The following information needs to be covered in the plan:

Data	Description
Overview of the existing AD DS infrastructure	Based on the data that was collected from the audit, you need to provide an overview of the existing infrastructure. This should include information about the logical and physical topology of AD.

Overview of the proposed solution	Based on the data that was collected from the audit and business requirements, we can provide a detailed design of the proposed solution. This should include data about the topology changes, new domain controller placements, FSMO role placements, new site links, IP addresses, hostnames, required hardware or VM resources, required firewall rule changes, and more.
Risks	One main objective of the audit exercise is to identify the potential risks that can impact the AD DS migration. These can be due to wrong design, the bad health of the AD services, or other infrastructure or application issues. The recognized risks can be categorized based on impact (for example, high, medium, and low).
Risk mitigation plan	Once the risks have been identified, we need to provide a plan to describe what action can be taken to address them. If possible, include a task list, estimated time frame, and budget in the plan.
Service interruptions	During the implementation process, there can be service interruptions. This can be due to events such as application migrations or server IP changes. In this section, make a list of these service interruptions, along with the expected time range, so that the relevant parties can be informed prior to the migration process.
Recommendations	During the audit process, you may have found things that you could do to improve AD DS performance, security, or manageability. What wasn't covered in the initial business requirements can be listed as recommendations. Note that these shouldn't have any direct impact on the AD DS migration process. If this is done, it should be listed in the proposed solution section.
Task list and schedule	The plan should have a detailed task list and schedule for the AD DS migration implementation process. It should also include roles and responsibilities for completing each task.
Test plan	It is also required that you have a detailed test plan in order to test the AD functions after the AD DS migration process so that you can verify its health and integrity. This must be used during the implementation process and should include evidence to prove the successful completion of each test (such as screenshots, events, reports, and more).
Recovery plan	After a successful audit and planning process, there is a very low possibility of project failure. However, the plan still needs to provide a recovery plan that will be used in the event of a failure. It also should include a process for testing the existing DR or backup solution that will be used in the recovery, prior to starting the project.

Once the plan has been produced, explain it to everyone involved in the project. You will need to explain their roles and responsibilities in the project. You also may need to get the plan approved by management before implementation.

Implementation

After a successful audit and planning process, the next step is to perform the implementation. If you did your homework correctly in the previous phases, this process will be straightforward and will end with a successful result.

AD migration checklist

Based on the previous phases, I have created the following checklist that you can use for the AD migration process:

- Evaluate the business requirements for AD migration
- Perform an audit on the existing AD infrastructure
- Provide an implementation plan
- Prepare the physical/virtual resources for the domain controller
- Install Windows Server 2022 Standard/Datacenter
- Patch the servers with the latest Windows updates
- Assign a dedicated IP address to the domain controller
- Install the AD DS role
- Migrate the application and server roles from the existing domain controllers
- Migrate the FSMO roles to the new domain controllers
- Add new domain controllers to the existing monitoring system
- Add new domain controllers to the existing DR solution
- Decommission all the old domain controllers
- Raise the domain and forest functional levels
- Perform ongoing maintenance (Group Policy review, new feature implementations, identifying and fixing AD infrastructure issues, and more)

As part of this exercise, I am going to demonstrate how to perform migration from AD DS 2008 R2 to AD DS 2022.

The Windows Server 2008 and Windows Server 2008 R2 operating systems reached the end of their support cycle on 14th January 2020. Because of this many organizations wanted to migrate away from these legacy operating systems. End-of-life operating systems have a direct impact on various industry compliances, IT audits, penetration tests, and so on. Even if a business does not have a business requirement to upgrade, end-of-life operating systems leave us no choice but to upgrade.

Design topology

As per the following diagram, the `rebeladmin.net` domain has two domain controllers:

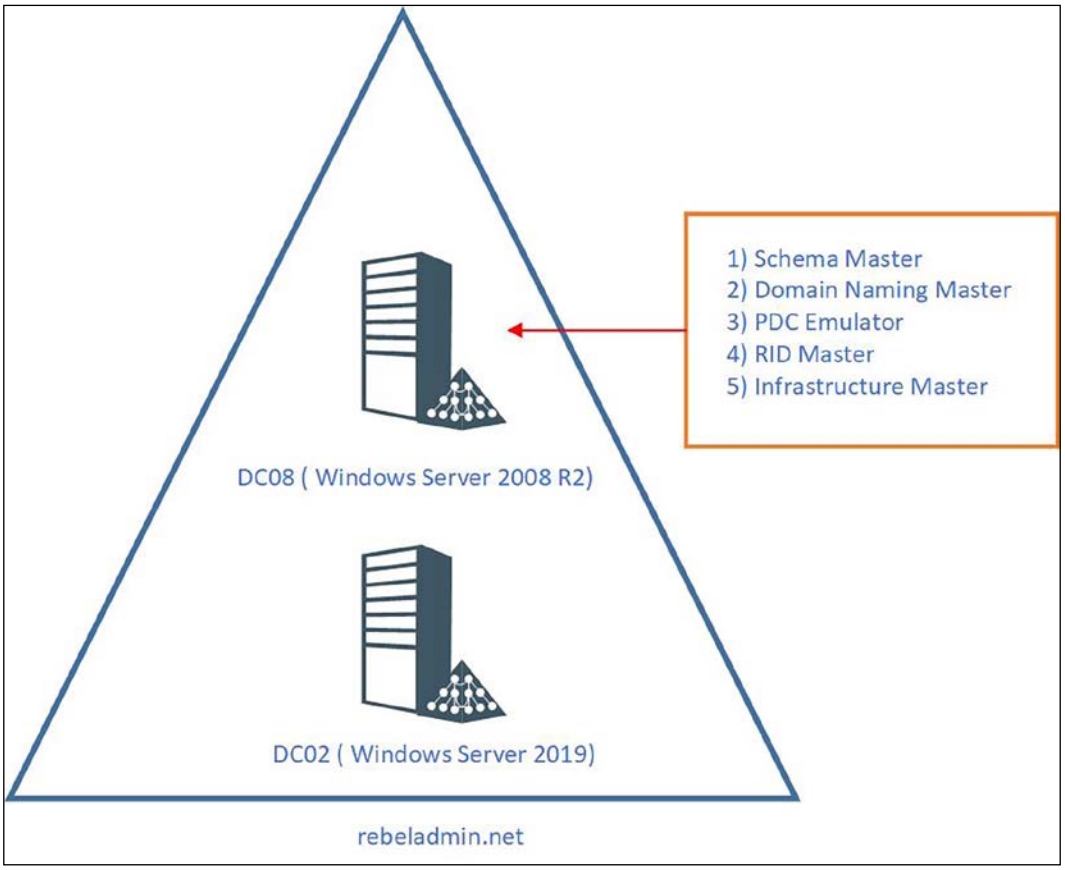


Figure 6.10: Design topology – AD DS 2008 R2 migration

As explained in *Figure 6.10*, the FSMO role holder `DC08` is a Windows Server 2008 R2 domain controller. The domain and forest functional levels currently operate in Windows Server 2008 R2. A new domain controller with Windows Server 2022 will be introduced and will be the new FSMO role holder for the domain. Once the FSMO role migration is complete, the domain controller running Windows Server 2008 R2 will be decommissioned. After that, the forest and domain functional levels will be raised to Windows Server 2016.

Here, `DC08` is the domain controller with Windows Server 2008 R2, while `DC22` is the domain controller with Windows Server 2022.

When you introduce new domain controllers to an existing infrastructure, it is recommended that you introduce the forest root level first and then go to the domain tree levels.

Installation steps

Using the following steps, we can install the new domain controller and migrate FSMO roles to it:

1. Log in to the server as a member of the local administrators group.
2. Add the server to the existing domain as a member.
3. Log in to the **DC08** domain controller (Windows Server 2008 R2) as a Domain Admin.
4. Then open the PowerShell console as an administrator and run `dfsrmig /getmigrationstate`. If the command returns the state as `eliminated`, it means DFSR is already in use for SYSVOL replication. If it is not, we must migrate SYSVOL replication to DFSR as Windows Server 2022 does not support FRS replication. FRS to DFSR migration steps are covered in a blog post I wrote and it can be accessed via `https://bit.ly/3CM8VeG`.

   In this demo environment, the domain controller is already using DFSR:

```
PS C:\Users\dfrancis> dfsrmig /getmigrationstate

All Domain Controllers have migrated successfully to Global state ('Eliminated').
Migration has reached a consistent state on all Domain Controllers.
Succeeded.
PS C:\Users\dfrancis> _
```

Figure 6.11: DFSR status

5. Log in to **DC22** (Windows Server 2022) as an Enterprise Admin.
6. Before the configuration process, we need to install the AD DS role in the given server. In order to do that, we can use the following command:

```
Install-WindowsFeature -Name AD-Domain-Services
 -IncludeManagementTools
```

7. Then launch the PowerShell (7.1) console and run the following commands to verify the current FSMO role holder:

```
Get-ADDomain | Select-Object InfrastructureMaster, RIDMaster,
PDCEmulator

Get-ADForest | Select-Object DomainNamingMaster, SchemaMaster
```

As we can see, all five FSMO roles currently belong to the **DC08** (Windows Server 2008 R2) domain controller:

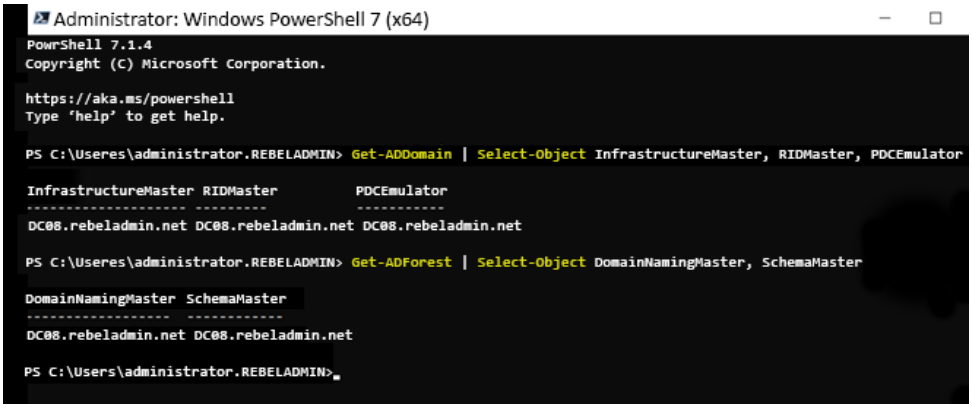


Figure 6.12: Current FSMO role holder

8. Configure the new server as an additional domain controller (these steps were covered in the *Setting up an additional domain controller* section).
9. Migrate all five FSMO roles to the new domain controller by running the following command in the **DC22** server:

```
Move-ADDirectoryServerOperationMasterRole -Identity
  DC22 -OperationMasterRole SchemaMaster,
  DomainNamingMaster, PDCEmulator, RIDMaster,
  InfrastructureMaster
```

In the preceding command, `DC22` is the domain controller running Windows Server 2022.

Figure 6.13: Migrate FSMO roles

10. Once we're done, we can verify the new FSMO role holder using the following commands:

```
Get-ADDomain | Select-Object InfrastructureMaster, RIDMaster,
PDCEmulator

Get-ADForest | Select-Object DomainNamingMaster, SchemaMaster
```

```
PS C:\Users\administrator.REBELADMIN> Get-ADDomain | Select-Object InfrastructureMaster, RIDMaster, PDCEmulator

InfrastructureMaster RIDMaster          PDCEmulator
-------------------- ---------          -----------
DC22.rebeladmin.net  DC22.rebeladmin.net DC22.rebeladmin.net

PS C:\Users\administrator.REBELADMIN> Get-ADForest | Select-Object DomainNamingMaster, SchemaMaster

DomainNamingMaster  SchemaMaster
------------------  ------------
DC22.rebeladmin.net DC22.rebeladmin.net
```

Figure 6.14: New FSMO role holder

As expected, now FSMO roles are successfully moved to the **DC22** domain controller (Windows Server 2022).

11. The next step is to decommission the old Windows domain controllers running on Windows Server 2008 R2. To do that, execute the DCPromo wizard as an Enterprise Admin from the relevant domain controller.

    Once this has been completed, DC08 will be a member server of the `rebeladmin.net` domain:

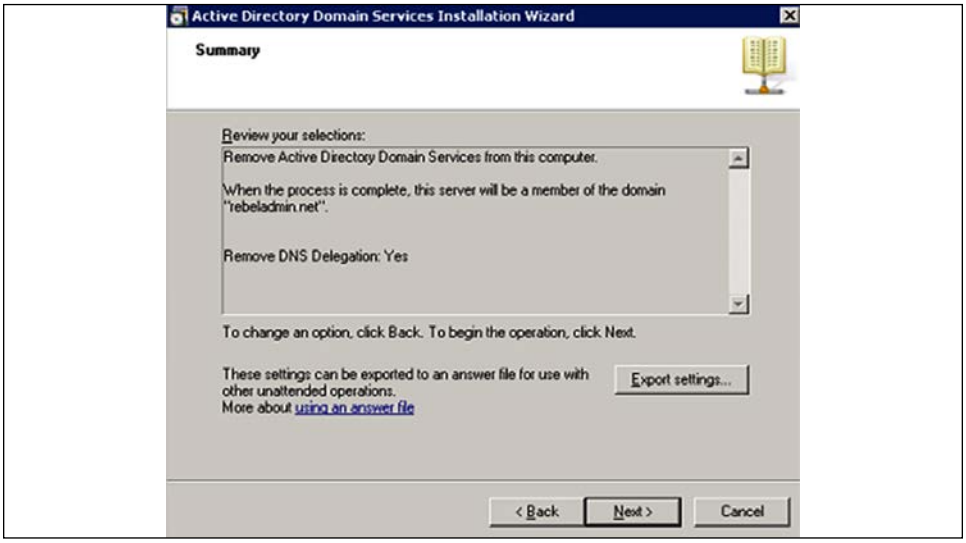


Figure 6.15: DCPromo wizard

If it's Windows Server 2012 or above we can use `Uninstall ADDSDomainController -DemoteOperationMasterRole -RemoveApplicationPartition` to uninstall AD DS.

12. The next step of the migration is to raise the domain and forest functional levels to Windows Server 2016. As I explained in *Chapter 2, Active Directory Domain Services 2022*, Windows Server 2022 doesn't have a new domain or forest functional level. To change the domain functional level to Windows Server 2016, use the following command:

```
Set-ADDomainMode -identity rebeladmin.net
 -DomainMode Windows2016Domain
```

Figure 6.16: Upgrading the domain functional level

To upgrade the forest functional level, use the following command:

```
Set-ADForestMode -Identity rebeladmin.net
 -ForestMode Windows2016Forest
```

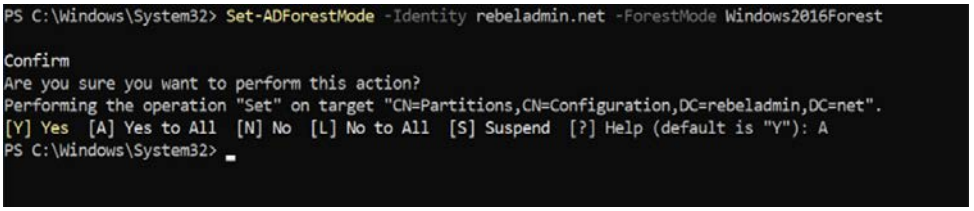


Figure 6.17: Upgrading the forest functional level

Now, we have completed the migration from AD DS 2008 R2 to AD DS 2022. The same steps apply when you're migrating from Windows Server 2012, Windows Server 2012 R2, Windows Server 2016 and Windows Server 2019.

Verification

Although the migration is complete, we still need to verify whether it's completed successfully. The following command will show the current functional level of the domain after the migration:

```
Get-ADDomain | fl Name,DomainMode
```

The following command will show the current forest functional level of the domain after migration:

```
Get-ADForest | fl Name,ForestMode
```

```
PS C:\Users\administrator.REBELADMIN> Get-ADDomain | fl Name,DomainMode

Name       : rebeladmin
DomainMode : Windows2016Domain


PS C:\Users\administrator.REBELADMIN> Get-ADForest | fl Name,ForestMode

Name       : rebeladmin.net
ForestMode : Windows2016Forest


PS C:\Users\administrator.REBELADMIN>
```

Figure 6.18: Verifying the forest and domain functional levels

You can also use the following command to verify the forest and domain functional level updates:

```
Get-EventLog -LogName 'Directory Service' | where {$_.eventID -eq 2039
-or $_.eventID -eq 2040} | Format-List
```

The following screenshot shows events `2039` and `2040` in the `Directory Service` log, which verify the forest and domain functional level updates:

```
PS C:\Users\administrator.REBELADMIN> Get-EventLog -LogName 'Directory Service' | where {$_.eventID -eq 2039 -or $_.eventID -eq 2040} | Format-List

Index              : 100
EntryType          : Information
InstanceId         : 1073743864
Message            : The functional level of this forest has been updated.


                     New forest functional level:7

Category           : Directory Access
CategoryNumber     : 8
ReplacementStrings : {7}
Source             : NTDS General
TimeGenerated      : 02/09/2021 15:54:10
TimeWritten        : 02/09/2021 15:54:10
UserName           : REBELADMIN\Administrator

Index              : 87
EntryType          : Information
InstanceId         : 1073743863
Message            : The functional level of this domain has been updated.


                     Domain: DC=rebeladmin,DC=net

                     New domain functional level:7

Category           : Directory Access
CategoryNumber     : 8
ReplacementStrings : {DC=rebeladmin,DC=net, 7}
Source             : NTDS General
TimeGenerated      : 02/09/2021 15:53:10
TimeWritten        : 02/09/2021 15:53:10
UserName           : REBELADMIN\Administrator


PS C:\Users\administrator.REBELADMIN>
```

Figure 6.19: Verifying the forest and domain functional levels – event log

Get-EventLog is not recognized as a command in PowerShell 7.1. We have to use native Windows PowerShell for that:

Event ID 1458 verifies the transfer of the FSMO roles:

```
Get-EventLog -LogName 'Directory Service' | where {$_.eventID -eq 1458}
| Format-List
```

You can use the following command to verify the list of domain controllers and make sure that the old domain controller is gone:

```
Get-ADDomainController -Filter * | Format-Table Name, IPv4Address
```

Apart from these, you can also go through the directory service and DNS logs to see whether any issues have been recorded.

Maintenance

I am a petrol head; I love the smell of burning fuel. I always service my car at the right time, wash it regularly, put in the best oil, and do all the tune-ups when required. Because of that, my little beast never gets me into trouble when I'm on the go – touch wood!

Likewise, it doesn't matter how good your AD infrastructure is today; if you don't maintain and tune it, you aren't going to get much out of it. Here, I have listed things you need to do after AD migration to get the most out of it:

- **Add to the monitoring system**: The new domain controllers now hold the responsibility of being your identity infrastructure. It is important to be notified if a part of the hardware or system service has failed that will affect the company operations. For that task, I prefer to use an advanced application-layer monitoring system, such as SCOM or Azure Sentinel, that not only alerts you about service and system failures but also predicts issues in advance and allows engineers to rectify them.
- **Add to the DR solution**: In the event of a hardware failure or natural disaster, the company should be able to recover its workloads to continue its operations. There are many different solutions out there that we can use as DR solutions for an AD environment. My preference for this is to keep additional domain controllers in DR sites, along with a backup. In a disaster, this will allow other applications to continue their operations with minimum impact. Once you add new domain controllers to the backup or DR solution, make sure to test them periodically to verify their validity.

- **Implement new features**: Once the domain and forest functional levels have been updated, you can start using the features of AD DS 2022, which I described in *Chapter 2*, *Active Directory Domain Services 2022*. Applying new features is one of the main objectives of any AD DS migration project.

  When you're applying features, try to apply them to test devices or a group of test users first before applying them organization-wide. This will minimize their impact if you need to alter or completely remove them. The features you can use for your organization depend on the organization's business model and operations.

- **Group Policy reviews**: Group policies can be used to manage systems, application and security settings for users, devices, and other resources in the AD infrastructure. As the system migrates from one AD DS version to another, Group Policy capabilities change too. In an infrastructure, there can be group policies that contain legacy settings that are no longer valid for the organization's operations. Otherwise, the newer AD DS version may have a *better way* of doing things. Therefore, after AD DS migrations, review your group policies and make any required amendments or implementations. For Group Policy testing, always try it against a test group and test devices before applying it to production.
- **Documentation**: Documentation is required for any system implementation. Once the migration process is complete, prepare a document that includes information about the design, implementation steps, configuration changes, test results, the resources that have been used, Group Policy changes, new feature configurations, and more. It will be a good starting point for engineers and management so that they can plan future AD DS migrations. It will also help engineers when they do system troubleshooting.

Summary

The first few chapters of this book were focused on understanding AD DS and its capabilities. This chapter is different from those as it is more focused on the *implementation* of AD DS. In the first part of this chapter, we learned about the implementation of domain controllers in different scenarios. The second part of this chapter was focused on AD DS migration from an older version of AD DS to AD DS 2022. As part of this learning experience, we looked at how to perform AD health checks, application audits, information gathering, and AD design reviews. Last but not least, we learned how to migrate from AD DS 2008 R2 to AD DS 2022.

In the next chapter, we are going to learn about managing AD objects.

7
Managing Active Directory Objects

I started my career as a web content developer. I still remember my first day at work. It was a small software development company with about 20 engineers. I didn't know anything about **Active Directory** (**AD**) back then. On my first day, after the introduction process, my manager showed me to my desk. Then he told me my username and password for login. So, I turned on the computer and typed in my username and password to log in. Then, an on-screen message appeared saying I needed to set a new password. I typed in the most complex password I could think of for extra security, as instructed by my manager. After that, I logged in and started working. It was a pretty busy morning. After a quick break in the late afternoon, I came back to my seat to continue my work. I typed in my complex password to log in but failed. I tried it again but had the same result. I kept on trying, and after a few attempts I saw an account lockout message. I was kind of panicked, as this was the first time that I had come across this. I showed it to the guy next to me and he told me to go and talk to the system administrator. I walked into the server room and talked about my account issue to the administrator. He then logged in to a remote machine and opened up a type of console on his screen. Then, he expanded some folders in a folder tree and selected something that had my name on it. After a few clicks, he opened a small box and asked me to type in a new password. After a few years (and after I had changed careers), I realized that that was the **Active Directory Users and Computers** (**ADUC**) **Microsoft Management Console** (**MMC**). In the beginning, all of the theory about objects and attributes was all Greek to me. But when I started using **Active Directory** (**AD**) object management consoles, it started to make sense. It is impossible to explain AD without these tools, which manage AD objects as visual components of the AD infrastructure.

As explained in *Chapter 1*, *Active Directory Fundamentals*, the things we need to represent in AD are created and saved as objects. These can be users, computers, printers, or groups. Attributes are used to describe these objects. It's similar to the way we use characteristics to describe people or things. There are different tools and methods we can use to add, modify, or remove objects from the AD database.

AD object management is one of the basic skill requirements for AD administration. Adding/removing objects and modifying the attributes of objects are quite common AD administration tasks in an AD environment. By going through this chapter, you will learn about the different tools and techniques you can use to improve the AD object management experience.

In this chapter, we will cover the following:

- Tools and methods for managing AD objects
- Creating, modifying, and removing objects in AD
- Finding objects in AD

There are many different tools and methods to manage AD objects. Each tool or method has its own advantages and disadvantages. It is difficult to say which tool or method is best as it depends on the user's preferences, but knowing about the different tools and methods will help engineers to select the best tool for the job.

Tools and methods for managing objects

There are different tools and methods we can use to manage AD objects. When you install AD DS on a server, it will also enable access to these management tools. There are other third-party vendors who build AD management tools as well. But in this chapter, we will only be using built-in tools on Windows Server systems.

Windows Admin Center

Windows Admin Center is the latest Microsoft server management tool. It is a web-based application that you can use to manage Windows Server instances, Windows 10 PCs, clusters, and hyper-converged systems in an infrastructure. We also can use this tool to manage AD servers and AD objects. Before we look at that, let's study the architecture behind Windows Admin Center:

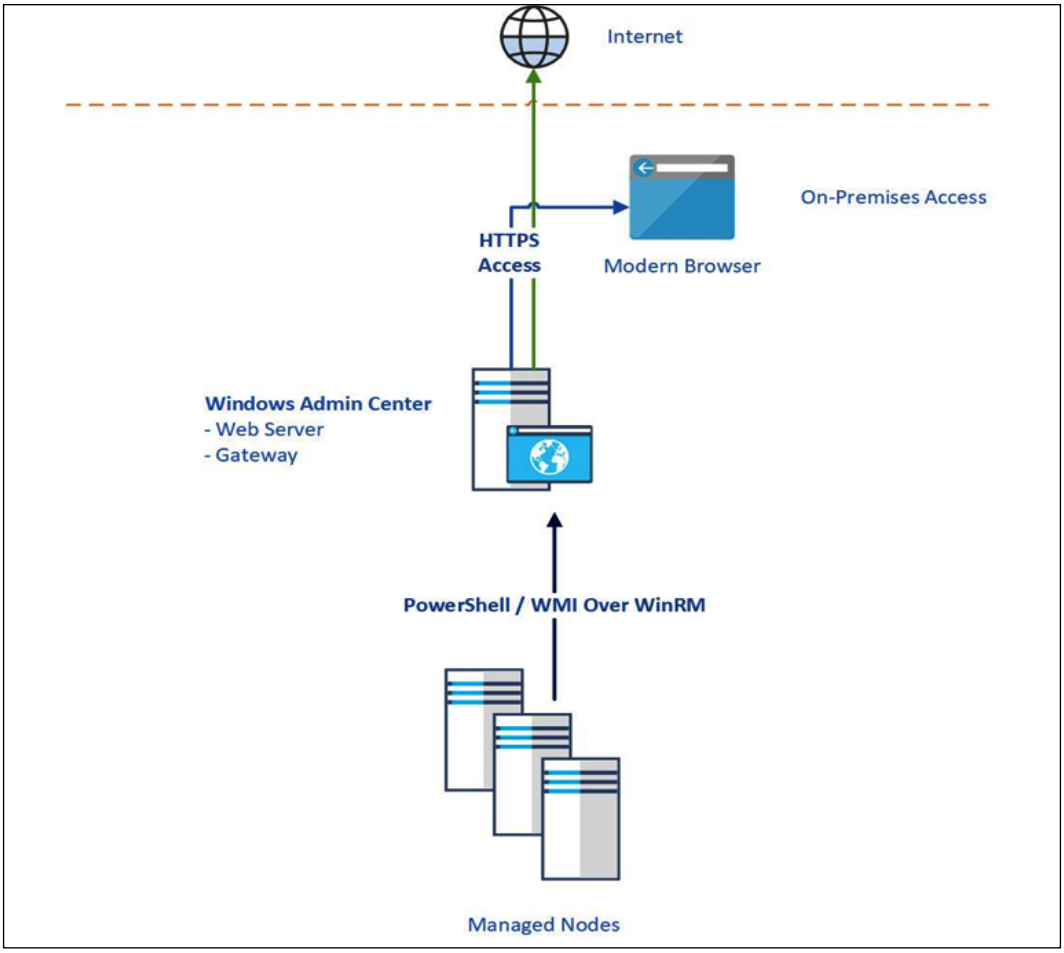


Figure 7.1: Windows Admin Center architecture

Once deployed, Windows Admin Center can be accessed from anywhere as long as the necessary firewall rules are in place. Windows Admin Center is a lightweight installation and can be installed on Windows 10 clients or any other managed host. It also has gateway roles, which you can use to manage Windows Server 2022, Windows Server 2019, Windows Server 2016, Windows Server 2012 R2, Windows Server 2012, and Windows 10 PCs using PowerShell and WMI over WinRM. Windows Admin Center is a free product and does not require any additional licenses. If needed, Windows Admin Center can be installed on an Azure virtual machine. Windows Admin Center can also integrate with many Azure services, such as Azure AD, Azure Site Recovery, and Azure Backup.

I am not going to explain here how to install Windows Admin Center as how that is done will depend on any additional business requirements that you may have. More information about deployment, planning, and installation is available at `https://bit.ly/3xhs5bb`. You cannot install Windows Admin Center in a domain controller. It must be a member server and it doesn't need to be domain joined.

Once Windows Admin Center is installed, log in to the portal and install the AD extension. By default, it is not installed:

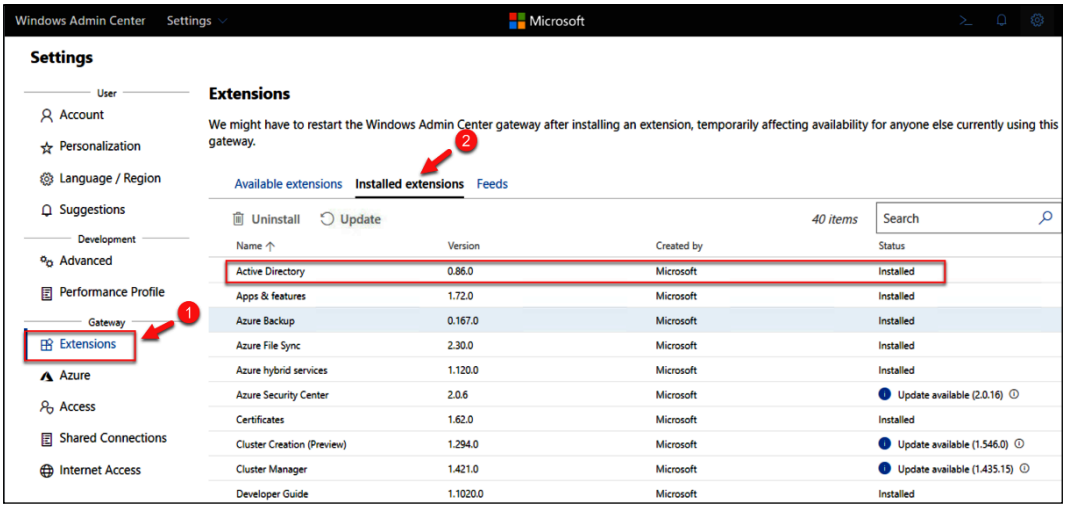


Figure 7.2: Windows Admin Center extensions

Then add a domain controller to Windows Admin Center by using the domain admin account. When the server is connected, go to the server properties and click on **Active Directory**:

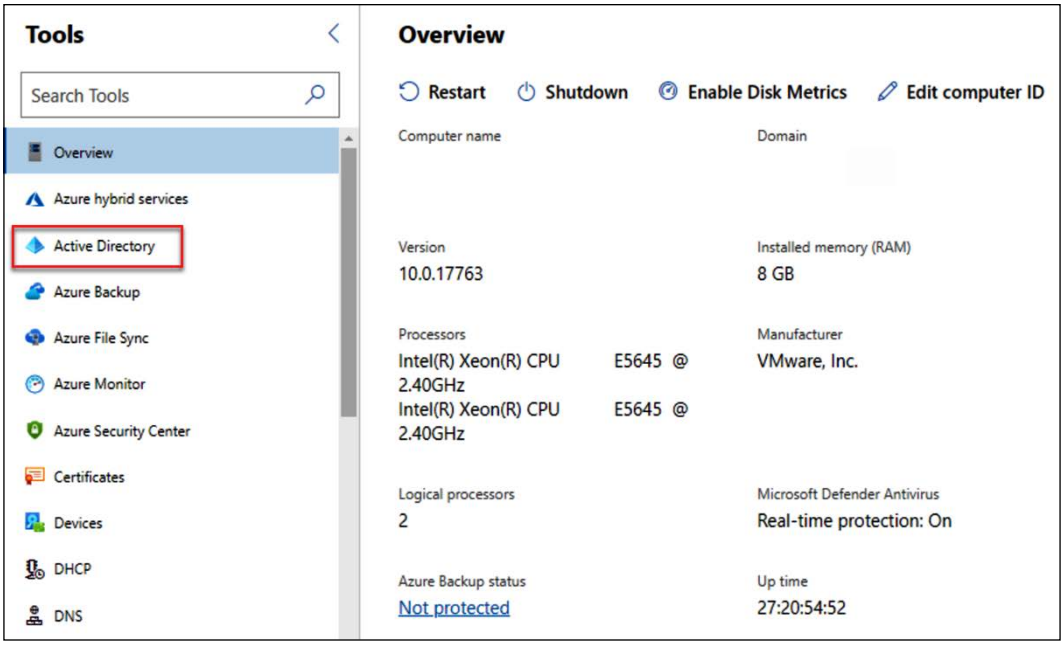


Figure 7.3: Windows Admin Center AD role management

We can start managing AD objects by searching for an object using the **Search** option:

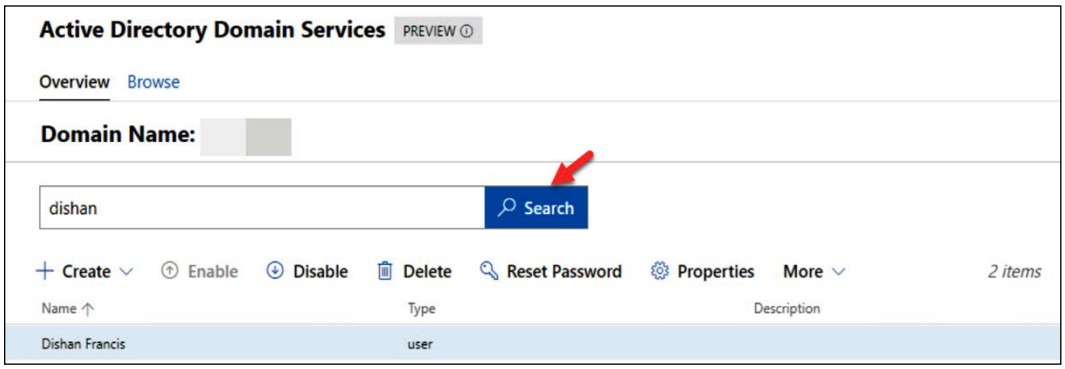


Figure 7.4: Searching AD users using Windows Admin Center

Once an object is filtered, we can view the properties of the account by using the **Properties** option:

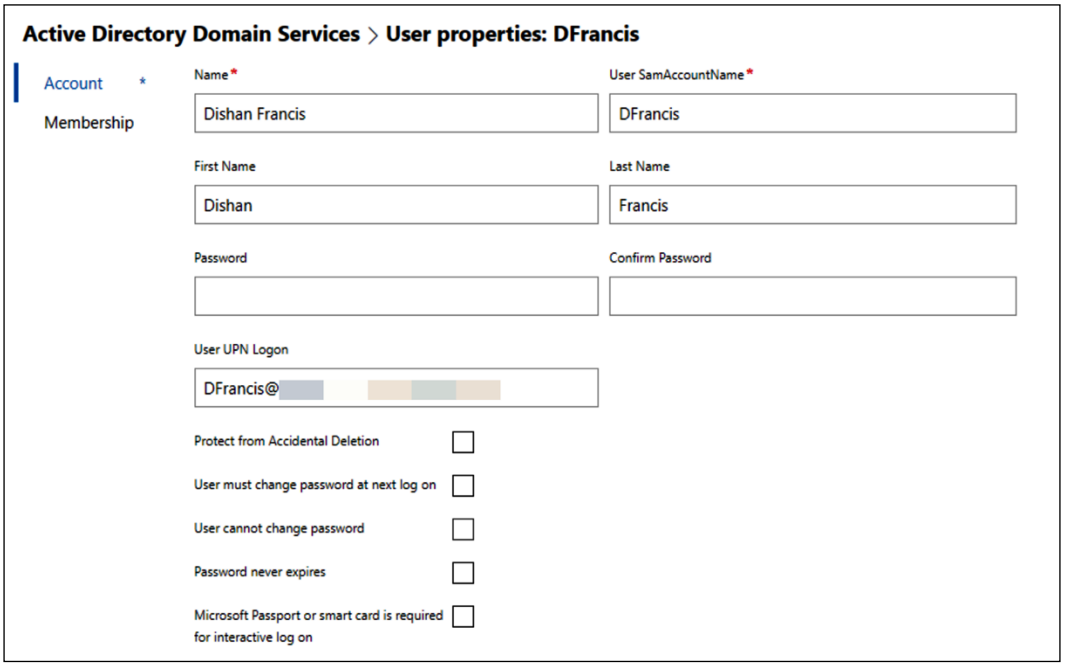


Figure 7.5: Modifying a user object using Windows Admin Center

Apart from that, we also can disable an object, delete an object, or reset the password for an object:

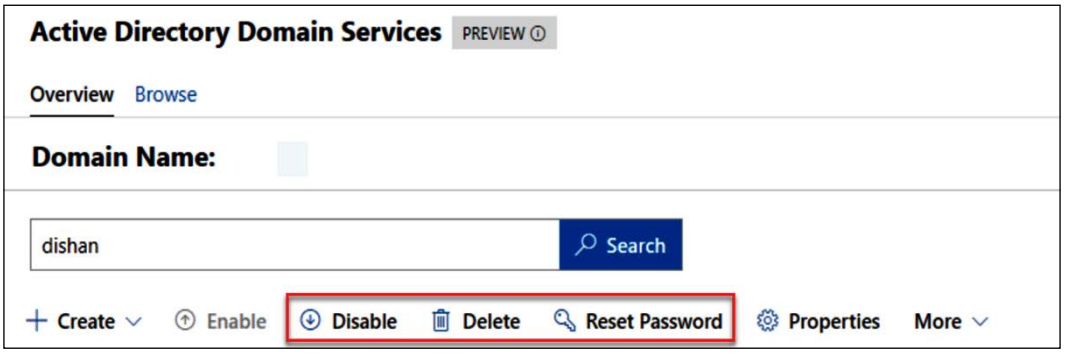


Figure 7.6: Disabling, deleting, or resetting the password for an object using Windows Admin Center

We also can create a user, group, or **organizational unit** (**OU**) using Windows Admin Center:

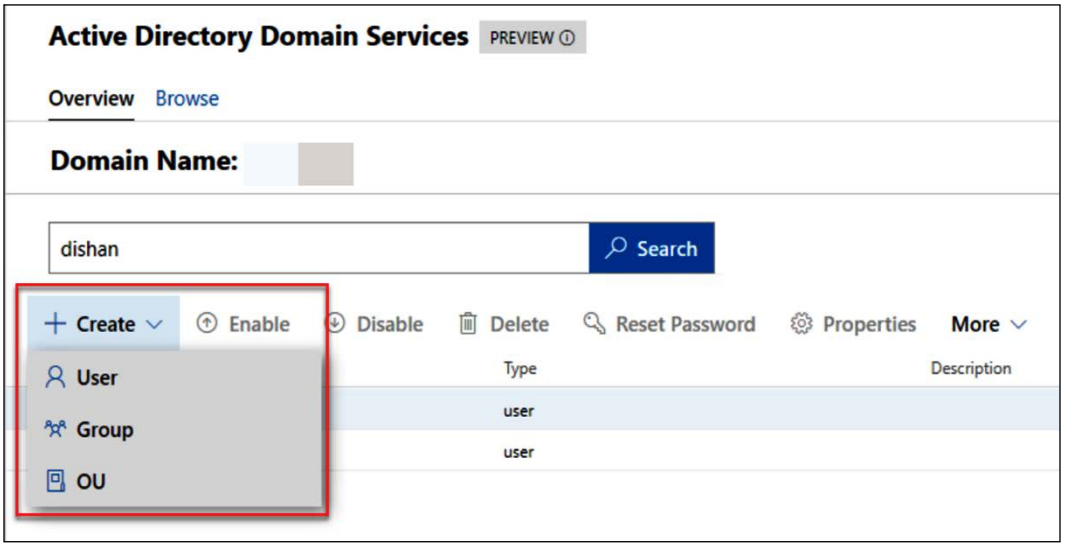


Figure 7.7: Creating an object using Windows Admin Center

If you are already in a hybrid environment, you can use Azure AD accounts to log in to Windows Admin Center. This helps to provide a unified SSO experience and to enable another layer of protection by using solutions such as Azure MFA. To do that, you need to register Windows Admin Center with Azure. More information about this process is available at `https://bit.ly/30S4XV7`.

Active Directory Administrative Center

The ADUC MMC is the most commonly used tool to manage AD environments. This tool has been built into the system since the early versions of AD and has continued right to the present. With AD DS 2008 R2, Microsoft introduced **Active Directory Administrative Center** (**ADAC**), which is built on top of PowerShell. It provides an enhanced GUI that can be used to manage AD objects in an efficient way. With AD DS 2012, Microsoft introduced the PowerShell History Viewer, which helps administrators to learn about the PowerShell commands associated with AD object management tasks. The ADAC console is used less compared to the ADUC MMC. This tool comes as part of the AD DS role and it doesn't require any additional configuration.

To access the ADAC console, you can type `dsac.exe` into PowerShell or the **Run** box:

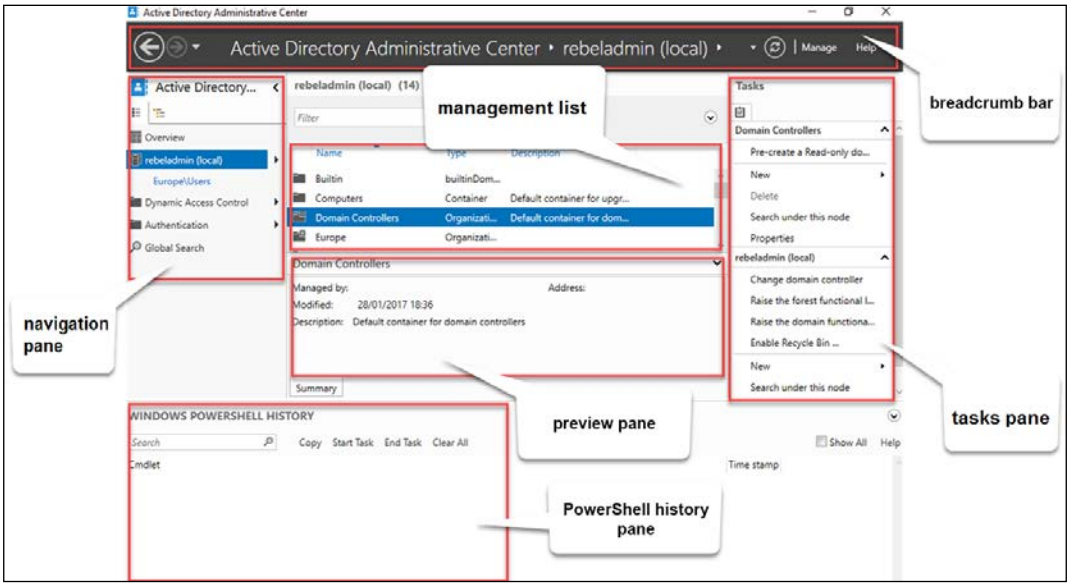


Figure 7.8: ADAC

The preceding screenshot shows the default interface for ADAC and its components, which can be used to manage AD objects:

- **Breadcrumb bar**: This can be used to navigate to different containers directly. In order to navigate to a specific container, you need to use its name. It can also be used the other way around, to find out the name of a container:

Figure 7.9: Breadcrumb bar in ADAC

  Using the **Manage** option allows us to add navigation nodes to the navigation pane. Basically, it's similar to adding a shortcut to specific containers.

- **Management list**: In this section, we can find a list of containers, objects contained in the containers, object search results, and more. The data display in this section will change based on the options selected in the navigation pane.

- **Preview pane**: This section shows a summary of the objects selected in the management list. The summary contains certain attribute values, such as the description, DNS name, and username, as well as the time the object was modified:

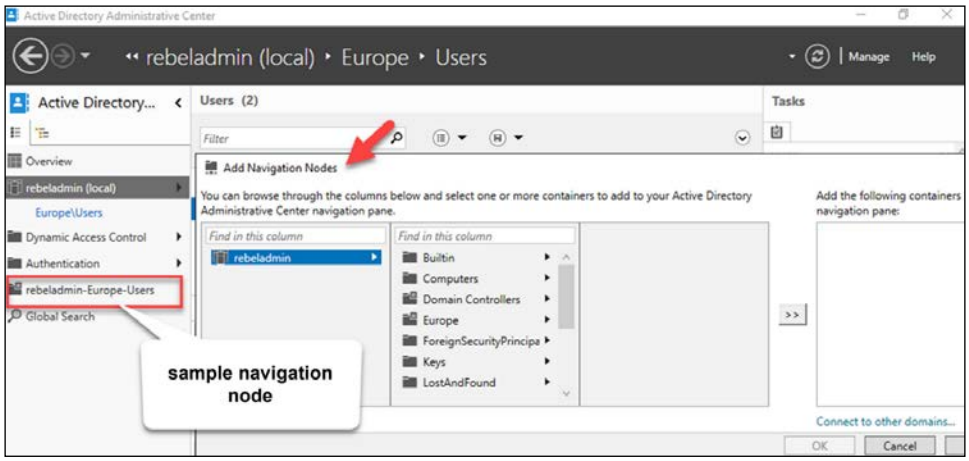


Figure 7.10: Navigation node in ADAC

- **Navigation pane**: This is similar to the navigation pane in the ADUC MMC. By using it, you can navigate to different containers in your domain. This can also be used to upgrade the domain and forest functional levels and enable the AD recycle bin. Using the navigation pane, we can add objects such as users, groups, OUs, and computers to the directory:

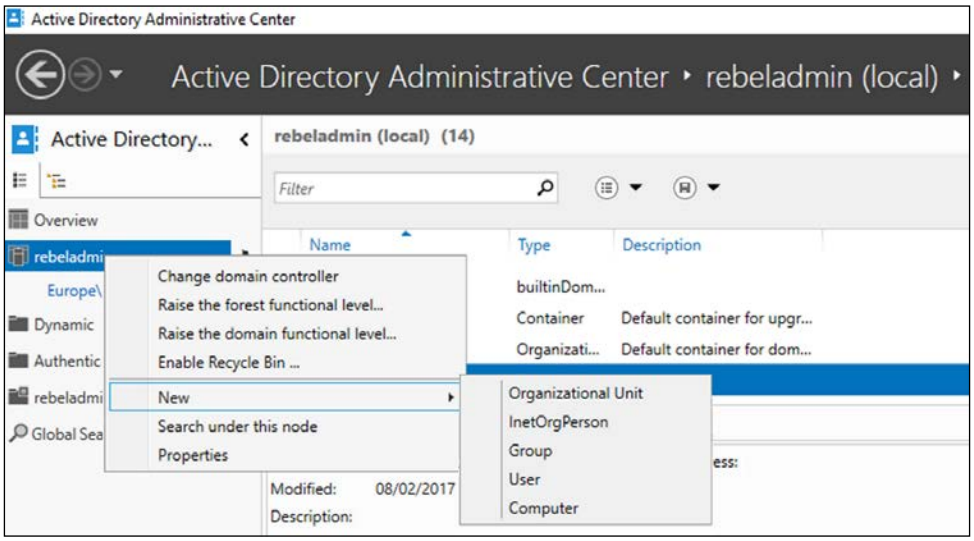


Figure 7.11: Adding objects in ADAC

The navigation pane also lists the **Global Search** option, which can be used to locate AD objects in the directory. Once the search returns an object, it also provides options to perform administrative tasks:

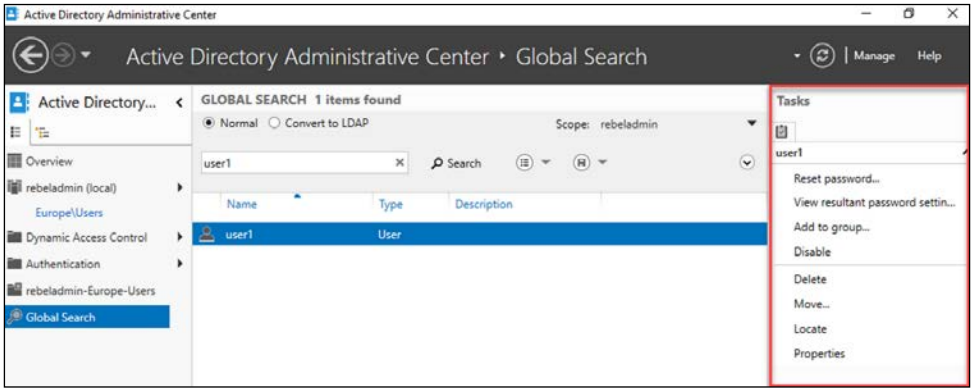


Figure 7.12: ADAC global search

- **Tasks pane**: The tasks pane lists the administrative tasks associated with the objects you select, such as moving objects, password resets, properties, and deletion. The list of administrative tasks will change based on the object type.
- **PowerShell history pane**: ADAC is built based on PowerShell command-line interface technology, so each and every task performed in ADAC is executed as a PowerShell command. In this pane, there is a list of all the executed PowerShell commands. Engineers can copy these commands and reuse or develop them further to manage AD objects via PowerShell directly. It also allows us to search for commands if required.

When you open ADAC for the first time, you will not see the PowerShell history pane in expanded mode, as shown in the following screenshot. You need to click on the **WINDOWS POWERSHELL HISTORY** bar to expand it:

Figure 7.13: ADAC PowerShell history

ADAC also allows us to manage objects from other domains. It can also be opened using **Server Manager | Tools | Active Directory Administrative Center**. If domains have one-way or two-way trust between them, it will allow us to add them to the same ADAC console. In order to do that, you need to go to the breadcrumb bar and click on **Manage | Add Navigation Nodes**, and then click on **Connect to other domains...** in the window:

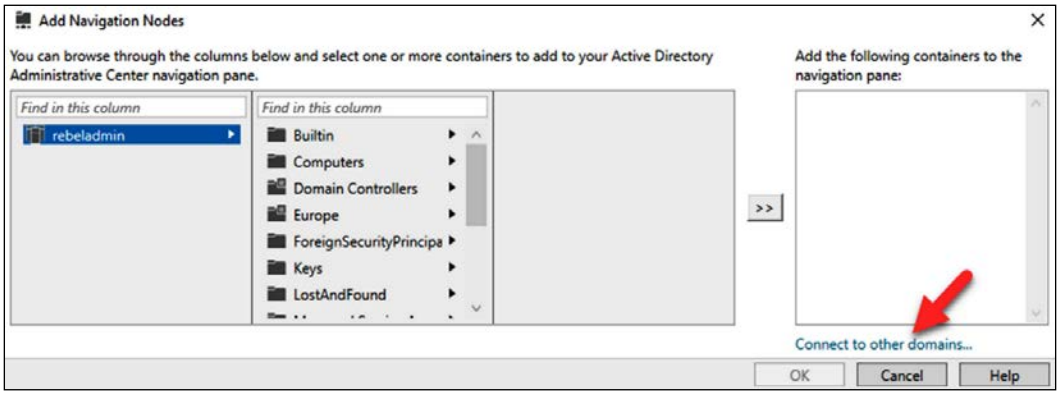


Figure 7.14: Connect to other nodes

Another advantage of ADAC is the advanced object property window. If you've used the ADUC MMC before, you may already know that in order to view an object's properties, we need to go through lots of different tabs. But with the ADAC advanced object properties window, we can view a lot of data in one window. If required, you can easily navigate to different sections.

Using the same window, you can run administrative tasks related to objects. Not only that, but it also allows us to modify the sections of the properties page as we like:

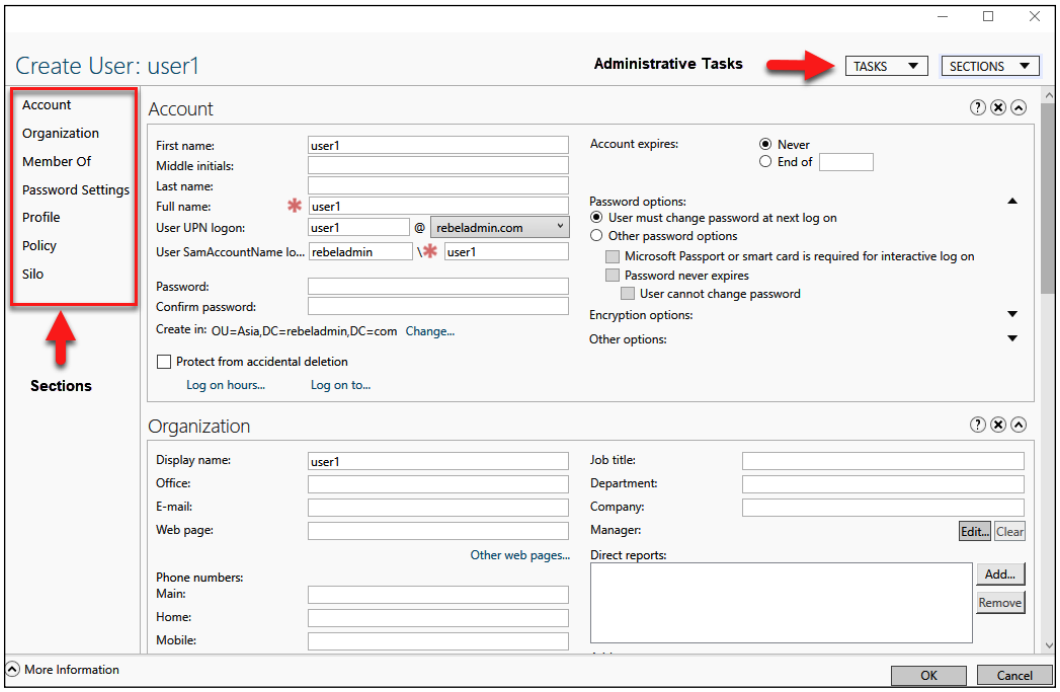


Figure 7.15: Administrative tasks

ADAC's capabilities can be summarized as follows:

- Creating users, groups, computer accounts, and OUs
- Managing users, groups, computer accounts, and OUs
- Removing users, groups, computer accounts, and OUs
- Managing AD objects from other trusted domains
- Filtering AD objects using queries

In the next section, we are going to look at the most commonly used tool to manage AD objects.

The ADUC MMC

The ADUC MMC is the most commonly used tool to manage AD objects. This tool was first available on AD DS 2000 and, over the years, hasn't changed much in terms of look and feel. This MMC also comes as part of the AD DS role. It is also part of **Remote Server Administration Tools** (**RSAT**), which can be installed on any computer.

We can open this tool by using `dsa.msc` in PowerShell or use the **Run** box from the **Start** menu:

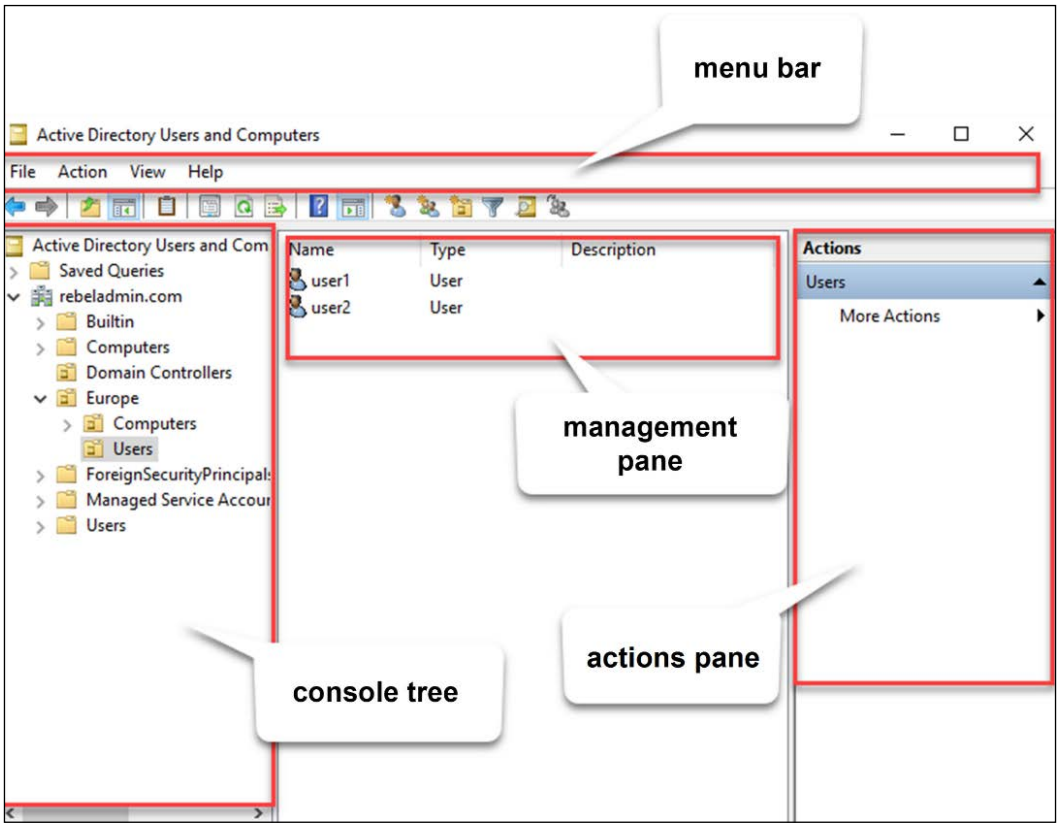


Figure 7.16: ADUC

Let's go through the main sections of the console:

- **Menu bar**: This contains menus with different options. Most of the options mentioned in the menus can also be executed using icons beneath the menu bar or **Actions** pane.

- **Console tree**: The console tree lists the structure of AD components and helps us to navigate through containers and find objects.
- **Management pane**: This displays the objects inside the selected container in the console tree. It can display different objects' types, such as **User**, **Group**, and **Device**. The content will change depending on the selected container.
- **Actions pane**: The **Actions** pane contains the administrative tasks related to selected AD objects. As an example, if a user object is selected, the **Actions** pane will list administrative tasks, such as moving the object, deleting it, resetting the password, and disabling the account.

We won't be looking at the ADUC MMC's functions too much here as it's the most commonly used AD management tool, but I am going to list some of the main features:

- **Advanced features**: By default, the MMC will not list all of the containers and object properties related to advanced system administration. In order to access these options, you need to enable them using **View | Advanced Features**.
- **Saved queries**: Using the MMC, we can create custom queries to filter AD objects and save these queries to rerun at a later time. This saves time as administrators do not need to spend time navigating through containers to find objects.

  To create a query, right-click on **Saved Queries** and select **New Query**. In this window, we can build a query using the **Define Query...** option:

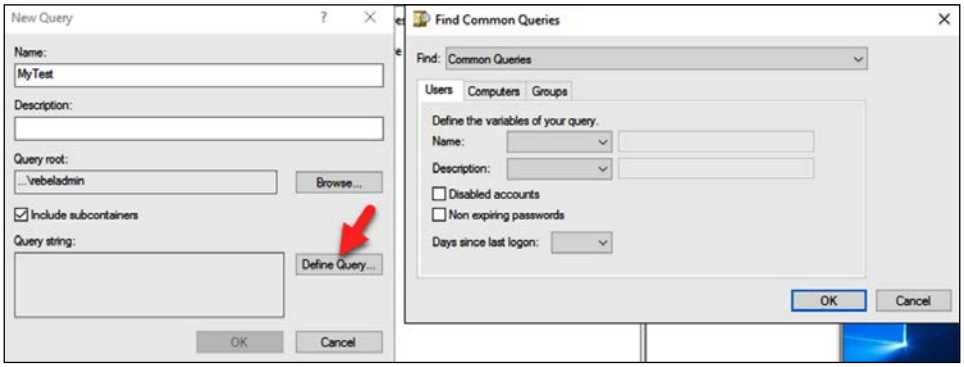


Figure 7.17: Working with queries

- **Access different domains**: If a domain has relationships of trust with other domains, the same console can be used to access them and manage the objects:

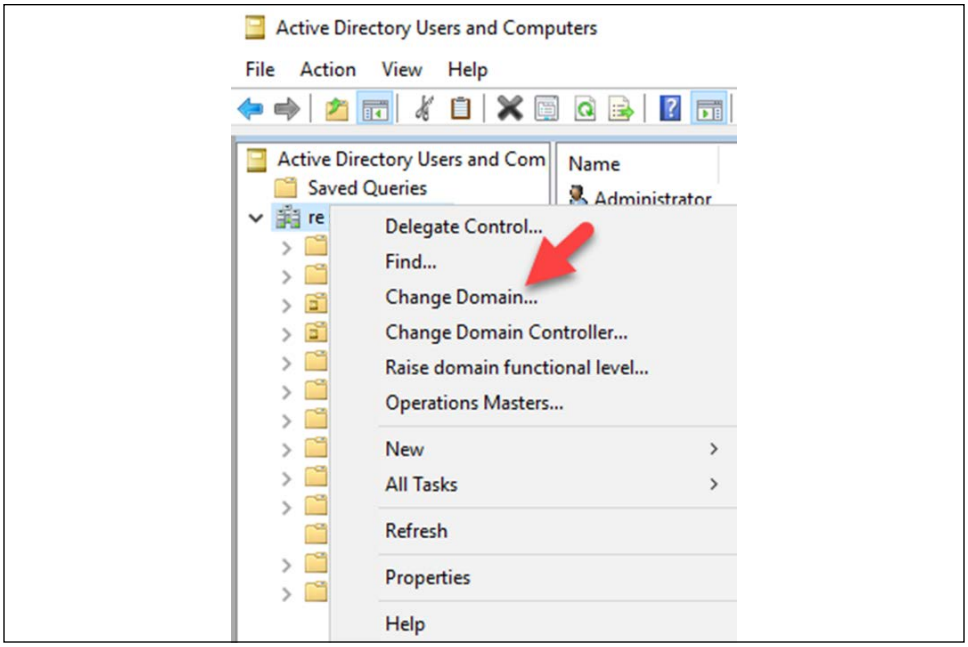


Figure 7.18: Accessing different domains

The capabilities of the ADUC MMC can be summarized as follows:

- Adding, editing, and removing users, groups, computers, and OUs
- Managing objects in different domains (needs two-way or one-way trust)
- Building queries to filter objects
- Searching for objects in directories
- Changing object properties

So far, we have mainly gone through the GUI tools that we can use to manage AD objects. We can do the same using PowerShell.

AD object administration with PowerShell

PowerShell provides more control over Windows' system functions and operations. PowerShell also simplifies the service/role configuration and management process. A single-line command can sometimes replace 10-15 clicks on a GUI. AD DS comes with the **AD PowerShell module**, which can be used to manage AD DS, **Active Directory Lightweight Directory Services** (**AD LDS**), and objects. AD objects can still be managed using Command Prompt, but PowerShell provides advanced, centralized control over AD components and services.

Any server that runs AD DS or AD LDS role services has the AD PowerShell module by default. It can also be enabled on a desktop computer or member server by installing RSAT.

If RSAT tools are installed on computers running PowerShell 2, you need to run `Import-Module ActiveDirectory` before using commands to manage AD.

Creating, modifying, and removing objects in AD

Creating, modifying, and removing objects are the most commonly performed management tasks in an AD environment. Using the different tools and methods described in the previous section, we can perform these tasks. Each tool and method has its own pros and cons. My recommendation is to use a mix of tools as not every tool is good for every administrative task.

Creating AD objects

Each and every object type has a different set of attributes. When you create objects, you need to provide values for those attributes. Some of these are mandatory and some are not. Based on the company's operations and preferences, some custom attributes may need to be added to the object types.

Creating user objects

In order to create a user object in AD, we can use the `New-ADUser` PowerShell cmdlet. You can view the full syntax of the command along with the accepted data types using the following command:

```
Get-Command New-ADUser -Syntax
```

In order to create a new user account using PowerShell, the minimum value you need to pass is `-Name`. It will create a disabled user account, and you can define values for other attributes later.

Here is an example that can be used to create a user account:

```
New-ADUser -Name "Talib Idris" -GivenName "Talib" -Surname "Idris"
-SamAccountName "tidris" -UserPrincipalName "tidris@rebeladmin.com"
-Path "OU=Users,OU=Europe,DC=rebeladmin,DC=com" -AccountPassword(Read-
Host -AsSecureString "Type Password for User") -Enabled $true
```

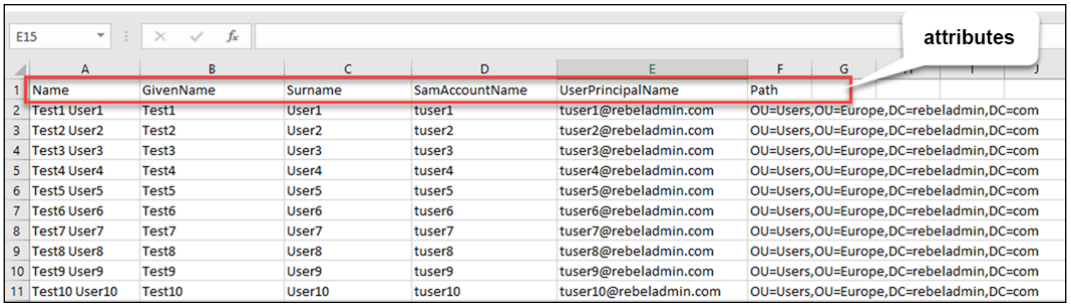
This command has the following parameters:

- `-Name`: This parameter defines the full name.
- `-GivenName`: This parameter defines the first name.
- `-Surname`: This parameter defines the surname.
- `-SamAccountName`: This parameter defines the username.
- `-UserPrincipalName`: This parameter defines the **User Principal Name** (**UPN**) for the user account.
- `-Path`: This defines the OU path. The default location is `CN=Users,DC=rebeladmin,DC=com`. If you do not define the `-Path` value, it will create the object under the default container.
- `-AccountPassword`: This will allow the user to input a password for the user, and the system will convert it into the relevant data type.
- `-Enabled`: This defines whether the user account status is enabled or disabled.

You can create a user account with the minimum attributes, such as a name and UPN. Later, you can define a password and enable the account. A user account cannot be enabled without a password. To define a password, you can use the `Set-ADAccountPassword -Identity` cmdlet, and to enable an account, you can use the `Enable-ADAccount -Identity` cmdlet.

Instead of executing multiple commands to create multiple user objects, we can create a **Comma-Separated Values** (**CSV**) file that includes data for attributes and use it to create multiple accounts in one go.

For demonstration purposes, I am using the following CSV file:



	A	B	C	D	E	F
1	Name	GivenName	Surname	SamAccountName	UserPrincipalName	Path
2	Test1 User1	Test1	User1	tuser1	tuser1@rebeladmin.com	OU=Users,OU=Europe,DC=rebeladmin,DC=com
3	Test2 User2	Test2	User2	tuser2	tuser2@rebeladmin.com	OU=Users,OU=Europe,DC=rebeladmin,DC=com
4	Test3 User3	Test3	User3	tuser3	tuser3@rebeladmin.com	OU=Users,OU=Europe,DC=rebeladmin,DC=com
5	Test4 User4	Test4	User4	tuser4	tuser4@rebeladmin.com	OU=Users,OU=Europe,DC=rebeladmin,DC=com
6	Test5 User5	Test5	User5	tuser5	tuser5@rebeladmin.com	OU=Users,OU=Europe,DC=rebeladmin,DC=com
7	Test6 User6	Test6	User6	tuser6	tuser6@rebeladmin.com	OU=Users,OU=Europe,DC=rebeladmin,DC=com
8	Test7 User7	Test7	User7	tuser7	tuser7@rebeladmin.com	OU=Users,OU=Europe,DC=rebeladmin,DC=com
9	Test8 User8	Test8	User8	tuser8	tuser8@rebeladmin.com	OU=Users,OU=Europe,DC=rebeladmin,DC=com
10	Test9 User9	Test9	User9	tuser9	tuser9@rebeladmin.com	OU=Users,OU=Europe,DC=rebeladmin,DC=com
11	Test10 User10	Test10	User10	tuser10	tuser10@rebeladmin.com	OU=Users,OU=Europe,DC=rebeladmin,DC=com

Figure 7.19: CSV file with user attribute values

I have used data for some of the attributes via the CSV file, and some common values will be passed through the following script:

```
Import-Csv "C:\ADUsers.csv" | ForEach-Object {
$upn = $_.SamAccountName + "@rebeladmin.com"
New-ADUser -Name $_.Name `
 -GivenName $_."GivenName" `
 -Surname $_."Surname" `
 -SamAccountName $_."samAccountName" `
 -UserPrincipalName $upn `
 -Path $_."Path" `
 -AccountPassword (ConvertTo-SecureString "Pa$$w0rd" -AsPlainText
-force) -Enabled $true
}
```

In this script, the `Import-Csv` cmdlet is used to import the CSV file that I created in the previous step. I also defined the UPN value, `-UserPrincipalName`, using `$upn = $_.SamAccountName + "@rebeladmin.com"`. At the end, I defined a common password for all of the accounts using `-AccountPassword (ConvertTo-SecureString "Toronto@1234" -AsPlainText -force)`.

By following this section, now we know how to create user objects using PowerShell. In the next section, we are going to learn how to create computer objects in AD using PowerShell and GUI tools.

Creating computer objects

When a desktop computer or member server is joined to a domain, it will create a computer object in AD.

This computer object can be created before being added to the domain. This will not add the device to the domain, but it can be used with offline domain joins and RODC domain joins.

In order to create a computer object, we can use the `New-ADComputer` cmdlet. To view the complete syntax of the command, use this:

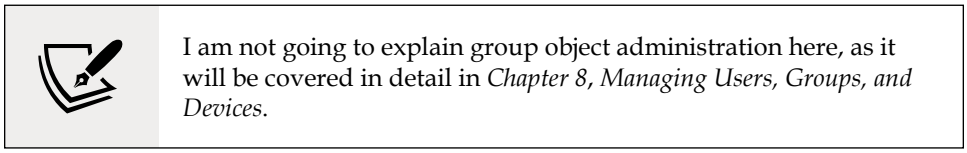
```
Get-Command New-ADComputer -Syntax
```

The attribute you need to create a computer object is `-Name`:

```
New-ADComputer -Name "REBEL-PC-01" -SamAccountName "REBEL-PC-01" -Path
"OU=Computers,OU=Europe,DC=rebeladmin,DC=com"
```

In the preceding example, the command will create the `REBEL-PC01` computer object in the `OU=Computers,OU=Europe,DC=rebeladmin,DC=com` OU. If you do not define the path, it will create the object under the default computer container, `CN=Computers, DC=rebeladmin, DC=com`.

We very rarely need mass computer-object creation in an organization. If it's required, we can do it using a CSV method similar to user-object creation.



I am not going to explain group object administration here, as it will be covered in detail in *Chapter 8*, *Managing Users, Groups, and Devices*.

User and **computer** objects can also be created using ADAC or ADUC. The following figure shows the new user creation window in ADAC:

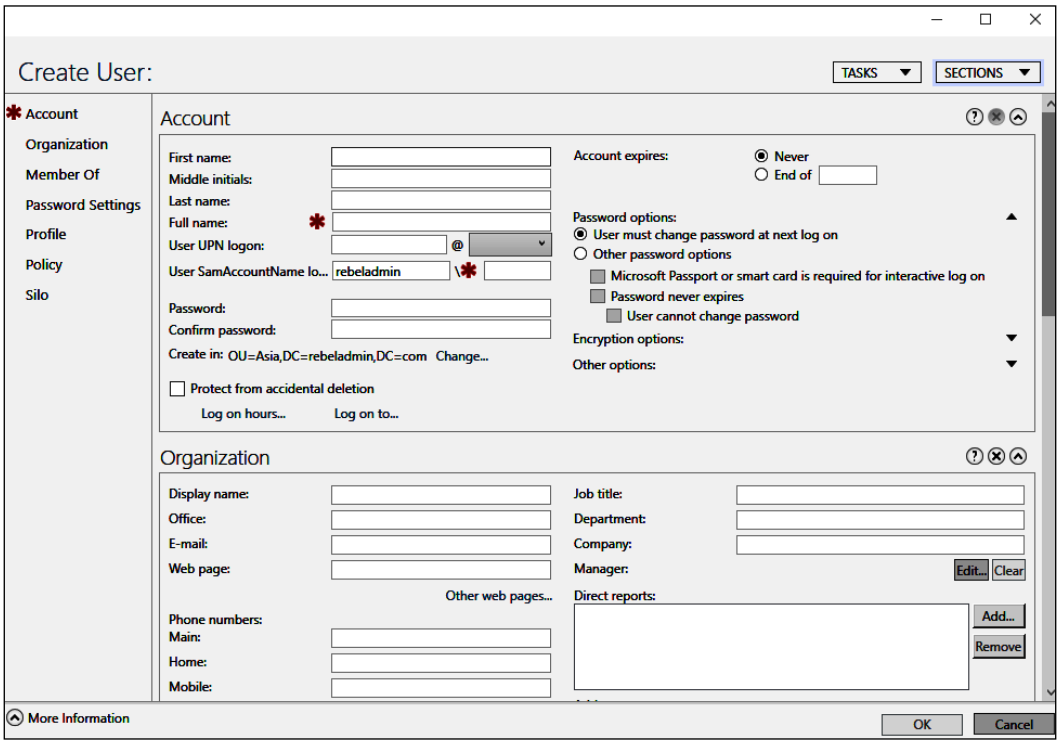


Figure 7.20: Creating a new user by using ADAC

The following screenshot shows the wizard to create a **computer** object using ADUC:

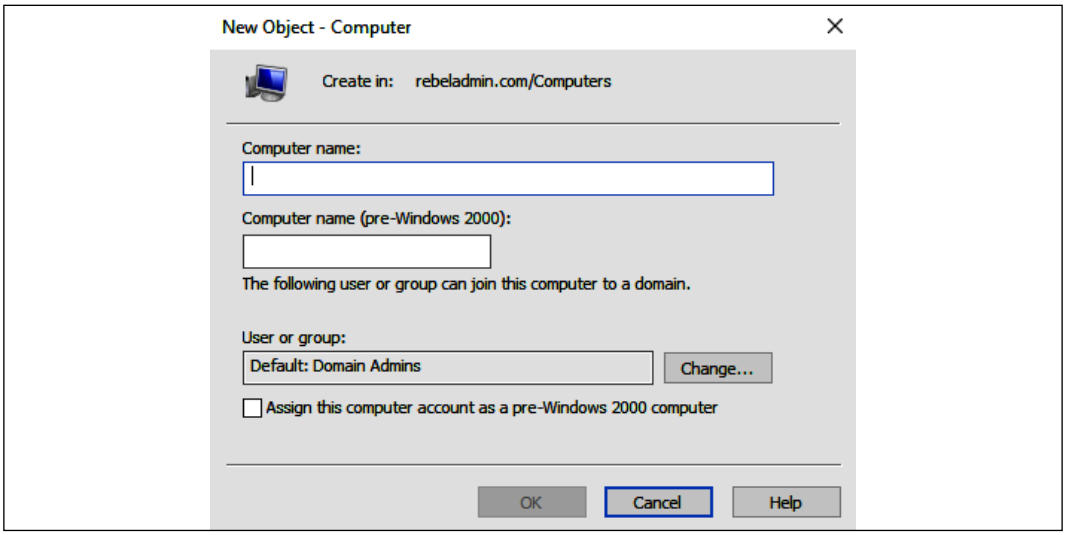


Figure 7.21: Creating a computer object

In this section, we learned how to create computer objects using different tools (ADAC, ADUC, and PowerShell). When it comes to AD object management, we also have to modify object attributes from time to time. In the next section, we are going to look into AD object modification using PowerShell.

Modifying AD objects

When we create objects, we define values for attributes. After we've created the objects, there may be situations where we need to edit the values of those attributes or add values to empty attributes.

We can use the `Set-ADUser` cmdlet to change and add attribute values to existing AD user objects:

```
Set-ADUser tidris -OfficePhone "0912291120" -City "London"
```

In the preceding sample command, we're adding values for the `-OfficePhone` and `-City` attributes for the `tidris` user.

There are occasions where you may need to change the existing value of an attribute:

```
Set-ADUser tidris -OfficePhone "0112291120"
```

In the preceding command, I'm replacing an existing value with a new one.

In the aforementioned commands, I defined the exact user account, but it's not practical if you need to do this for a large number of accounts. To do that, we need to combine the `Set-ADUser` cmdlet with the `Get-ADuser` cmdlet. This will allow us to search for objects first and then push the changes:

```
Get-ADUser -Filter * -SearchBase 'OU=Users,OU=Europe,DC=rebeladmin,DC=c
om' | Set-ADUser -City "London"
```

In the preceding command, we search for all of the user objects located in `OU=Users,OU=Europe,DC=rebeladmin,DC=com` and set the `City` value to `London`:

```
Get-ADUser -Filter {City -like "London"} | Set-ADUser -City "Kingston"
```

In the preceding example, I searched for all of the users in the directory who have the `City` value defined as `London` and changed it to `Kingston`.

Combining a search query with object modification saves us a lot of manual work, and it's not something we can do easily using other GUI tools.

Computer object values can also be added/changed using a similar method. In order to do so, we need to use the `Set-ADComputer` cmdlet:

```
Set-ADComputer REBEL-PC-01 -Description "Sales Computer"
```

The preceding command sets the `Description` object value of the computer named `REBEL-PC-01`.

This cmdlet can also be combined with a search query using the `Get-ADComputer` cmdlet:

```
Get-ADComputer -Filter {Name -like "REBEL-PC-*"} | Set-ADComputer
-Location "M35 Building"
```

In the preceding command, it searches for computers with the name `REBEL-PC` and sets the location value to `M35 Building`.

In the ADAC and ADUC GUI tools, it's just a matter of double-clicking on an AD object and then editing the attribute values. If required, we can also select multiple objects and edit particular attribute values in one go.

We can't change unique attribute values with multiple selection. As an example, we can't change the user names for multiple accounts. However, we can change the values for common attributes, such as City and Department, with multiple object selection.

Removing AD objects

In order to remove AD user objects, we can use the `Remove-ADUser` cmdlet. We can find the complete syntax information using the following command:

```
Get-Command Remove-ADUser -Syntax
```

When using the cmdlet, we need to use a value for the `-Identity` parameter to specify the account. We can use a distinguished name, GUID, SID, or the `SamAccountName` value to identify the account. If it is an LDS environment, we need to define the object partition parameter too:

```
Remove-ADUser -Identity "dzhang"
```

The preceding command will remove the AD user object called `dzhang` from the directory. It will ask for confirmation before it removes the object.

This cmdlet can also be combined with the search query to find objects before removing them:

```
Get-ADUser -Filter {Name -like "Test1*"} | Remove-ADUser
```

In the preceding command, we search the entire directory for the user whose name starts with `Test1` and then remove that user.

The `Remove-ADComputer` cmdlet can be used to remove computer objects from the directory:

```
Remove-ADComputer -Identity "REBEL-PC-01"
```

The preceding command will remove the `REBEL-PC-01` computer object from the directory. We can also combine it with a search query:

```
Get-ADComputer -Filter * -SearchBase 'OU=Computers,OU=Europe,DC=rebelad
min,DC=com' | Remove-ADComputer
```

In the preceding command, we search for the computer objects in the given OU and then remove the findings from the directory.

ADAC and ADUC tools can also be used to remove objects from a directory. Both tools have a search function that can be used to locate specific objects and then delete them.

Finding objects in AD

The AD NTDS database can store more than two billion objects. So how easily can we locate a specific AD object? The most common way to locate an object in AD is to use ADAC or ADUC and browse through the containers. As the number of objects increases, so does the difficulty of locating objects in AD. In this section, we are going to look at more efficient ways of locating objects in an AD environment.

ADAC has enhanced query and filter capabilities. Since it's used via a GUI, it helps us to retrieve results faster compared to PowerShell.

In its management list, there is a filter box in the top section (once you've clicked on the domain name in the navigation pane). It doesn't change the view as you navigate through the containers. It helps you to filter the data displayed in the management list quickly. However, it doesn't search for objects at multiple levels (in different child containers):

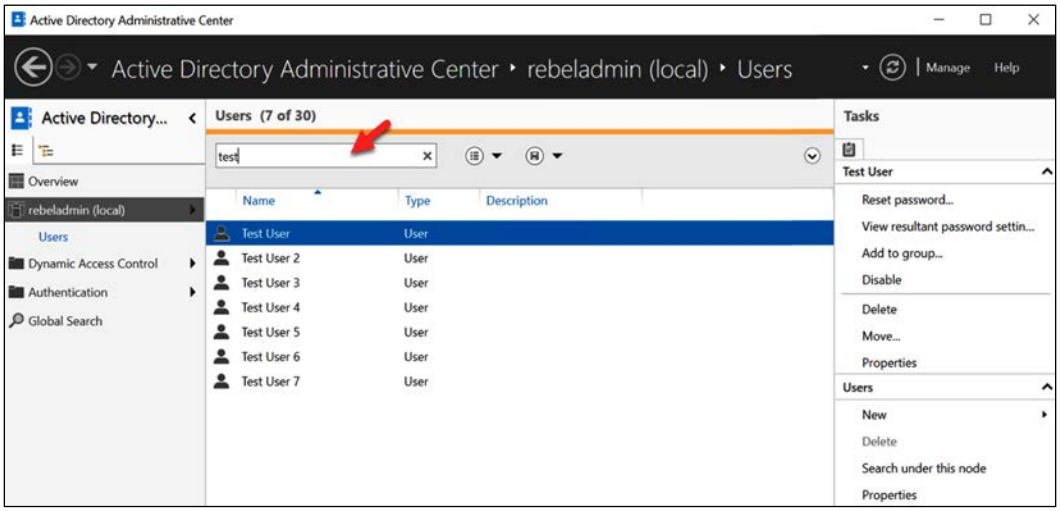


Figure 7.22: Searching for AD objects

ADAC has a feature called **Global Search**, which can be used to search for objects in the entire directory. This allows a typical text-based search or advanced **Lightweight Directory Access Protocol** (**LDAP**)-based queries:

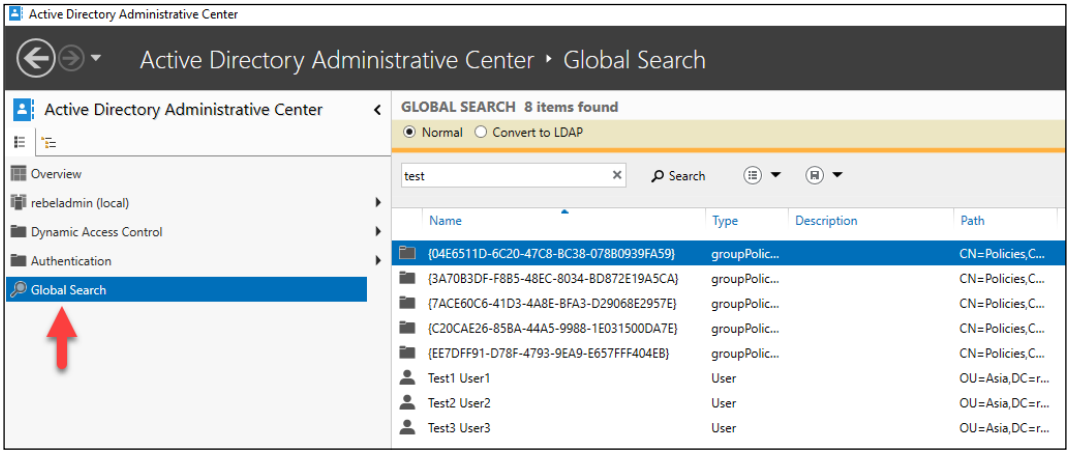


Figure 7.23: Global Search in ADAC

Let's now look at how to perform LDAP-based searches. The method also allows us to define the search scope:

Figure 7.24: LDAP query-based search

In a *normal* search, we can use any text pattern. This can be a complete or partial value. In an LDAP search, we need to use exact syntax. Using the **Convert to LDAP** option, we can generate an LDAP query from a *normal* search.

This **Global Search** function can also be accessed from the **Overview** page.

In ADUC, object search functions are very limited compared to ADAC and PowerShell. It does have the **Find...** option, which allows us to locate objects in the directory.

It can be accessed by right-clicking on any container:

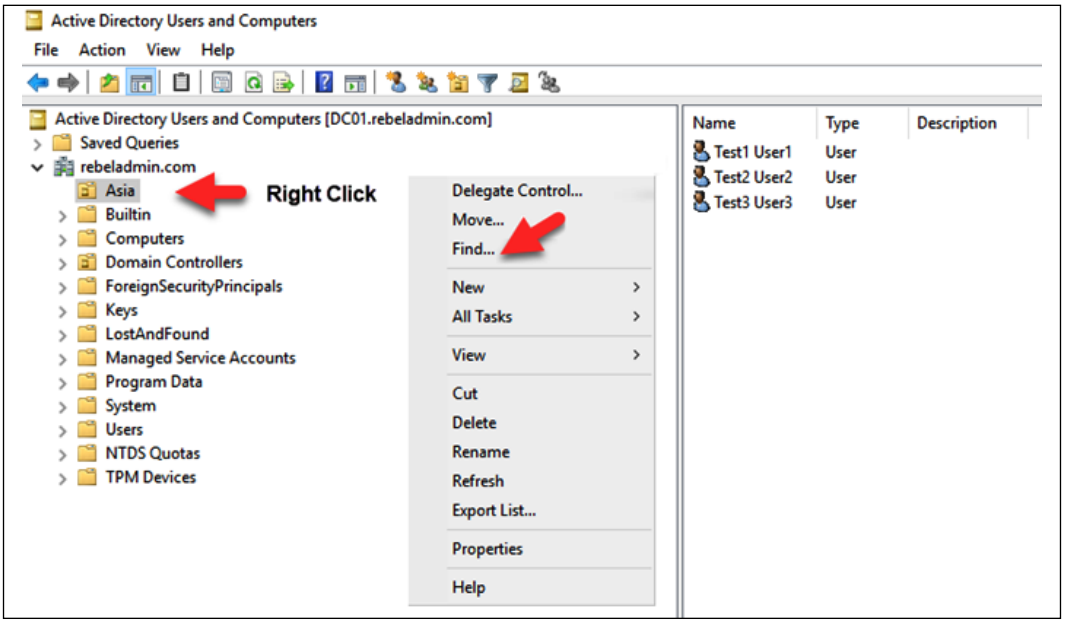


Figure 7.25: Finding objects using ADUC

It allows us to do a text-based search as well as an advanced search based on attributes and their values. It also allows us to define the search scope:

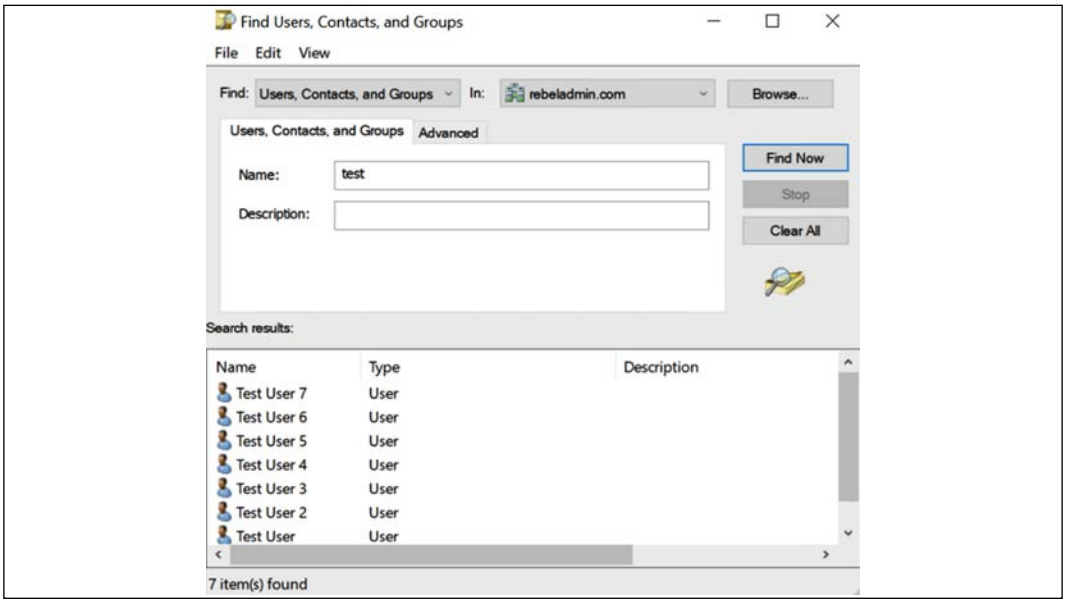


Figure 7.26: Search results

The ADUC and ADAC tools are the easiest way to search for an object in AD. The PowerShell method not only allows us to search objects but also combines searching with additional tasks such as the CSV file export of search results, object value modification, and moving and removing AD objects. In the next section, we are going to learn how to find objects using PowerShell.

Finding objects using PowerShell

In the previous section, we learned about the `Get-ADUser` and `Get-ADComputer` cmdlets and how they can be used with other commands to filter out objects from AD. They can also be used to retrieve specific attribute values from filtered objects:

```
Get-ADUser -Identity user1 -Properties *
```

The preceding command will list all of the attributes and values associated with `user1`. This helps us to find the exact attribute names and common values, which can be used for further filtering.

I need to know the values for `Name`, `UserPrincipalName`, and `Modified` for all of the users. The following command will create a table with relevant attributes and their values:

```
Get-ADUser -Filter * -Properties Name,UserPrincipalName,Modified | ft
Name,UserPrincipalName,Modified
```

I can see some accounts in the list that are service and administrator accounts. I only want to see the accounts in the `Kingston` office:

```
Get-ADUser -Filter {City -like "Kingston"} -Properties
Name,UserPrincipalName,Modified | ft Name,UserPrincipalName,Modified
```

The preceding command filters users further based on the `City` value.

Now I have the list of data I need, and I'd like to export it to a CSV file for future use, like so:

```
Get-ADUser -Filter {City -like "Kingston"} -Properties
Name,UserPrincipalName,Modified | select-object
Name,UserPrincipalName,Modified | Export-csv -path C:\ADUSerList.csv
```

This example demonstrates how a search query can be built up to gather the required information.

The `Search-ADAccount` cmdlet can also be used to search for AD objects based on the account and password status. The full syntax of the cmdlet can be retrieved using the following command:

```
Get-Command Search-ADAccount -Syntax
```

As an example, it can be used to filter accounts that are locked out:

```
Search-ADAccount -LockedOut | FT Name,UserPrincipalName
```

This command will list all of the locked-out accounts with a name and UPN.

Unlike the graphical tools, PowerShell queries can be built to filter exact objects and data from AD.

Preventing the accidental deletion of objects

When working with AD objects, it is possible to delete an AD object accidentally. When an AD object is deleted accidentally, the impact on the business will depend on the AD object's role. As an example, if a service account for a critical service is deleted, the business impact will be higher than for the deletion of a test user account. With AD DS 2008, Microsoft introduced a small but important feature to prevent accidental AD object deletion. This is not a solution to recover from disasters but a solution to prevent disasters. In every AD object, under the **Object** tab, there is a small checkbox to enable this feature. This can be enabled when we create objects using PowerShell. Even if we're not using PowerShell, it can still be enabled using the **Object properties** window at any time. When creating an OU, this feature is enabled by default:

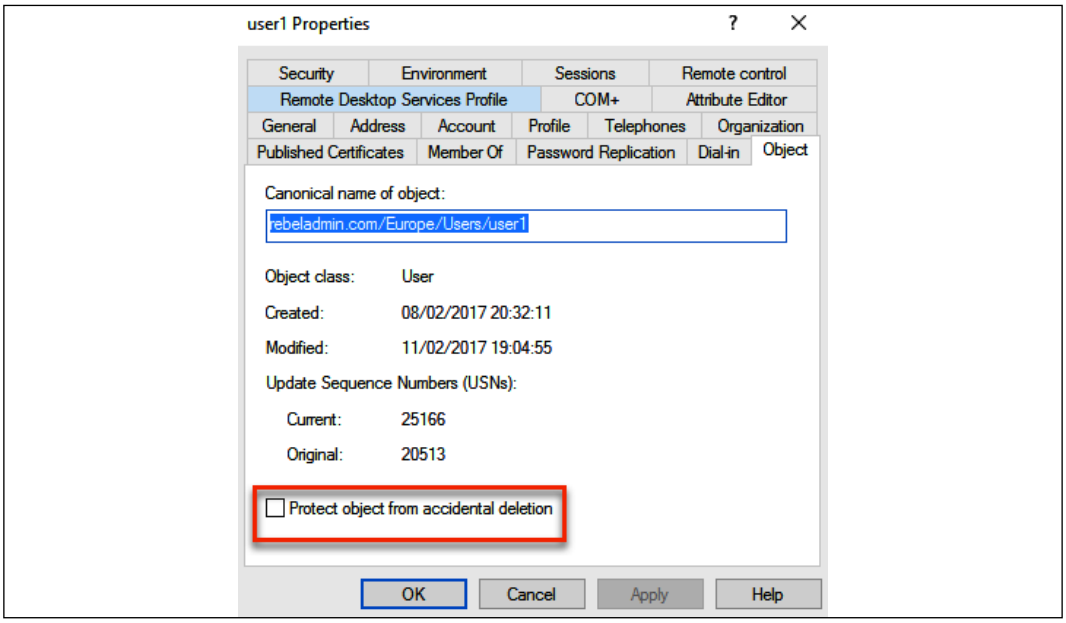


Figure 7.27: Protect object from accidental deletion

When this option is enabled, it will not allow you to delete an object unless you disable the option:

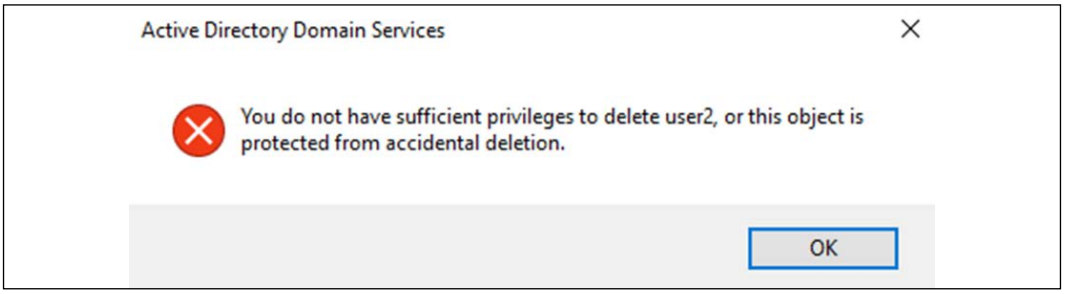


Figure 7.28: Protect object from accidental deletion – error message

In PowerShell, this can be done using the `-ProtectedFromAccidentalDeletion $true` parameter.

As an example, let's look at how we can enable this feature for a specific user account:

```
Set-ADObject -Identity 'CN=Dishan Francis,DC=rebeladmin,DC=com'
-ProtectedFromAccidentalDeletion $true
```

In the preceding command, I am enabling the **Protect object from accidental deletion** feature for user account Dishan Francis.

When we create OUs, this feature will be enabled by default.

AD recycle bin

When an object is deleted from AD, it is not permanently deleted. As soon as an object is deleted, the system will set the `isDeleted` attribute value to `True` and move the object to `CN=Deleted Objects`:

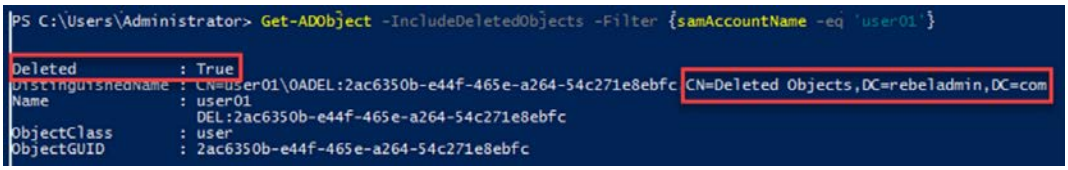


Figure 7.29: Deleted AD object

Then, the deleted object will stay there until the system reaches the *tombstone lifetime* value. By default, this is 180 days, and it can be changed if required. As soon as the object passes the tombstone lifetime value, it can be permanently deleted.

An essential function of an AD database is online defragmentation. This process uses the garbage collector service to remove deleted objects from the AD database and release space back to the database. This service runs every 12 hours. Once the deleted object exceeds the tombstone lifetime value, the object will be permanently removed in the next garbage collector service cycle. The problem with this is that during the tombstone process, most of the object values are stripped off. So, even if you were able to recover objects, attribute values would need to be re-entered.

With Windows Server 2008 R2, Microsoft introduced the *Active Directory Recycle Bin* feature. When this feature is enabled, when an object is deleted, the system still sets the `isDeleted` attribute value to `True` and moves the object to `CN=Deleted Object`. But, instead of the tombstone lifetime value, it's now controlled by the **Deleted Object Lifetime** (**DOL**) value. Object attributes will remain the same at this stage, and they are easily recoverable. By default, the DOL value is equal to the tombstone lifetime value. This value can be changed by modifying the `msDS-deletedObjectLifetime` attribute value**.** When a deleted object's DOL value is reached, the system will set the object to the `Recycled` state and the `isRecycled` attribute value is set to `True`. If an object is in this state, we can't recover it, and the object will remain in this state until the tombstone lifetime value is exceeded. When the tombstone lifetime value is exceeded, the system will delete the object permanently from AD.

The AD Recycle Bin feature requires a minimum of a Windows Server 2008 R2 domain and a forest functional level. Once this feature is enabled, it cannot be disabled.

This feature can be enabled using the following command:

```
Enable-ADOptionalFeature 'Recycle Bin Feature' -Scope
ForestOrConfigurationSet -Target rebeladmin.com
```

In the preceding command, `-Target` can be replaced with your domain name.

Once `Recycle Bin Feature` is enabled, we can restore objects, which are deleted using the following command:

```
Get-ADObject -filter 'isdeleted -eq $true' -includeDeletedObjects
```

The preceding command searches for objects where the `isdeleted` attribute is set to `true`.

Once we have found a deleted object, it can be restored using the following command:

```
Get-ADObject -Filter 'samaccountname -eq "dfrancis"'
-IncludeDeletedObjects | Restore-ADObject
```

The preceding command restores the user object `dfrancis`.

Summary

There are lots of different tools out there for managing AD objects. In this chapter, we looked at the tools built by Microsoft to manage AD objects. Each and every tool has different characteristics, and we learned how we can use them to add, edit, and remove AD objects effectively. We also learned how we can use different tools and technologies to search for specific AD objects and attribute values. Last but not least, we looked at features that we can use to prevent the accidental deletion of AD objects.

In the next chapter, we will dive deep into AD objects and attributes, evaluating different types of objects and their roles in an AD environment.

8

Managing Users, Groups, and Devices

In the previous chapter, we learned about **Active Directory** (**AD**) object types and how we can add, edit, and remove them. We also learned about the different management tools that help us to do these tasks. Last but not least, we learned how we can locate Active Directory objects or the value of an attribute when required. In this chapter, we are going to further explore Active Directory objects and attributes.

The characteristics of an Active Directory object are described using attributes. Some of these attributes are common across different types of objects and some are unique. If required, we can also add our own attributes to an object. In this chapter, you will learn about object attributes and how we can manage them. You will also learn how to add custom attributes. If an organization is using custom attributes in a hybrid environment, it is possible that custom attributes will also need to sync to Azure AD. In this chapter, I will demonstrate how we can sync custom attributes from on-prem Active Directory to Azure AD. In an Active Directory environment, user objects usually represent people. But we also use user objects to represent service accounts. Microsoft recommends that you use **Managed Service Accounts** (**MSAs**) and **Group Managed Service Accounts** (**gMSAs**) for services and applications instead of typical user accounts. In this chapter, you will learn about the characteristics of these different service account types and how to use them. In an Active Directory environment, we group objects together using Active Directory groups. These groups have different categories. In this chapter, you will learn about these different types of Active Directory groups and how to use them effectively.

Below, I have summarized the topics that I will cover in this chapter:

- Object attributes
- Creating custom attributes
- Syncing custom attributes to Azure AD
- Different types of user accounts and their roles
- Different types of groups and their roles
- Different types of devices and other objects that can be managed via AD
- Object management best practices

Without attributes, we can't describe an Active Directory object. So let's go ahead and start the chapter by looking at attributes.

Object attributes

My daughter, Selena, loved Julia Donaldson's books. I usually read her a story every night. Some time ago, I was reading her one of the Gruffalo series books called *The Gruffalo's Child*. In that book, the Gruffalo's child asks about the big bad mouse who lives in the snowy forest. The Gruffalo describes the mouse, saying he is strong, his eyes are big, his tail is very long, and he has got whiskers thicker than wires. Then, the Gruffalo's child goes out to find this mouse on a snowy night.

During his journey, he finds animals that match one or a few of the characteristics that his father had described, but none matches all of them. In the end, only a shadow of a small mouse matches the characteristics of the animal he was looking for, not a real living creature. Transposing this idea to the world of objects, we can say the mouse is an object. The Gruffalo describes it to his kid using characteristics, which are similar to the attributes of an object. When the Gruffalo's kid goes to find it, he also finds other animals that have similar characteristics. Similarly, objects can have attributes that apply to other objects in the same class. In the end, he finds the best match for all the characteristics because some of them didn't match with the characteristics of any other animal. Similarly, some attributes must be unique, which makes objects unique in the same object class.

In the following screenshot, I have opened a user object via the **Active Directory Users and Computers** (**ADUC**) **Microsoft Management Console** (**MMC**).

In there, under **Attribute Editor**, we can see all the attributes associated with this object, including the values:

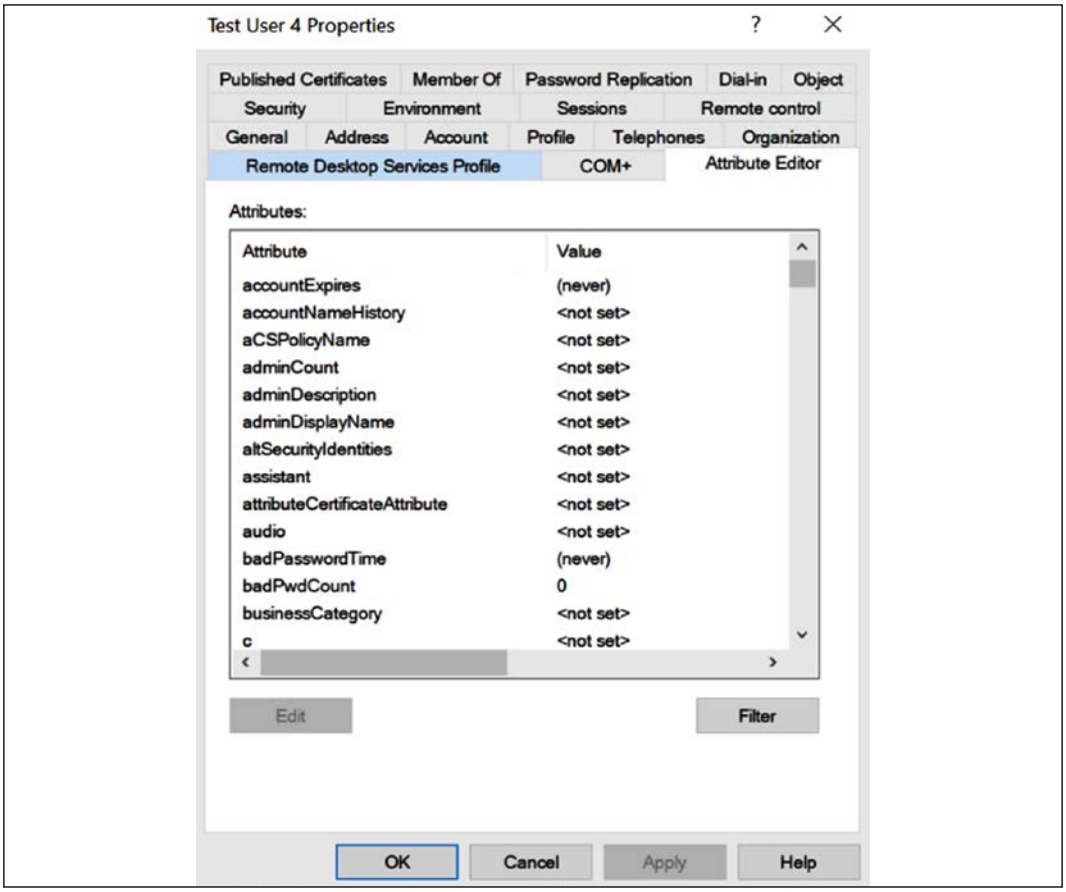


Figure 8.1: Attribute Editor

To view **Attribute Editor**, we need to click on **View | Advanced Features** in ADUC.

Using this window, we can add, edit, or remove values from some attributes. Most of these attribute names do not match with the names in the wizard you get when you create the object. As an example, the `givenName` attribute (the **Lightweight Directory Access Protocol** or **LDAP** name) maps to the first name (the display name) in the user account creation wizard.

All the user accounts in AD will have the same set of attributes. This is defined by a class in the AD schema:

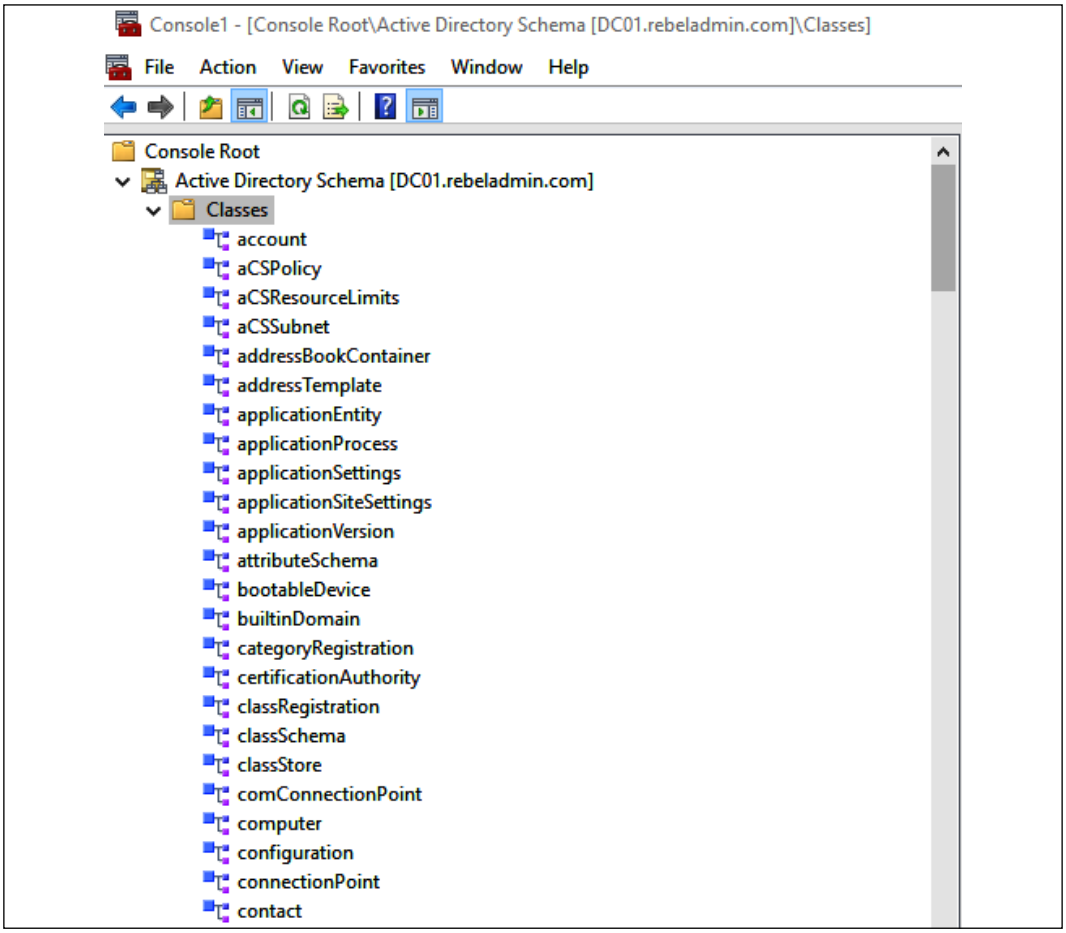


Figure 8.2: AD schema snap-in

To open the AD schema snap-in, you need to run the `regsvr32 schmmgmt.dll` command from the domain controller. After that, you can use MMC and add the AD schema as a snap-in.

In the preceding screenshot, I have opened an Active Directory schema snap-in and have opened the **user** class. To open a snap-in, go to **Run | MMC | File | Add/ Remove Snap-in**, and then select **Active Directory Schema** from the list. After that, click **Add...** and then click **OK** to complete the wizard.

In the object properties window, we can see the list of attributes that associate with objects under the **user** class. The same attributes can be part of different classes, too.

As an example, the **mail** attribute is part of the **user** and **group** classes:

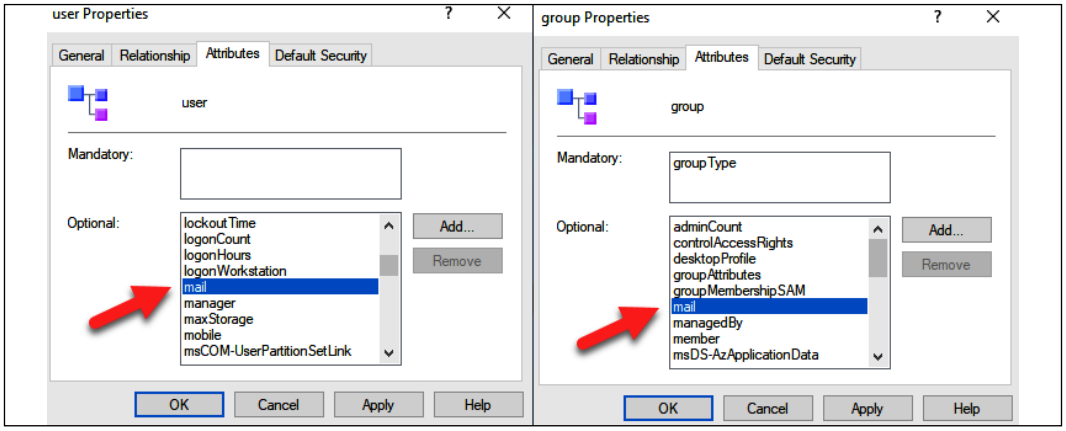


Figure 8.3: Attributes for user and group classes

We can review the attribute details by opening attributes from the `Attributes` container:

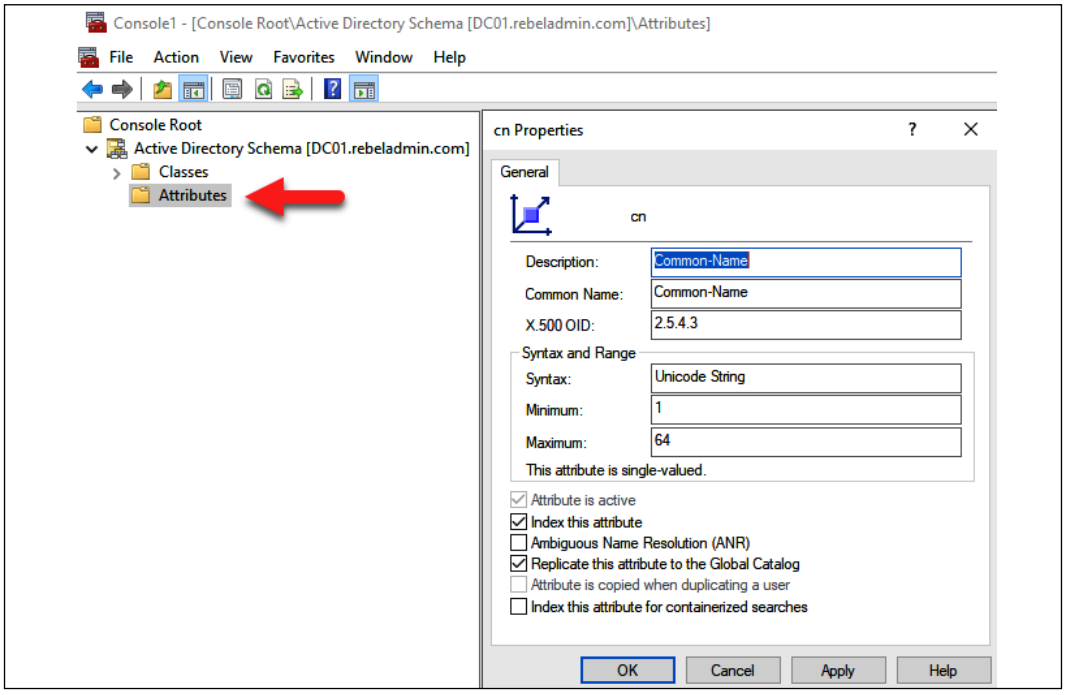


Figure 8.4: Attributes details

In the preceding screenshot, I opened the **cn** attribute, and in the properties window, it shows details such as **Common Name**, **Syntax**, and the value type it accepts.

Custom attributes

The Active Directory schema accepts custom attributes. Based on business requirements, sometimes organizations will have to introduce custom attributes to object classes. Most of the time, it is related to application integration requirements with Active Directory. Some applications have their own way of handling their user accounts and privileges. These applications can also have their own attributes defined by their database systems to store data. Sometimes, these application attributes may not match the attributes on Active Directory.

Some time ago, I was helping a pharmaceutical company on an Active Directory project. The customer already had an Active Directory environment in place. They were also maintaining an HR system that was not integrated with Active Directory. They had a new requirement for an employee collaboration application, which required data to be input in a specific way. It had defined fields in the database and we needed to match the data in the order that the fields dictated. Some of these fields required data about users that could be retrieved from Active Directory and some of the user data could be retrieved from the HR system. Instead of using two data feeds, we decided to treat Active Directory as the trustworthy data source for this new system. If Active Directory needed to hold all the required data, we also needed to store the additional data from the HR system. The solution was to add custom attributes to the Active Directory schema and associate it with the user class. Instead of both systems operating as data feeds, now the HR system could pass the filtered values to Active Directory, which would export all the required data in CSV format to the application.

In order to create custom attributes, go to the **Active Directory Schema** snap-in, right-click on the `Attributes` container, and select the **Create Attribute...** option.

Then, the system will give a warning about schema object creation. Click **OK** to continue and the following screen will open:

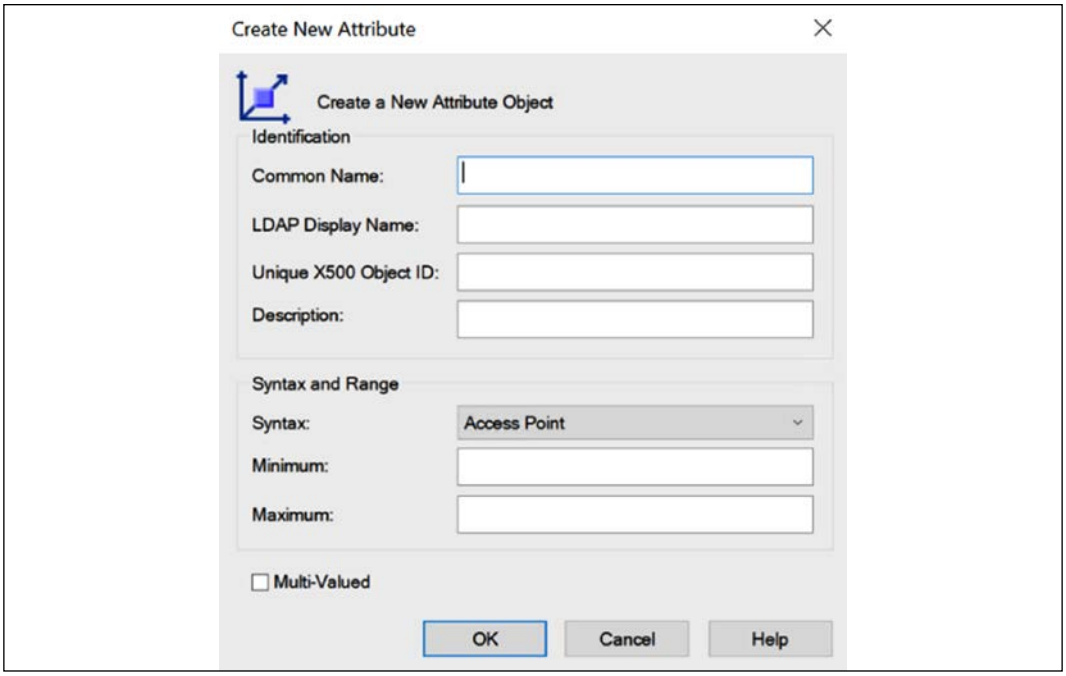


Figure 8.5: Create New Attribute

As shown in the preceding screenshot, a form will open up and this is where you need to define the details about the custom attribute:

- **Common Name**: This is the name of the object. You can only use letters, numbers, and hyphens for the **common name** (**CN**).
- **LDAP Display Name**: When an object is referring to a script, program, or command-line utility, it needs to be called using the LDAP display name instead of the CN. When you define the CN, it will automatically create an LDAP Display Name.
- **Unique X500 Object ID**: Each and every attribute in an AD schema has a unique **object ID** (**OID**) value. There is a script developed by Microsoft to generate these unique OID values:

```
#---
$Prefix="1.2.840.113556.1.8000.2554"
$GUID=[System.Guid]::NewGuid().ToString()
$Parts=@()
```

```
$Parts+=[UInt64]::Parse($guid.SubString(0,4),
"AllowHexSpecifier")
$Parts+=[UInt64]::Parse($guid.SubString(4,4),
"AllowHexSpecifier")
$Parts+=[UInt64]::Parse($guid.SubString(9,4),
"AllowHexSpecifier")
$Parts+=[UInt64]::Parse($guid.SubString(14,4),
"AllowHexSpecifier")
$Parts+=[UInt64]::Parse($guid.SubString(19,4),
"AllowHexSpecifier")
$Parts+=[UInt64]::Parse($guid.SubString(24,6),
"AllowHexSpecifier")
$Parts+=[UInt64]::Parse($guid.SubString(30,6),
"AllowHexSpecifier")
$OID=[String]::Format("{0}.{1}.{2}.{3}.{4}.{5}.{6}.{7}",
$prefix,$Parts[0],$Parts[1],$Parts[2],$Parts[3],$Parts[4],
$Parts[5],$Parts[6])
$oid
#---
```

- **Syntax**: This defines the storage representation for the object. You are only allowed to use a syntax defined by Microsoft. One attribute can only associate with one syntax. In the following table, I have listed a few commonly used syntaxes:

Syntax	**Description**
Boolean	True or false
Unicode String	A large string
Numeric String	String of digits
Integer	32-bit numeric value
Large Integer	64-bit numeric value
SID	Security identifier value
Distinguished Name	String value to uniquely identify an object in AD

Along with the syntax, we can also define the minimum or maximum number of values. If they're not defined, the custom attribute will take the default values.

As an example, I would like to add a new attribute called `nINumber` and add it to the **user** class:

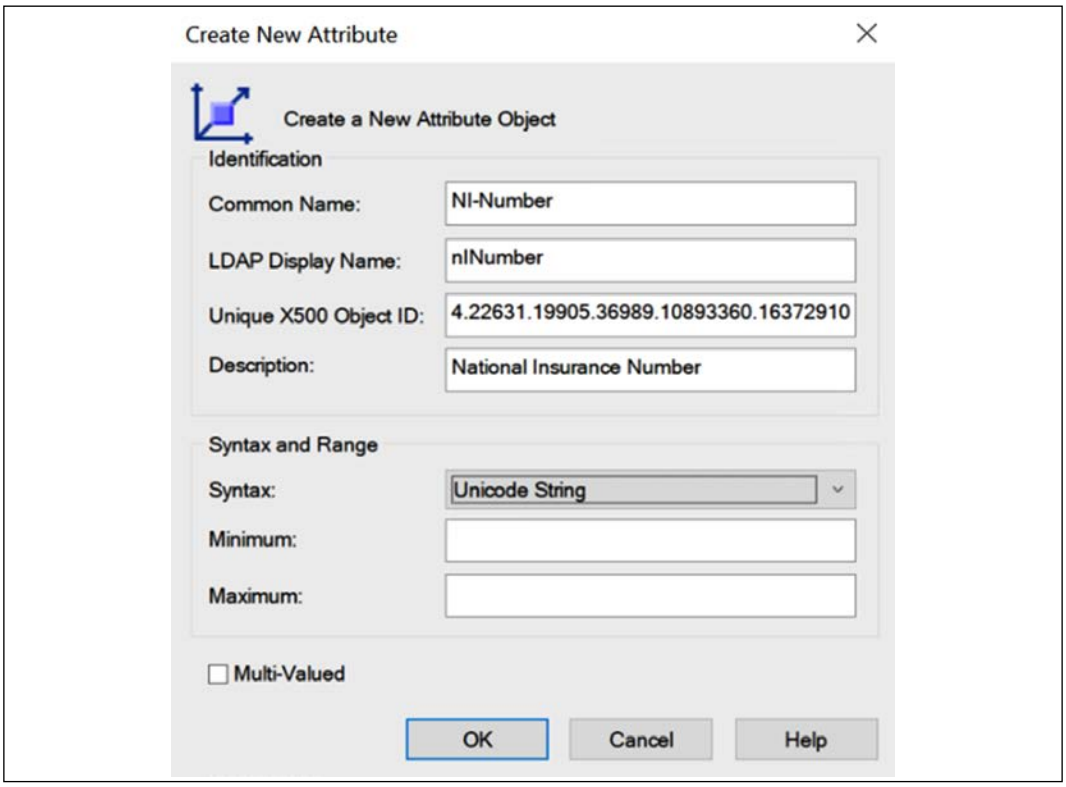


Figure 8.6: Custom attribute properties

As the next step, we need to add it to the **user** class. In order to do that, go to the `Classes` container, double-click on the **user** class, and click on the **Attributes** tab.

In there, by clicking the **Add...** button, we can browse and select the newly added attribute from the list:

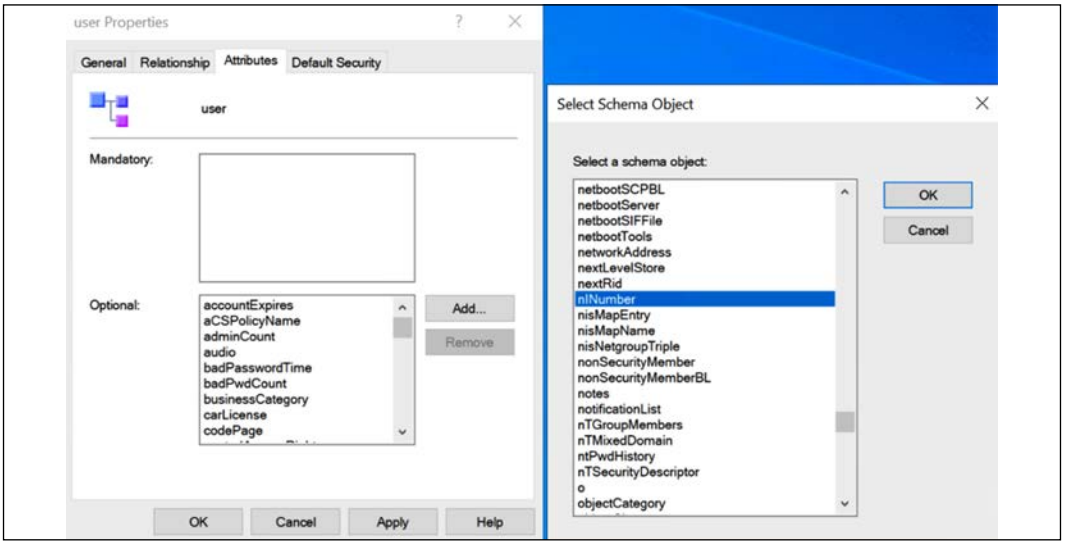


Figure 8.7: Custom attribute under Classes

Now when we open a user account, we can see the new attribute:

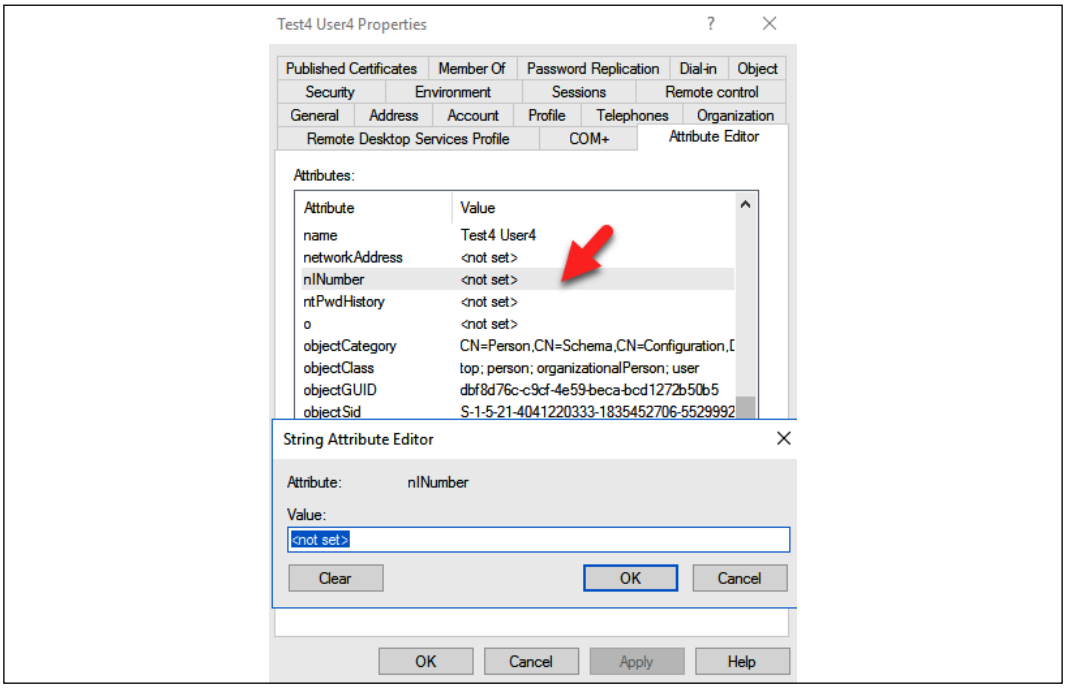


Figure 8.8: Custom attribute for user

Once the data has been added, we can filter out the information as required:

```
Get-ADuser "tuser4" -Properties nINumber | ft nINumber
```

Here, we learned how to add a custom attribute manually. Sometimes, AD-integrated applications also require certain attributes to be in place in order to store additional data. This is mostly done via an automated schema extension process. Microsoft Exchange Server is a great example of an application that requires an AD schema extension. During the Exchange Server installation process, the AD schema will be extended with new classes. More information about AD schema changes for Microsoft Exchange Server can be found at `https://bit.ly/3nJ0nRL`.

To add attributes to the schema, you need to have Schema Admin privileges or Enterprise Admin privileges. After adding new attributes, it's recommended that you reboot the system in order to reflect the new changes to the schema. However, on some occasions, even after a reboot, the new attribute may not be available under the object. In such situations, open **ADSI edit** and connect to each naming context | right-click | run **Update Schema Now**. After running it for all four naming contexts, reboot the domain controller.

Syncing custom attributes to Azure AD

In the previous section, I explained how we can create custom Active Directory attributes. When we sync users from on-prem Active Directory to Azure AD by using Azure AD Connect, some of the attributes will also sync but not all. The complete list of attributes that will sync with Azure AD Connect is available at `https://bit.ly/32lpFNn`.

But sometimes, there can be business requirements to sync these custom Active Directory attributes to Azure AD. It could be to use them with cloud applications (SaaS) or to use them with a legacy Line-of-Business (LOB) application that is migrated to Azure. We can sync these custom attributes to Azure AD by using the Azure AD Connect "*Directory extension attribute sync*" feature.

In my demo environment, I am going to demonstrate how to sync a newly created custom Active Directory attribute (user class) to Azure AD. To simplify the process, I have already installed Azure AD Connect and configured it for Azure AD sync. If you are not familiar with the Azure AD Connect configuration, please refer to *Chapter 18, Hybrid Identity*.

To configure custom attribute sync:

1. Launch the Azure AD Connect console in the Azure AD Connect server.
2. Then, from the list of options, select **Customize synchronization options** and click on **Next**.

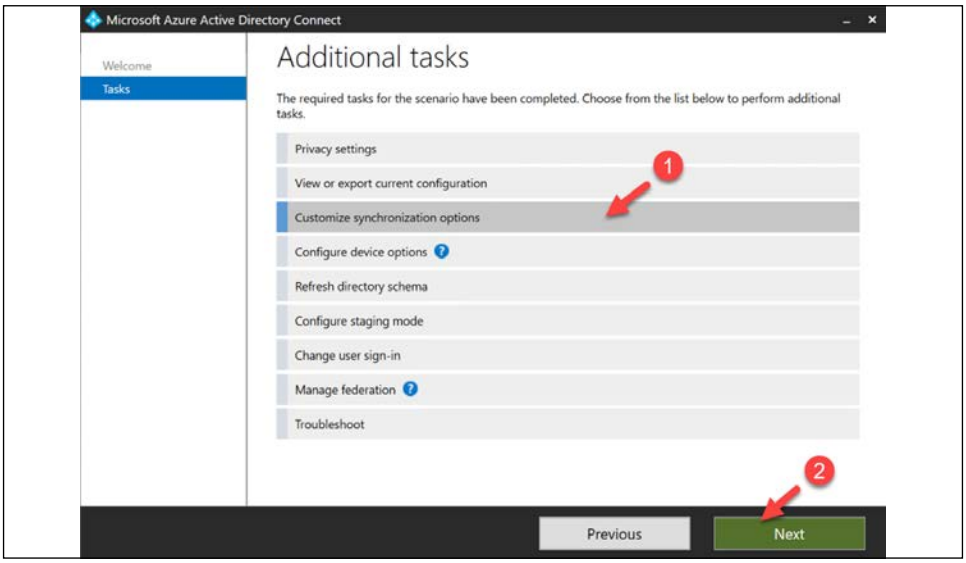


Figure 8.9: Azure AD Connect – modify synchronization options

3. Follow the authentication steps first and then, in the **Optional features** window, click on **Directory extension attribute sync** | **Next**.

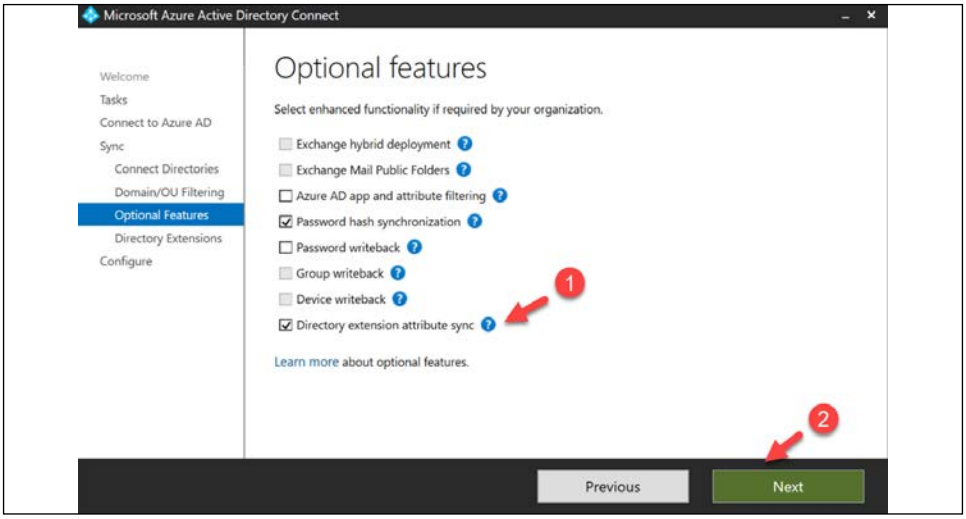


Figure 8.10: Azure AD Connect – Directory extension attribute sync

4. The next window lists down all the available attributes. From the list, I selected the newly created custom attribute **nINumber** and clicked **Next** to complete the configuration process.

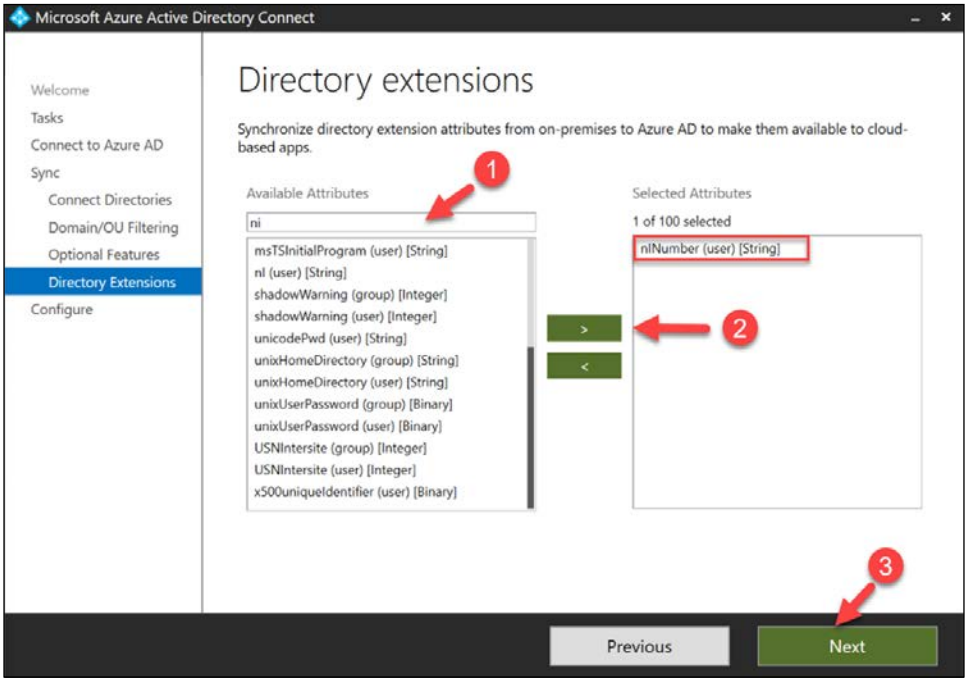


Figure 8.11: Azure AD Connect – select custom attribute

At the end of the configuration wizard, the new attribute values will sync to Azure AD.

We can verify these synced values by using **Microsoft Graph Explorer**. If you are new to this tool, don't worry – we will be covering it in *Chapter 16, Active Directory Security Best Practices*. Here, all I am using it for is to verify that the object sync is working as expected.

In my demo environment, I am using the following to verify the `nINumber` attribute value:

```
https://graph.microsoft.com/v1.0/users/testuser2@M365x581675.onmicrosoft.com?$select=displayName,givenName,extension_7c51781fcb5b481a98ba6efa5c7578d6_nINumber
```

In the above, `testuser2@M365x581675.onmicrosoft.com` is the user account. `displayName` and `givenName` are the attributes synced from the on-prem Active Directory via the default Azure AD Connect configuration.

As you can see, I have used `extension_7c51781fcb5b481a98ba6efa5c7578d6_nINumber` to define the `nINumber` attribute. In this string, `7c51781fcb5b481a98ba6efa5c7578d6` is the application ID value for the application called "Tenant Schema Extension App." All these custom attribute values are available under the Tenant Schema Extension App. If you are referring to a custom attribute in an Azure AD group or application, you must use the same format. The application ID value is different from one tenant to another. This application can be found under Azure AD Enterprise apps.

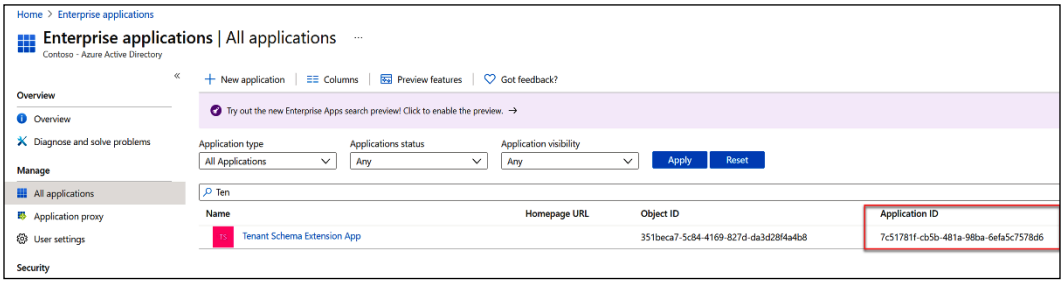


Figure 8.12: Application ID

When I use **Microsoft Graph Explorer**, as expected, I can see the custom attribute value.

Figure 8.13: Custom attribute value in Azure AD

Also, it matches the value in an on-prem user object.

```
PS C:\Users\dfrancis> Get-ADUser -Identity testuser2 -Properties "nINumber"

DistinguishedName : CN=Test User 2,OU=Test Users,DC=rebeladmin,DC=com
Enabled           : True
GivenName         : Test
Name              : Test User 2
nINumber          : 222222
ObjectClass       : user
ObjectGUID        : b640c70d-574c-4af2-967c-0c5248c61fb8
SamAccountName    : testuser2
SID               : S-1-5-21-3261202270-1104924415-3188262351-1602
Surname           : User 2
UserPrincipalName : testuser2@rebeladmin.com
```

Figure 8.14: Custom attribute value in Active Directory

As we can see, it is possible to sync custom attributes to Azure AD if required.

User accounts

In an Active Directory environment, what is the most common administrative task? Obviously, it's creating and managing user accounts. A user account does not only hold a username and password; it also holds data such as group memberships, the roaming profile path, the home folder path, login script information, remote dial-in permissions, and much more. Every time we set up a new account, we need to define values for these attributes. When the number of attributes increases, the number of mistakes that can happen during the account creation process also increases. An organization's identity is at stake here; even a small mistake can cost an organization a lot. As an example, if you add a user to the wrong user group accidentally, they will have access to some resources that they are not supposed to have access to.

When I create a **Statement of Work (SoW)** or implementation plan for a customer, I always start with a template. This template contains sections outlining what I need to change according to each customer's requirements, but it has lots more information that is common for all customers. When I use a template, it not only saves me time but also eliminates any risk of mistakes that can happen during document formatting. Similarly, in an Active Directory environment, user accounts may have common attributes and privilege levels. As an example, all users in a sales department will be members of the same security groups. They will also have the same logon scripts to map network shares. Therefore, instead of creating a sales user account from scratch, I can create a template with all the common attribute values and use it to create a new account. From Windows NT onward, Microsoft supports the creation of user account templates.

Even though there are attribute values that are common across user accounts, some attributes still need to have unique values or not a null value.

There are a couple of things to consider when creating user templates:

- **Do not copy user accounts as templates**: I've seen a lot of engineers just randomly select a user under the same **Organizational Unit** (**OU**) and use it as a template for a new user account. Templates should be a baseline. Other user accounts may have some unique privileges and attribute values, even if they are all under the same OU. Therefore, always keep user account templates separate.
- **Disable accounts**: No one should use template accounts to authenticate. When creating templates, make sure to set them as disabled accounts. During the new user creation process, the status of the account can be changed.

To demonstrate, I am going to create a user template for `Technical Department` users. The command I will use is as follows:

```
New-ADUser -Name "_TechSupport_Template" -GivenName "_TechSupport"
-Surname "_Template" -SamAccountName "techtemplate" -UserPrincipalName
"techtemplate@rebeladmin.com" -Path "OU=Users,OU=Europe
Office,DC=rebeladmin,DC=com" -AccountPassword(Read-Host -AsSecureString
"Type Password for User") -Enabled $false
```

The preceding command creates a user account called `_TechSupport_Template`. It also sets the new user account as a disabled account.

I'm also going to add the account to the `Technical Department` security group:

```
Add-ADGroupMember "Technical Department" "techtemplate"
```

Now, when we go to ADUC, we can see the new template. In order to create a new account from it, right-click and click **Copy...**:

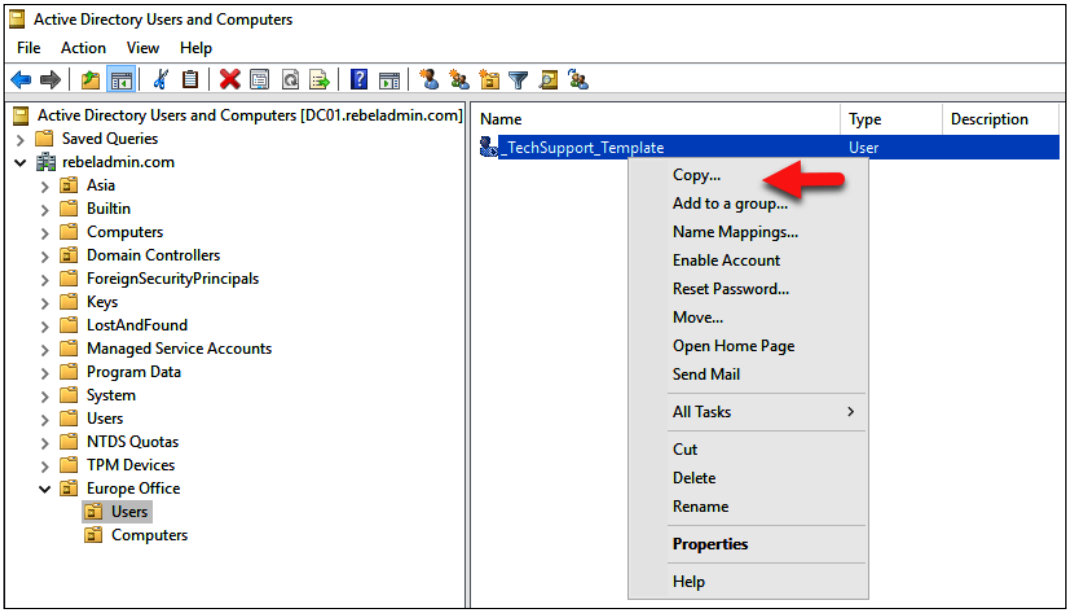


Figure 8.15: Copying a user account

Then, go and fill in the information in the wizard and complete the user account creation process.

In this section, we learned how to create user templates and how to use them in an Active Directory environment. Adopting user templates is a great way to maintain the consistency of user permissions in an Active Directory environment.

In the next section, we are going to learn about the different types of service accounts and how we can use them correctly in an Active Directory environment.

Managed Service Accounts (MSAs)

Service accounts are usually used when installing applications or services. A service account is a dedicated account with specific privileges that is used to run services, batch jobs, and management tasks. In most environments, service accounts are general user accounts. Since these service accounts are not used regularly, administrators have to keep track of these accounts and their credentials. The trouble is, if you reset the password of service accounts, you will need to update the services, databases, and application settings with a new password. Apart from that, engineers also have to manage the **service principal name** (**SPN**), which helps to identify service instances uniquely.

After considering all these challenges, Microsoft introduced MSAs with Windows Server 2008 R2. Microsoft's MSAs have the following:

- There is no more password management. MSAs use a complex and random, 240-character password that changes automatically when it reaches the domain or computer password expiry date.
- An MSA cannot be locked out or used for interactive login.
- A single MSA only can be used on one computer. It cannot be shared between multiple computers.
- An MSA provides simplified SPN management; the system will automatically change the SPN value if the `SamAccountName` details of the computer change or the DNS name property changes.

In order to create an MSA, we can use the following command. I am running this from the domain controller:

```
New-ADServiceAccount -Name "MyAcc1" -RestrictToSingleComputer
```

In the preceding command, I have created a service account called `MyAcc1` and I have restricted it to one computer.

We need to use PowerShell to create MSAs and GMSAs. It is not possible to create them using built-in GUI tools.

The next step is to associate the service account with the host `REBEL-SRV01` server:

```
Add-ADComputerServiceAccount -Identity REBEL-SRV01 -ServiceAccount
"MyAcc1"
```

The next step is to install the service account in the `REBEL-SRV01` server. We need the Active Directory PowerShell module for this. We can install it using **Remote Server Administration Tools** (**RSAT**). This can be done by running the `Install-WindowsFeature RSAT-AD-Tools` command. Once it's ready, run the following command:

```
Install-ADServiceAccount -Identity "MyAcc1"
```

We can test the service account using the following command:

```
Test-ADServiceAccount "MyAcc1"
```

It returns `True`, which means the test was successful.

From the Active Directory server, we can verify the service account by running the following command:

```
Get-ADServiceAccount "MyAcc1"
```

When configuring an MSA in service, be sure to leave the password empty. You do not need to define a password as the system autogenerates a password.

Group Managed Service Accounts (gMSAs)

In the previous section, we talked about MSAs. One MSA can be used with one computer only. But there are operational requirements that require the same service account to be shared in multiple hosts. Microsoft's **Network Load Balancing** (**NLB**) and **Internet Information Services** (**IIS**) server farms are good examples of this. All the hosts in these server groups are required to use the same service principal for authentication. gMSAs provide the same functionalities as MSAs. gMSAs were first introduced with Windows Server 2012.

The gMSA has the following capabilities:

- No password management
- Supports sharing across multiple hosts
- Can be used to run scheduled tasks (MSAs do not support the running of scheduled tasks)
- Uses Microsoft's **Key Distribution Center** (**KDC**) to create and manage passwords for the gMSA

Key Distribution Service (**KDS**) was first introduced with Windows Server 2012. KDS shares a secret (root key ID) among all the KDS instances in the domain.

This value changes periodically. When a gMSA requires a password, a Windows Server 2012 domain controller generates a password based on a common algorithm that includes a root key ID. Then, all the hosts that share the gMSA will query the domain controllers to retrieve the latest password.

These are the requirements for a gMSA:

- Windows Server 2012 or a higher AD forest level
- Windows Server 2012 or higher domain member servers (Windows 8 or higher domain-joined computers are also supported)
- 64-bit architecture to run PowerShell commands to manage the gMSA

gMSAs are not supported for failover clustering setups. However, gMSAs are supported for services that run on failover clusters.

In order to start the configuration process, we need to create a KDS root key. This needs to be run from the domain controller with Domain Admin or Enterprise Admin privileges:

```
Add-KdsRootKey –EffectiveImmediately
```

Once this is executed, there is a default 10-hour time limit to replicate the root key to all the domain controllers and start processing gMSA requests. In a testing environment with one domain controller, you can forcibly remove this waiting time and start responding to gMSA requests immediately. This is not recommended for a production environment. We can remove the 10-hour replication time limit by using the following command:

```
Add-KdsRootKey –EffectiveTime ((get-date).addhours(-10))
```

After that, we can create the first gMSA account. I have created an Active Directory group, `IISFARM`, and have added all my IIS servers to it. This farm will be using the new gMSA:

```
New-ADServiceAccount "Mygmsa1" -DNSHostName "web.rebeladmin.com" –
PrincipalsAllowedToRetrieveManagedPassword "IISFARM"
```

In the preceding command, **Mygmsa1** is the service account and `web.rebeladmin.com` is the **fully qualified domain name** (**FQDN**) of the service.

Once it's processed, we can verify the new account using the following command:

```
Get-ADServiceAccount "Mygmsa1"
```

The next step is to install `Mygmsa1` on the server in the IIS farm. `Mygmsa1` needs the AD PowerShell module to run. `Mygmsa1` can be installed using RSAT:

```
Install-ADServiceAccount -Identity "Mygmsa1"
```

If you created the server group recently and added the host, you need to restart the host computer to reflect the group membership. Otherwise, the aforementioned command will fail.

Once that's executed, we can test the service account by running the following command:

```
Test-ADServiceAccount " Mygmsa1"
```

Similar to the case with MSAs, when you configure a gMSA with any service, leave the password blank.

Uninstalling MSAs

You'll sometimes need to remove MSAs. This can be done by executing the following command:

```
Remove-ADServiceAccount –identity "Mygmsa1"
```

The preceding command will remove `Mygmsa1`. This applies to both types of MSAs.

Groups

In general, a **group** is a collection of users or resources that share the same characteristics and responsibilities. In an organization, individual identities get added and deleted, but roles and responsibilities do not change much. Therefore, the best way to manage privileges in organizations is based on roles and responsibilities rather than individuals. For example, in a sales department, salespersons will change quite often but their operational requirements will not change frequently. They all will access the same file shares, have the same permissions for the sales application, and have the same privileges to access each other's calendars. Active Directory groups allow you to isolate identities based on privilege requirements.

In an Active Directory environment, there are two categories of groups:

- **Security groups** are the type used to assign permissions to the resources. As an example, Rebeladmin Corp. has a team of 10 salespersons. They use a shared folder called `Sales` in the file server. Everyone in the sales team has the same access permissions to it. If the permissions were managed at the user level, the **Access Control List** (**ACL**) for the `Sales` folder would have 10 entries to represent the users.

  If a new salesperson joins the team, their account will need to be added to the ACL and match with the permissions manually by comparison with existing users in the ACL. Since this is a manual process, there is a possibility of the wrong privileges being applied by mistake. If it's based on security groups, we can create a group for the sales department and then add that to the `Sales` folder ACL with the relevant permissions. After that, we can remove individual entries for each sales user from the ACL. Thereafter, access to the `Sales` folder will be decided based on group membership.
- **Distribution groups** are to be used with an email system, such as Microsoft Exchange. This type of group is used to distribute incoming emails to group members. These groups are not security-enabled, so you cannot use them to assign permissions.

Group scope

Group scope helps to define the operation boundaries within the Active Directory forest. There are three predefined scopes to choose from when creating Active Directory groups:

- **Domain Local**: Domain Local groups can be used to manage privileges to resources in a single domain. This doesn't mean that the group can only have members within the same domain. It can have the following types of members:
  - User accounts from any trusted domain
  - Computer accounts from any trusted domain
  - Universal groups from any trusted forest
  - Domain Local groups from the same domain
  - Global groups from any trusted domain

  Domain Local group objects and their membership data will be replicated to every domain controller in the same domain.

- **Global**: Global groups can be used to manage privileges to resources in any domain under the same forest. Global groups can contain the following types of members:
    - User accounts from the same domain
    - Computer accounts from the same domain
    - Global groups from the same domain

  Global group objects and membership data will be replicated to every domain controller in the same domain. This group has limited membership and no high availability as the group will not be available for other domains in the forest. This is ideal when categorizing privileges based on roles and responsibilities.
- **Universal**: Similar to Global groups, Universal groups can be used to manage privileges in any domain in the forest. However, they allow you to have members from any domain. As an example, the `rebeladmin.com` and `rebeladmin.net` domains under the same forest can have one universal group called **Sales Managers**, with members from both domains. Then, I can use it to assign permissions to the folder in `rebeladmin.org` in the same forest. A Universal group can have the following types of members:
    - User accounts from any trusted domain
    - Computer accounts from any trusted domain
    - Global groups from any trusted domain
    - Universal groups from any domain in the same forest

  Universal group objects and membership data will be replicated to all the global catalog servers. Universal groups give the flexibility to apply permissions to any resource in any domain under the same forest.

Groups are supported to have other groups as members. These are called nested groups. This reduces the ACL changes further. In the preceding points, we have listed what types of groups are allowed to be added as nested groups under each scope.

Converting groups

Group scope needs to be defined during the group setup process. But with operational requirements or infrastructure changes, there could be occasions where the existing scope is not valid anymore. In such a situation, instead of setting up a new group, we can change the group scope.

However, it doesn't mean you can change the existing group scope to any scope you like. There are some rules to follow. The following table explains supported paths:

Group scope	Domain Local	Global	Universal
Domain Local	N/A	X	Yes (only if there are no other Domain Local groups as members)
Global	X	N/A	Yes (only if it's not a member of other Global groups)
Universal	Yes	Yes (only if there are no other Universal groups as members)	N/A

Setting up groups

Similar to user accounts, there are several methods we can use to create and manage groups:

- Active Directory Administrative Center (ADAC)
- ADUC MMC
- PowerShell cmdlets

In this section, I am going to use PowerShell cmdlets to set up and manage Active Directory groups.

The `New-ADGroup` cmdlet can be used to add a new group to an AD environment. We can review the full syntax for the command using this:

```
Get-Command New-ADGroup -Syntax
```

As an example, I am going to create a new security group called `Sales Team`:

```
New-ADGroup -Name "Sales Team" -GroupCategory Security -GroupScope
Global -Path "OU=Users,OU=Europe,DC=rebeladmin,DC=com"
```

In the preceding command, the following is true:

- `-GroupCategory`: This defines the type of the group (security or distribution).
- `-GroupScope`: This defines the scope of the group.
- `-Path`: This defines the path for the group object. If the `-Path` option is not used, the default container will be used, `Users`.

Due to the importance of the group, I want to protect this group object from accidental deletion:

```
Get-ADGroup "Sales Team" | Set-ADObject
-ProtectedFromAccidentalDeletion:$true
```

This can also be set using the group's properties window:

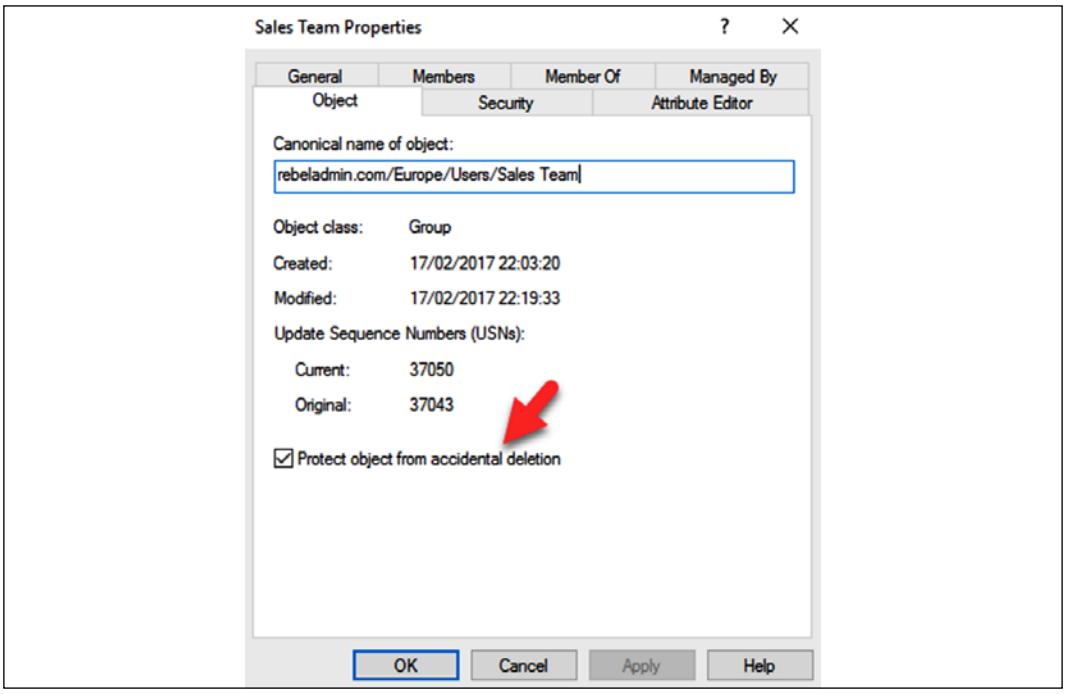


Figure 8.16: Protect object from accidental deletion

Now the group is ready for new members. To add new members, we can use the following:

```
Add-ADGroupMember "Sales Team" tuser3,tuser4,tuser5
```

The previous command will add the `tuser3`, `tuser4`, and `tuser5` users to the group.

If we need to remove a user from the group, we can use the following command:

```
Remove-ADGroupMember "Sales Team" tuser4
```

We can review the group properties using the `Get-ADGroup` cmdlet:

```
Get-ADGroup "Sales Team"
```

By using the following command, we can retrieve specific values from the group:

```
Get-ADGroup "Sales Team" -Properties DistinguishedName,Members | fl
DistinguishedName,Members
```

The preceding command will list the `DistinguishedName` and `Members` values of the `Sales Team` security group:

```
PS C:\Users\dfrancis> Get-ADGroup "Sales Team" -Properties DistinguishedName,Members | fl DistinguishedName,Members

DistinguishedName : CN=Sales Team,OU=Asia,DC=rebeladmin,DC=com
Members           : {CN=Test User 4,OU=Asia,DC=rebeladmin,DC=com, CN=Test User 3,OU=Asia,DC=rebeladmin,DC=com, CN=Test
                    User 2,OU=Asia,DC=rebeladmin,DC=com, CN=Test User,OU=Asia,DC=rebeladmin,DC=com}


PS C:\Users\dfrancis>
```

Figure 8.17: DN and Members values

If we need to change the scope of the group, that can be done using the following command:

```
Set-ADGroup "Sales Team" -GroupScope Universal
```

This will change the group scope from `Global` to `Universal`:

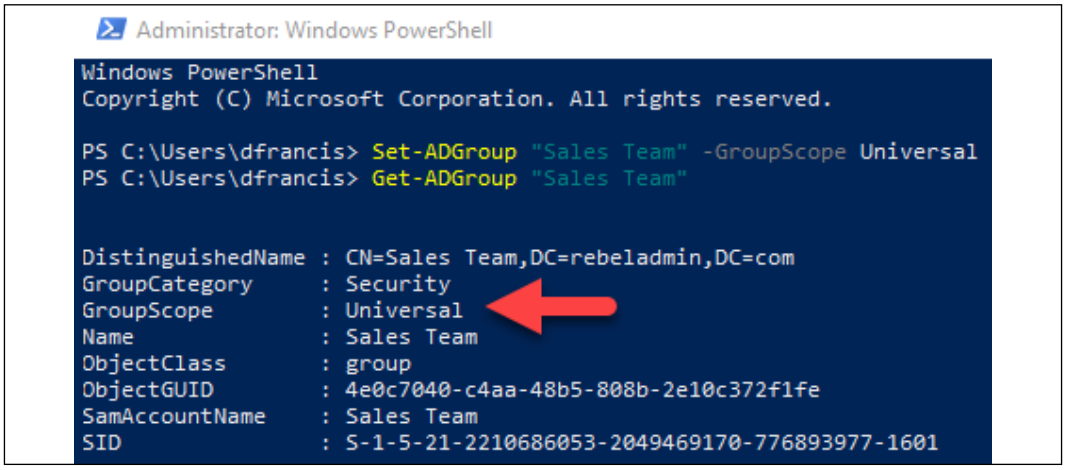


Figure 8.18: Group scope

Last but not least, a group can be removed using the `Remove-ADGroup` cmdlet:

```
Remove-ADGroup "Sales Team"
```

The preceding command will remove the `Sales Team` group.

If you have the accidental deletion option enabled in the group, you need to remove it before executing the Remove-ADGroup command. Otherwise, it will fail to execute. To remove the accidental deletion option, we can use the Get-ADGroup "Sales Team" | Set-ADObject -ProtectedFromAccidentalDeletion:$false command.

Devices and other objects

Apart from computers, Active Directory supports a few other devices and object types as well. In this section, we will look into these different object types:

- **Printers**: Printers are one of the most commonly shared resources in office networks. We can use several methods to configure shared printers on user computers. We can set them up using the printer setup wizard in Windows and connect to a printer via an IP address. We can also use logon scripts to map and install printers on workstations. If an organization uses printer servers, we can connect to them and install the printers, too. In an Active Directory environment, we can register a printer as an object in Active Directory. This will allow users to browse Active Directory, find the relevant printer, and then install it on the workstation.

  To register a printer with AD, go to **Printer properties** and then to the **Sharing** tab. There, you can check the **List in the directory** checkbox to list the printer on Active Directory.

  On the workstation, during the printer setup, we need to select the option to list printers from the directory in order to see the Active Directory-integrated printers:

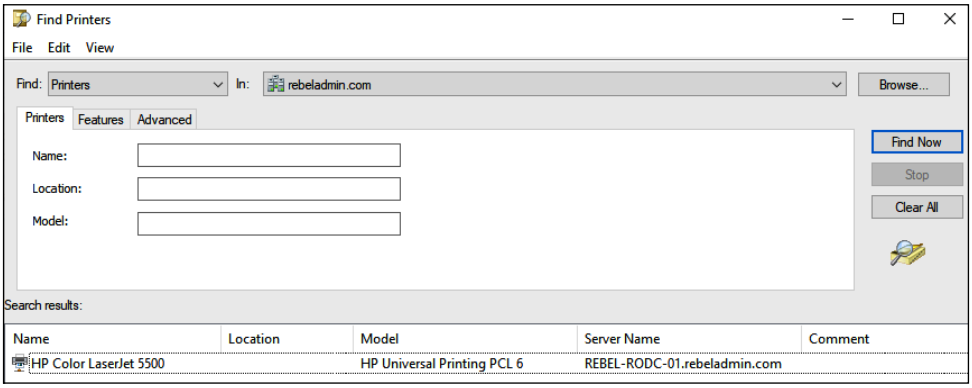


Figure 8.19: Printer object

- **iNetOrgPerson**: This object is defined in RFC 2798. It is used by other directory services that are based on LDAP and X.500. This object type exists in AD in order to support migration from non-Microsoft directory services and to support applications that require `iNetOrgPerson` objects. This object can be converted into a regular user if required.

  In order to add this object, we can use ADAC, ADUC, or PowerShell. In PowerShell, you would use the same `New-ADUser` cmdlet:

```
New-ADUser -Name "Inet User1" -GivenName "Inet"
 -Surname "User1" -SamAccountName "inetuser1"
 -UserPrincipalName "isuer1@rebeladmin.com"
 -AccountPassword (Read-Host -AsSecureString
 "Type Password for User")
 -Enabled $true -Path "OU=Users,OU=Europe,DC=rebeladmin,DC=com"
 –Type iNetOrgPerson
```

  The preceding command will add an `iNetOrgPerson` object called `Inet User1`. The only difference in the command from that of a regular Active Directory user is `–Type iNetOrgPerson`, which defines the account type as `iNetOrgPerson`.

  We can convert the `inetOrgPerson` object to a regular Active Directory user object using the following command:

```
Set-ADUser "inetuser1" -Remove @{objectClass='inetOrgPerson'}
```

Best practices

Here, we will look into some of the best practices that can be used to manage Active Directory objects:

- **Housekeeping**: It is important to review the validity of Active Directory objects from time to time. There can be objects that are no longer active in operations. There are several ways to handle these objects:
  - If it's possible to confirm that objects are not in use 100% of the time, objects can be completely deleted from Active Directory.
  - If it's not possible to confirm, the object can be disabled and monitored for events. If there are no events, the object can be removed from Active Directory.

  To manage disabled objects, it is advisable to create a different OU and move the disabled objects to that. This will allow you to keep track of them and allow easy access when required.

In Active Directory, there can be objects that are only used for a limited time. As an example, there can be contractors who only work on certain projects. The user accounts for these contractors are only used during the project. These types of objects can be kept disabled and only enabled when required. Past employee accounts also fall into the same category.

- **Adding description**: In Active Directory objects, there is an attribute where you can add a description of the object. It is recommended that you add a description to an object if it cannot be described with its given name. This mainly applies to service accounts and group objects. Object description allows engineers to locate an object quickly and understand its purpose.
- **Protecting objects from accidental deletion**: This feature was introduced with AD DS 2008. It can be enabled on user, computer, and group objects. This prevents objects from accidental deletion. It can be enabled at an individual object level or the OU/directory level.
- **Object naming convention**: When defining values for objects, always follow a standard. As an example, some organizations prefer to use the first few letters of the first name and last name as a username. Some may prefer the `firstname.lastname` format. These standards can differ from business to business. It is recommended to document these standards so other team members can also follow the same processes. If there is no such document, go and review similar types of objects to understand the standards being used.

Summary

In this chapter, we learned about Active Directory objects and attributes, and how they are defined in the Active Directory schema. We also learned how to add custom attributes to the Active Directory schema. After, we learned how we can sync custom attributes to Azure AD. We also looked into creating user account templates and the different types of service accounts. In an Active Directory environment, sometimes we need to manage permissions for groups of users who have similar operation requirements (with their department, job role, and so on). This is done using Active Directory groups. There are different group categories to choose from. In this chapter, we also looked into these group types and learned how to use them appropriately. We also went through object management best practices to help improve our Active Directory object management experience.

In the next chapter, we will be learning about designing and managing the OU.

9

Designing the OU Structure

The local library in my town has a collection of nearly 10,000 books. These books cover many different subjects. When I walk into the library, I can see that there are signs hanging from the ceiling that help to identify the different library aisles that belong to different book categories such as novels, history, arts, technology, and cooking. So, if I know the type of book I am looking for, I can easily go to the relevant aisle. Each of these aisles has multiple bookshelves. These bookshelves are further categorized into subcategories. At the top of each bookshelf, there is a sign describing which subcategory it belongs to. As an example, the History section has bookshelves with categories such as History of Europe, History of Asia, World History, and so on. This makes book selection even easier—telling me exactly which bookshelves to look for. When I go to a bookshelf, the books are usually organized in alphabetical order. Each book has a small label indicating the first character of the book title. If I am looking for a book on British history, I can look at the books with a *B* label. So, out of 10,000 books, within a few minutes, I can locate a book I need. If it wasn't structured, I would have to spend hours finding books I wanted. This doesn't only benefit the members; when the library receives new books, employees know exactly where to rack those up, as there is a defined system in place for everyone to follow.

But Kingston children's library is different—children do not follow the same rules, as they are too young to understand them. Therefore, books end on up the wrong shelves.

In the main library, it is easier to locate a book than in the children's library. Just having a structured system will not result in the same output. It depends on the way the system is designed and, more importantly, the way it is maintained.

In **Active Directory** (**AD**), there can be hundreds or thousands of objects based on an organization's size. Each of these objects has different operational and security requirements. We do not manage a user object in the same way as we manage computer objects. In the preceding library example, the structured environment helps library users to locate a book easily from lots of similar types of objects. It also helps the administrators to maintain the library service with integrity. In an Active Directory environment, **Organizational Units** (**OUs**) allow engineers to categorize objects into smaller administrative boundaries based on the object class and administrative requirements.

OUs can also be used to delegate control and manage the Group Policy processing order. In this chapter, we are going to discuss these in detail. We are also going to learn about different OU design models along with their advantages and disadvantages. Last but not least, we are also going to look into OU management using PowerShell.

In this chapter, we are going to look into the following topics:

- OUs in operations
- OU design models
- Managing the OU structure

Before we go into OU designs, it is important to understand the role of OUs and their benefits.

OUs in operations

In *Chapter 3*, *Designing an Active Directory Infrastructure*, we learned how we can represent an organization's logical structure based on domains. But this hierarchical design has border boundaries. Therefore, we do not consider object class requirements. OUs help us to define the hierarchical structure for objects within the domain boundaries.

There are three main reasons for creating an OU:

- Organizing objects
- Delegating control
- Applying group policies

Let's go ahead and look into each of these points in detail.

Organizing objects

An Active Directory domain controller supports holding nearly two billion objects. As the number of objects increases in the Active Directory environment, the effort we need to put in to manage them also increases. If we have a proper structure to group these objects into smaller groups, then we have more control over them and we know at a glance where we can find a specific object:

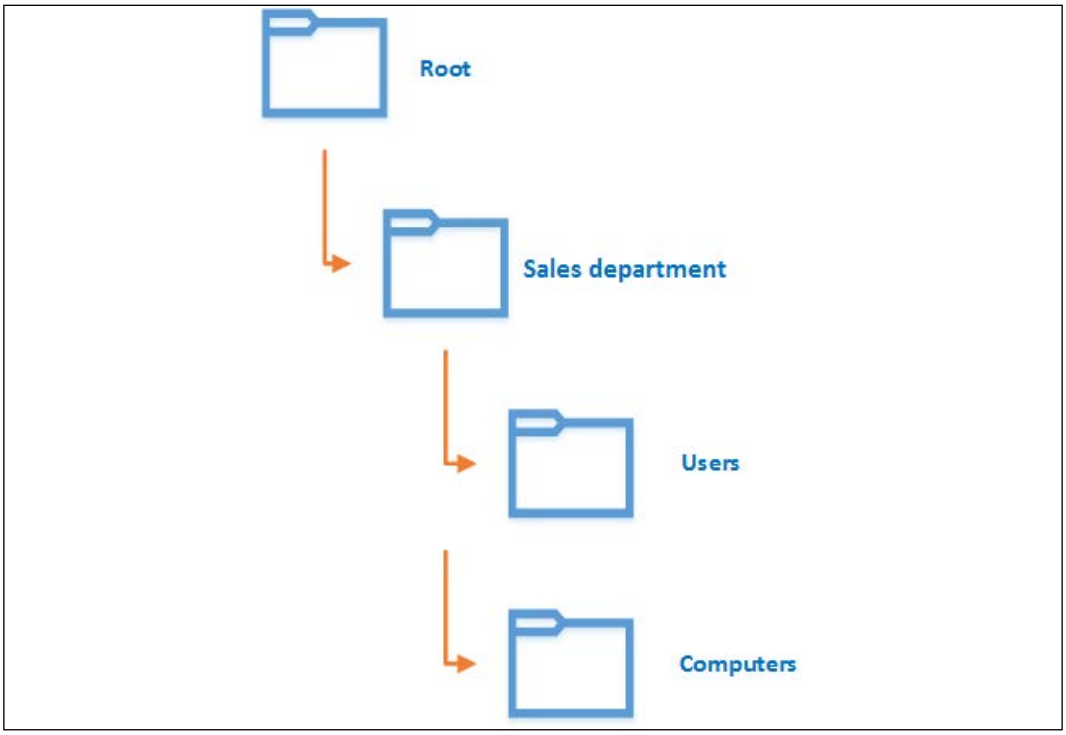


Figure 9.1: Hierarchical structure sample

In the preceding diagram, Rebeladmin Corp. has nearly 100 salespersons. They are also using nearly 150 desktops and laptops. There is no problem putting all of those objects into the **root** of the hierarchy (under default containers), but it will not be easy to identify **Sales department** objects from other Active Directory objects related to other departments. Instead, we can create an OU for **Sales department**. It can be further categorized into two OUs: **Users** and **Computers**. Both of these OUs will be in the same hierarchical level. Now, if we need to locate a user object that belongs to the **Sales department**, we definitely know it should be under the **Users** OU of the **Sales department** OU.

In the same way, if we need to add a new Active Directory object under **Sales department**, we now have a predefined structure to place it in. Every engineer in the IT department should follow the same structure when managing objects, and this will not change based on the individual's preferences.

Delegating control

OUs can be used to delegate the administration of a set of objects to individuals or groups. These individuals will have control over managing objects in that particular OU, and these privileges are usually defined by the Domain Admins or Enterprise Admins. Later on in this chapter, we will look into the configuration steps of Active Directory delegation.

Group policies

We cannot think of Active Directory without group policies. Using group policies, we can standardize application settings, security settings, and system settings of Active Directory objects. Each and every object in Active Directory has different operation and security requirements. As an example, sales department computer security requirements are different from a server that hosts the database system. Group policies can be bound to OUs. Therefore, objects that have different Group Policy requirements can be placed into different OUs and have corresponding policies applied to them. Even though this is the most common reason for OUs, this is where things mostly go wrong as well. If you do not consider Group Policy inheritance and Group Policy precedence, it will be difficult to target the relevant objects:

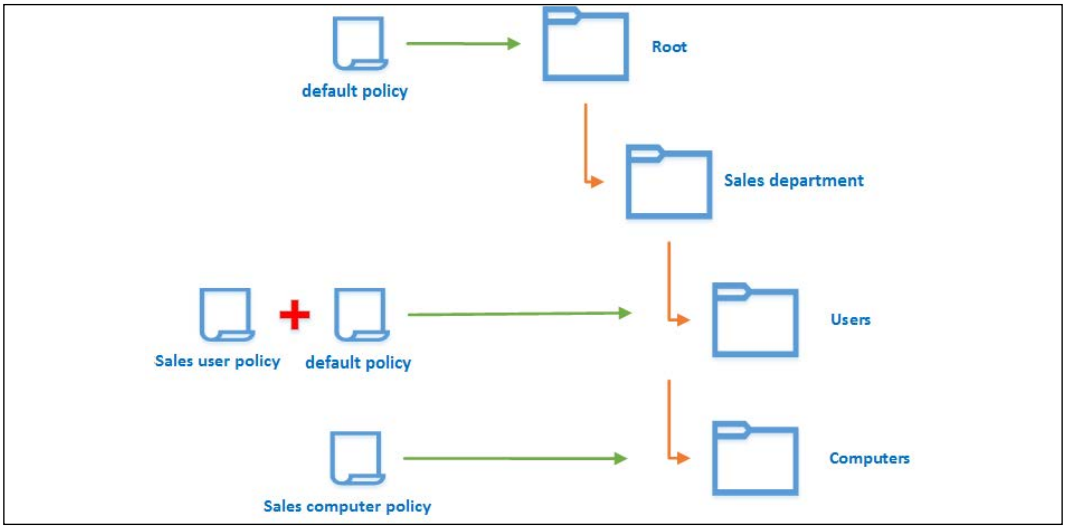


Figure 9.2: OUs with group policies

In the preceding example, the structure has a **default policy**, which applies to the majority of the objects in the Active Directory environment. Therefore, it is created at the top of the hierarchy. It is inherited by the child OU levels by default. Therefore, the **Sales department** and **Users** under the **Sales department** will have the **default policy** inherited by default; but the organization also would like to add specific sales user settings via a Group Policy.

For that, a new Group Policy is created with the name **Sales user policy** and is mapped to the **Users** OU of **Sales department**. By default, the **Computers** OU under **Sales department** will also have inherited the **default policy**, but due to operational requirements, it should block that policy and should only have **Sales computer policy**, which is linked to the **Computers** OU. In order to do that, we have to block the inheritance. Once it is blocked, it will no longer apply any inherited policy from the parent OUs and will only apply the policies that are linked to the OU directly. This explains how we can use OUs to apply group policies to relevant objects.

Before we implement the OU structure, we need to decide why we need each of these OUs. This needs to be evaluated based on the preceding three points, which are organizing objects, delegating control, and applying group policies; if the requirement does not fall under these three points, we should not set up an OU. Most engineers do not pay attention to designing the OU structure because these structures are easy to change compared to domain structures. If the OU structure does not match your needs, it can be replaced with a completely new structure. But it is followed by moving objects and the changes to Group Policy mapping. Even though OUs have the flexibility to adopt structural changes, it's important to get the initial requirements correct.

Containers vs. OUs

When you open the **Active Directory Users and Computers** (**ADUC**) **Microsoft Management Console** (**MMC**) with the advanced view, there will be pre-setup folders.

But not all of these are OUs. They are mostly **containers**:

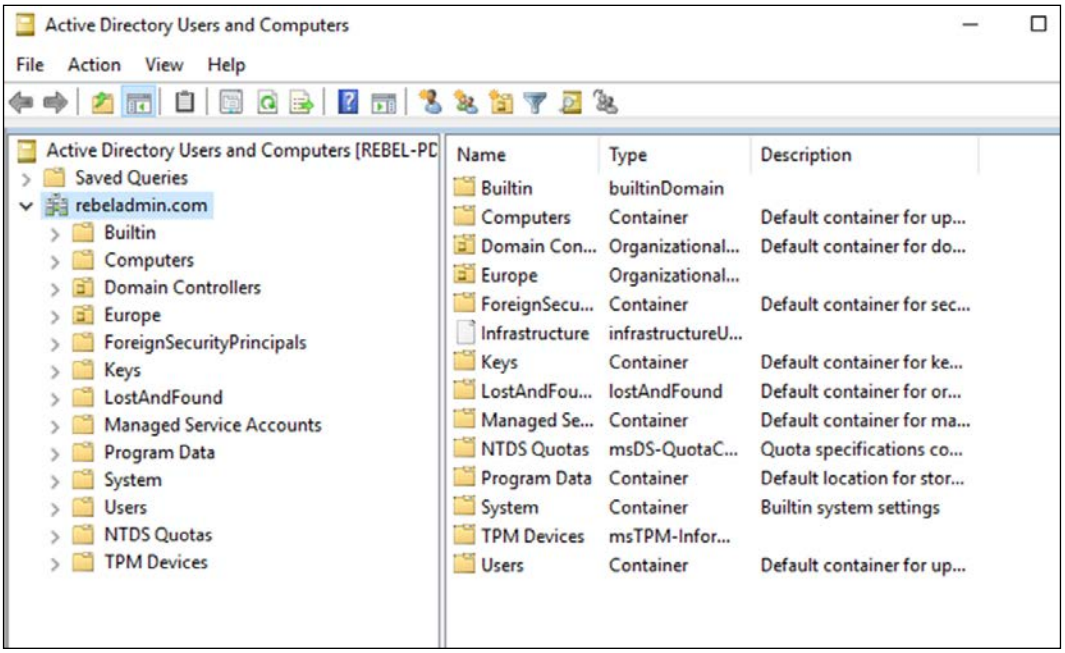


Figure 9.3: Containers and OUs

The only default OU in the AD environment is the `Domain Controllers` OU. All other folders in the tree are containers. Containers can also contain objects. The `Computers` and `Users` containers are good examples of that. By default, any computer object will be stored in the `Computers` container. All of the default user accounts and security groups are stored in the `Users` container. Similar to OUs, containers can also be used to delegate administrative control. The only difference between containers and OUs is that group policies cannot apply to containers. Group policies can be assigned only to OUs. The system also does not allow you to create new containers other than the containers that are created by the system.

Active Directory Groups vs. OUs

Active Directory Groups and OUs have certain similarities. Both can be used to group objects together. Both can be used with Group policies. But there are a number of differences between the two.

Feature	Active Directory Groups	OUs
Hierarchical Structure	A flat structure. A group can have different object types (users, devices, groups) as members but can't present them in hierarchical order.	Can use different models to arrange OUs in hierarchical order. Also, can change the structure easily whenever required.
Object placement	One object can be part of many different groups.	One object can only use one OU at a given time.
Access Control Lists (ACLs)	Groups can be added to ACLs.	OUs can't be part of ACLs.
Security Identifier (SID) value	Does have a SID value.	Doesn't have a SID value.

It is important to understand we can't replace Active Directory Groups with OUs or vice versa. We use each for different tasks in Active Directory environments. Active Directory OUs are best suited for grouping objects in hierarchical order and delegating control. Active Directory groups are best suited for use with resource permissions.

OU design models

In this section, we are going to look into different OU design models. This doesn't mean you need to stick to one of these. Modern infrastructure requirements are complex and challenging. These models will guide you to create a design that suits your organization's requirements.

The container model

In the *Containers vs. OUs* section, I mentioned default containers in an Active Directory environment. One of the characteristics of these default containers is that they have large administrative boundaries. As an example, the *Computers* container will contain any computers added to AD by default. It can be a physical server, virtual server, desktop computer, or laptop. The container model is based on a similar concept. This is mainly suited for small businesses where you have limited administrative and security requirements with Active Directory objects.

When OU boundaries are large, it is not possible to apply tailored group policies or precise delegated controls as each OU can contain different classes of objects:

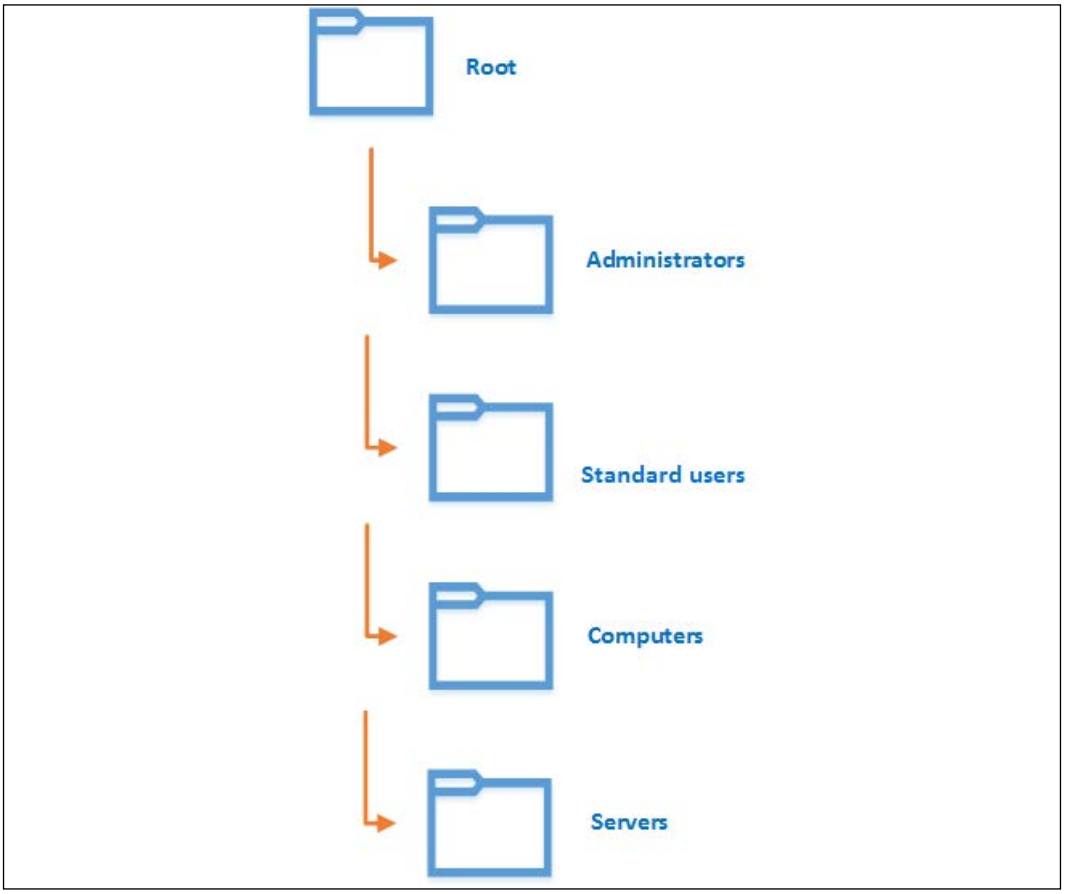


Figure 9.4: The container model

In the preceding diagram, the organization has four main OUs. In the container model, there will not be any child OUs. As we can see, each OU covers a large boundary. As an example, the **Standard users** OU will contain objects from each and every department. If we need to apply different policies to departments, we need to use other GPO targeting methods such as item-level targeting, WMI filters, etc.

Here, the user object's OU will be decided based on its privileges. If a user object has administrator privileges, it will be in the **Administrators** OU, and if not, it will be in the **Standard users** OU.

The following table discusses the advantages and disadvantages of the container model:

Advantages	Disadvantages
Easy to implement: There are no child OUs and there are no granular-level security and administrative boundary designs required.	**Less control**: It will not categorize objects in detail. Therefore, administrators will have less control over the objects. Since administrative boundaries are large, it is not practical to implement delegated controls either.
Fewer group policies: When OUs contain a large number of objects from different classes, it's hard to be specific about the system or security settings. Therefore, each OU will have a smaller number of group policies. Even on those, the settings will be more high-level than complex tune-ups.	**Less security**: It is difficult to apply different group policies to match the security requirements of objects and workloads as the OU structure doesn't help in grouping the relevant objects together.
Easy to change: Since each OU doesn't contain many group policies and complex inheritance, if needed, the structure can be changed completely with minimum impact.	**Less extensibility**: This is not a future-proof design. As the business grows, object management requirements will change as well. If further categorization is required, it will be difficult to implement without a complete structural change.

The object type model

Active Directory contains different types of objects, such as users, groups, and computers. It is possible to group these different types of objects into separate OUs and manage them.

Each of these types can be further categorized into child OUs based on geographical location or roles and responsibilities:

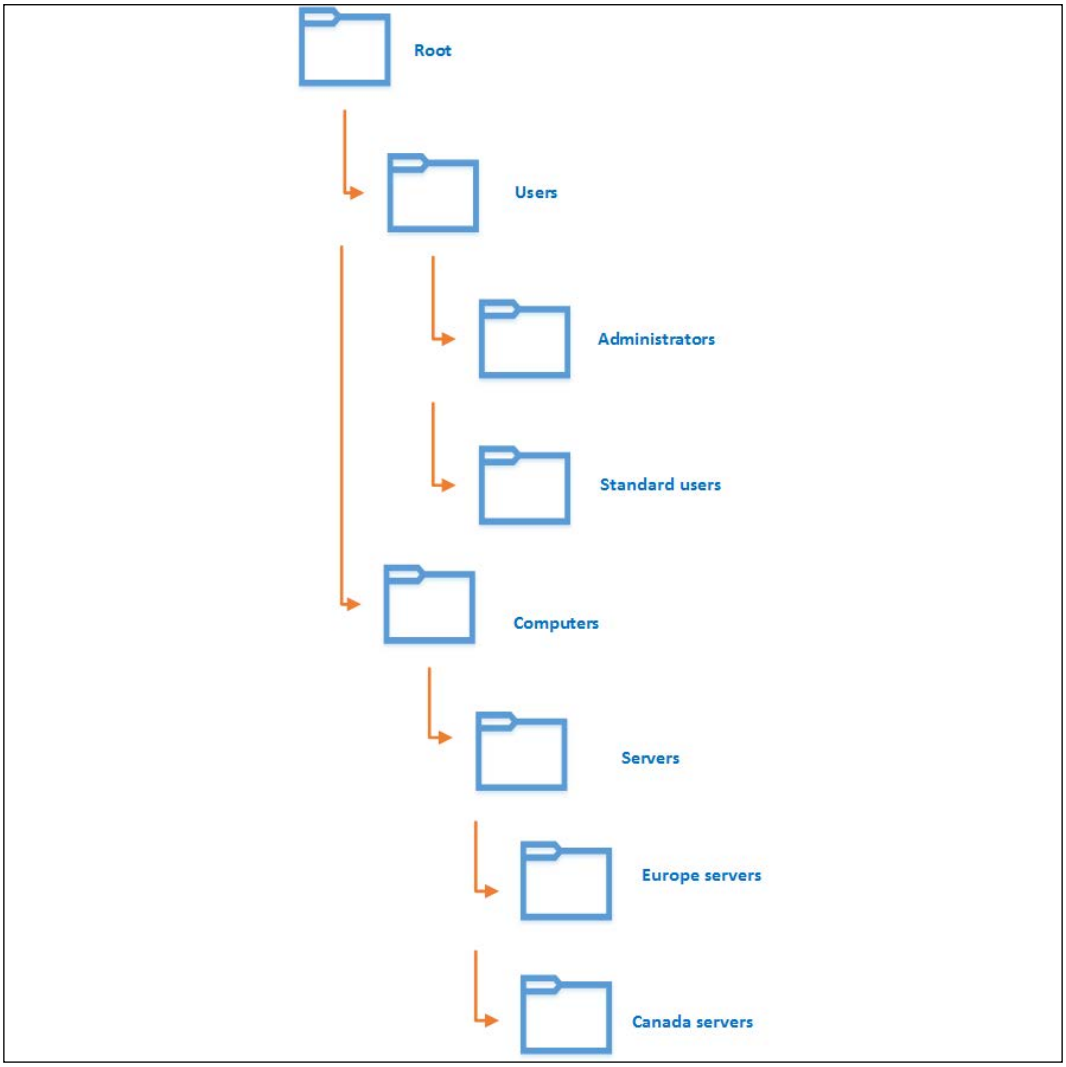


Figure 9.5: The object type model

In the preceding diagram, OUs are mainly categorized based on the **Users** and **Computers** object types. The **Users** OU is further categorized into **Administrators** and **Standard users**. These were based on the privilege level and each object's responsibilities within the organization. The **Computers** OU got the child OU for **Servers**. It plays a different role than other computer objects, such as desktops/ laptops. It was further categorized based on geographical locations.

The following table lists the advantages and disadvantages of the object type model:

Advantages	Disadvantages
Flexibility: It gives greater flexibility when it comes to categorizing objects. Under each object type, you can categorize objects further based on roles, responsibilities, geographical locations, teams, departments, and more.	**Complexity**: As this model gives freedom to engineers to categorize objects using many options, the structure can get complex to maintain. There is no limit to the number of levels OUs can break into, but when the number of levels increases, management gets complex too.
Easy management of AD objects: The core value behind this model is easy manageability. That's why it can use many methods to categorize objects. When objects are categorized into small administrative boundaries, it's easy to manage the objects in every aspect.	**Structural changes are difficult**: If there is a requirement to change the structure of OUs, it will be difficult as more tailored settings and delegated controls are applied to the objects. In structural changes, these specific settings will need to move with objects as well.
Extensibility: Since the model allows you to use a large number of options for categorized objects, it has greater extensibility to implement future organizational requirements with minimum impact.	N/A
Use of group policies: More tailored group policies can be applied to objects as categorization is more granular.	N/A

The functions model

In an Active Directory environment, there are mainly two types of objects – Accounts and Resources. In a small organization, we can group these Accounts and Resources based on "functions." As an example, let's assume Rebeladmin Inc. is running three IIS web servers and two Microsoft SQL servers to host their web-based applications. Web servers have a specific function in an organization. We can group these three IIS web servers as the "Web Servers" group. Each server in the group has a similar role to play in company operations. In the same way, we can group Microsoft SQL servers as "Database Servers." These servers will only be used to host databases. The same method can also apply to "Accounts." As an example, users in "Marketing Team" have the same set of responsibilities within the organization.

Therefore, it makes sense to manage Marketing team accounts as one group. Let's see how we can use this to create an OU structure.

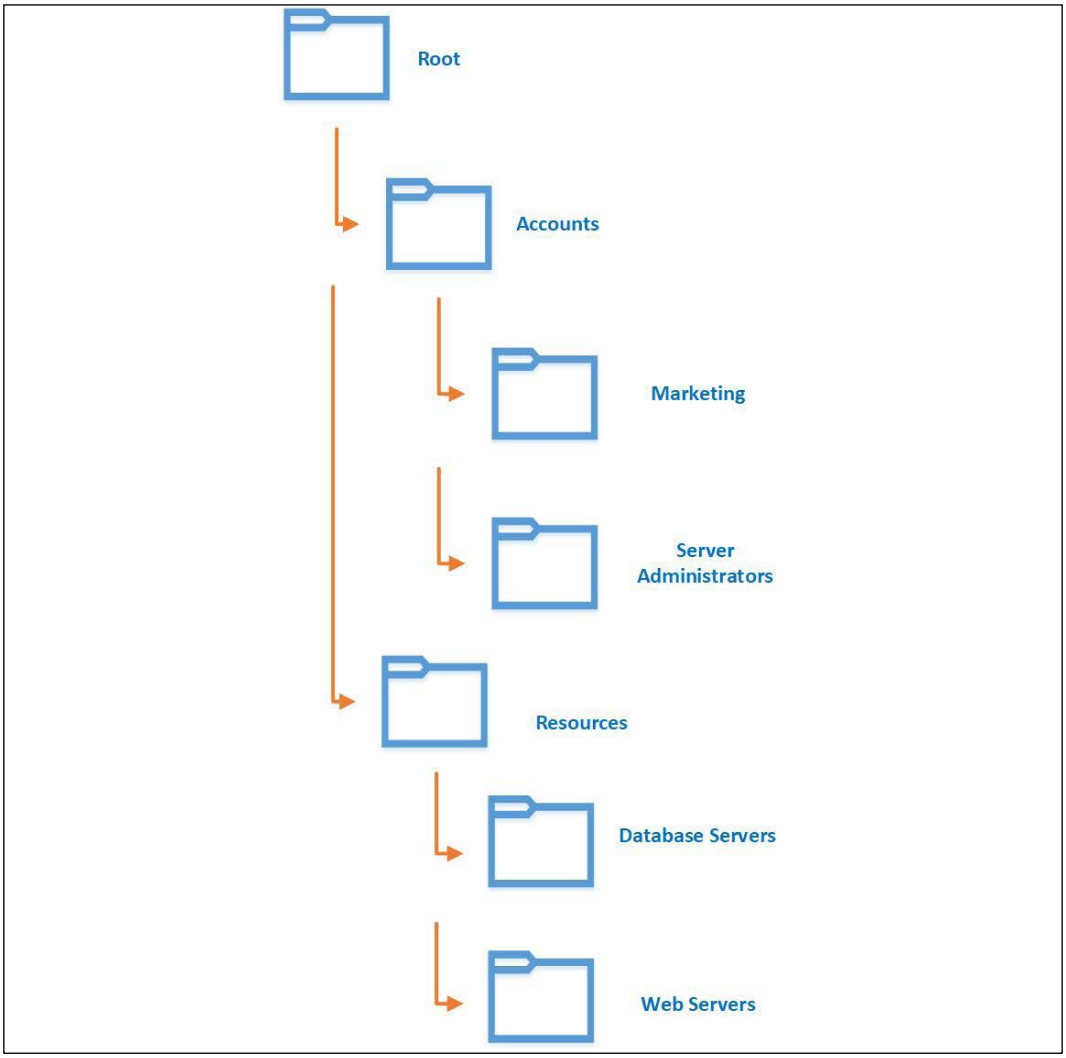


Figure 9.6: The functions model

In the above design sample, there are two main OUs – Accounts and Resources. Then, the Accounts OU is further broken down into different OUs based on the Account's function/responsibilities.

This doesn't need to necessarily follow the company organization chart. It is possible for each department to have multiple Accounts groups based on different functions. As an example, the IT department could have different OUs such as "Server Administrators," "Network Engineers," "Storage Administrators," and so on. However, there can be situations where one account has responsibilities that could fit into different OUs. Unlike groups, an object can only appear in one OU. So, it is important to understand the role and responsibilities of an object before its placement.

The following table lists the advantages and disadvantages of the functions model:

Advantages	**Disadvantages**
Easy delegation – when we use delegation, we mainly consider the responsibilities of an account. In this mode, we are already grouping objects based on responsibilities and it allows administrators to delegate permissions easily.	Not suitable for large organizations as it would add overheads for the IT department to figure out the functions/responsibilities of each and every object.
Based on the role and responsibilities of an account, we can decide if it's a privileged account or not. This model helps to identify privileged accounts easily and apply relevant security measures to protect them.	This possibly could cause conflict with the business operation model. The business may have a different organizational structure created based on roles and responsibilities by only considering business operations. As an example, the business sees the IT department as one entity. But in this OU model, the IT department could have a collection of OUs to cover different responsibilities.
	Additional administrative overhead for IT engineers to design and maintain. Before creating an OU or moving an object to an OU, engineers may need to talk to department heads, team leads, or even individual users to understand functions/responsibilities.

The geographical model

This is one of the most commonly used models for large organizations. The OU structure will be based on the geographical location of the branch offices.

Each of these branch offices may also have its own IT team. So, the main idea behind this model is to delegate administrative control:

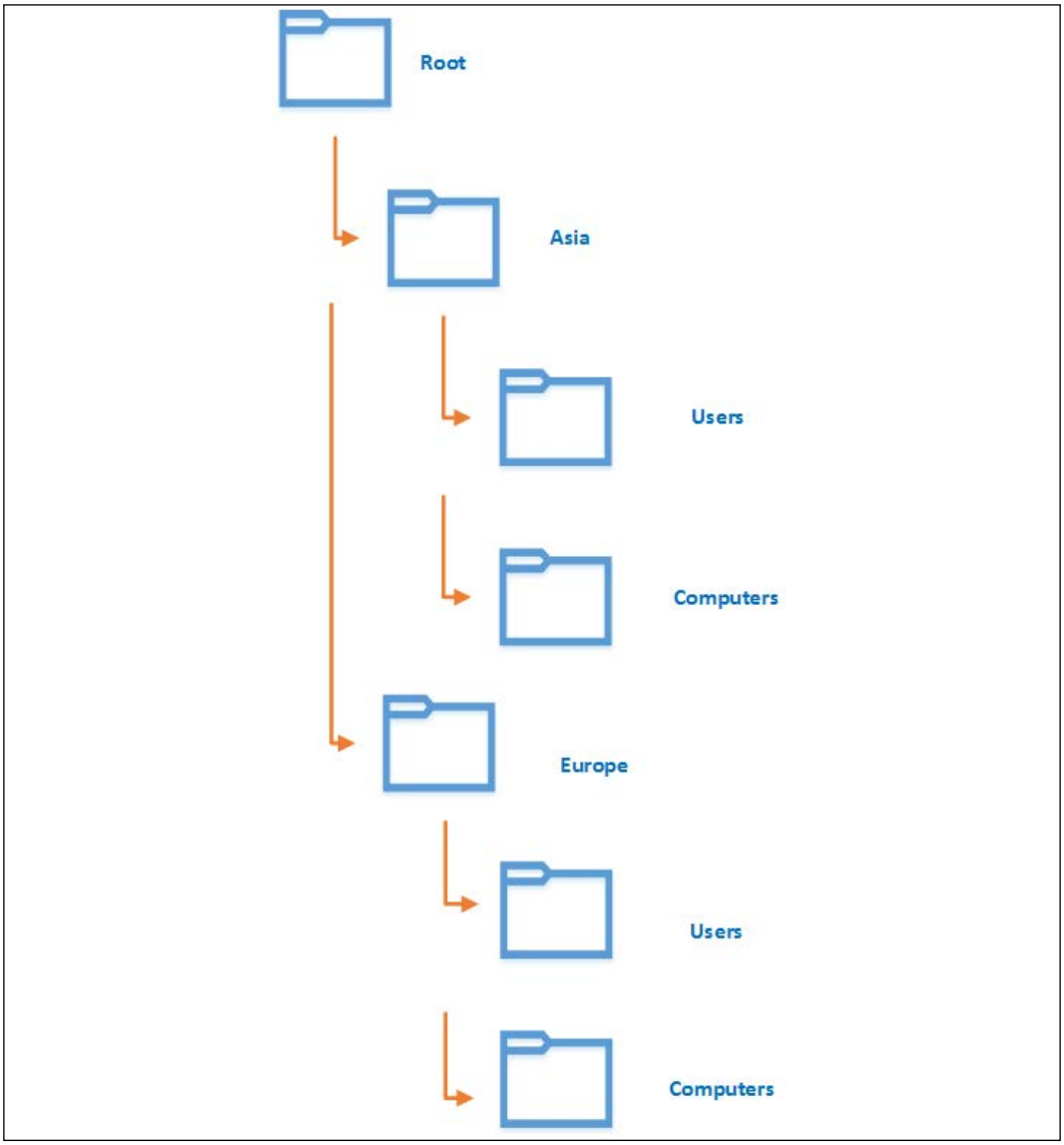


Figure 9.7: The geographical model

In the preceding diagram, the organization has two branches in Asia and Europe. The first level of the OUs was created based on geographical location and then it was further categorized based on the object types. **Asia** and **Europe** both have **Users** and **Computers** child OUs in their second level. In this model, on most occasions, each geographical location will follow the same structure in its child OUs. This allows you to delegate control for a group of administrators to manage branch office objects easily. This will improve infrastructure management and the productivity of IT operations.

The advantages and disadvantages of the geographical model are listed as follows:

Advantages	Disadvantages
Delegated control: The core value of this model is *easy delegated control*. Each object related to each branch is located in one structure, and it provides more control for administrators to delegate control.	**Extensibility**: Limited extensibility compared to the object type model. Most of the time, each branch structure should follow predefined standards. Therefore, if structural changes are required, the model will have limitations based on these standards.
Repetitive: Most of the time in this model, each branch office will have similar administrative, operational, and security requirements. Therefore, most of the Group Policy settings used in one branch will apply to another branch too.	**Operation limitations**: Each branch office IT team will have delegated control to manage the branch office's objects. But these privileges are limited. It is possible that, in order to perform certain tasks, they will still need to depend on the HQ IT team.
Maintaining standards: This model allows you to maintain administrative and security standards across the organization even if it has different branches. Even though branch IT teams have delegated control to perform certain tasks, privileged administrators can change or revoke these delegated controls.	N/A

The department model

Departments represent the hierarchical order of an organization as well as the categorization of responsibilities.

We can also use departments to categorize objects in Active Directory environments:

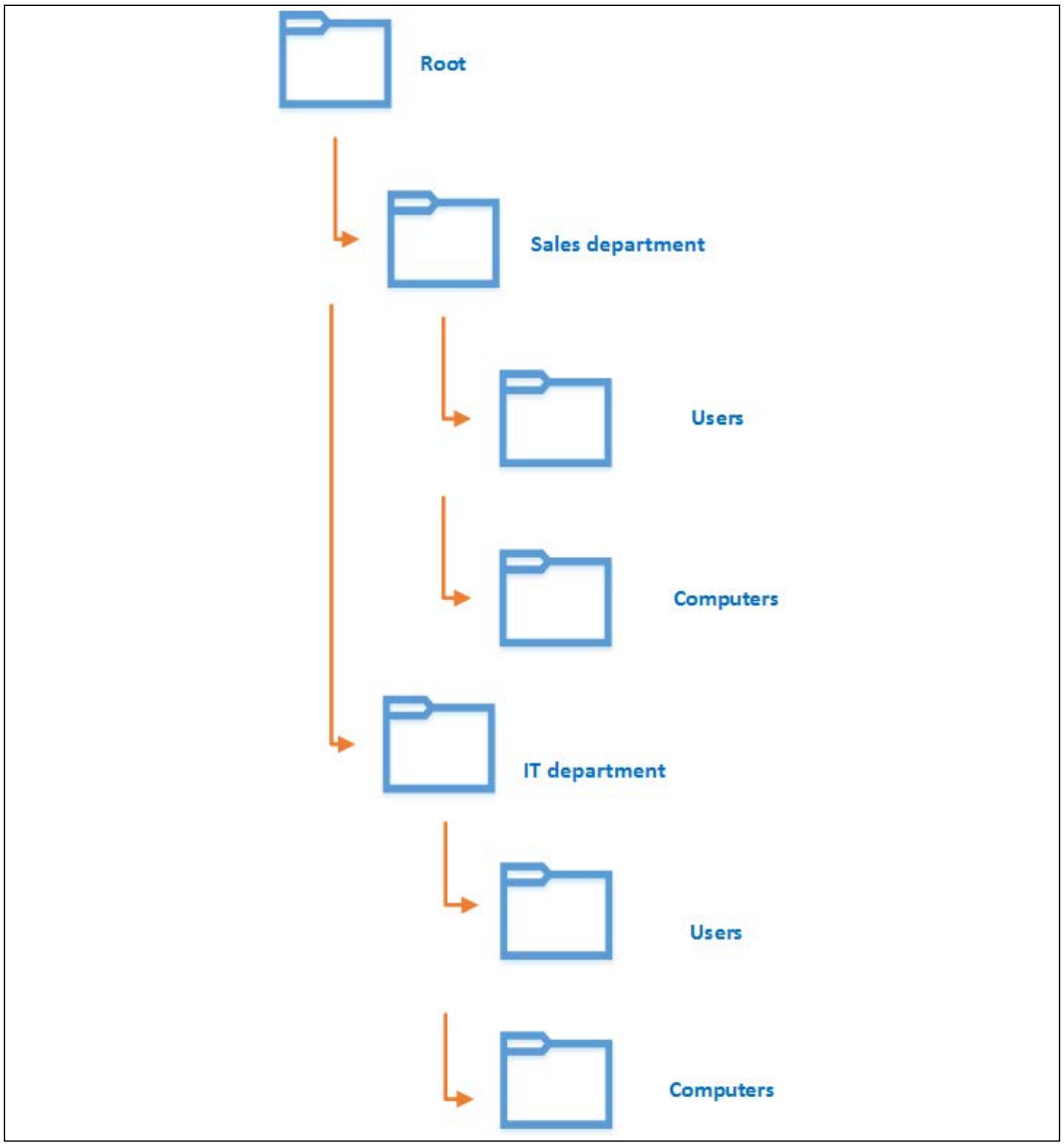


Figure 9.8: The department model

In the preceding diagram, the OU structure starts with the departments. Each department has its own OU, and its objects are further categorized based on the classes and responsibilities. This allows us to delegate control to objects in each department easily. As an example, managers in each department can be set up with delegated control to add or modify objects under their own departments.

The advantages and disadvantages of the department model are listed as follows:

Advantages	Disadvantages
Distributed delegated control: Under the department, objects are grouped together based on object classes and responsibilities. This allows administrators to delegate control to individuals or groups in order to manage objects in their own department's operation boundaries. Administrators do not need to change the structure based on delegation requirements as the model will group it the way it is needed by default.	**Object locations will be difficult to match with the structure**: Not all objects can match with this structure. In organizations, there are assets and services that are shared by different departments. As an example, printers, file servers, and mail servers are shared by all of the departments in the organization. This will need to be represented under the common OU, which is not going to match the organizational structure.
Minimum structural changes: Since the model matches the company's operational and structural designs, the changes to the OU structure will be minimal compared to other models.	**Limitation of applying company-wide settings**: In an organization, there are operational and security requirements that are not dependent on the departments. For example, an organization's data security policies, network security policies, and identity protection measures are common for every department and every user in the organization. Since objects are grouped based on department levels, we will have limitations in targeting objects to apply these different policies.
Less complex: The organization's operational structures (departments) are not complex usually. Departments have a well-defined structure to manage their assets and responsibilities. Since the OU structure is going to be almost a replica of that model, the OU structure will also be less complex to manage.	N/A

The hybrid model

So far in this chapter, we have been through a few different OU design models. Each of these has advantages as well as disadvantages. It doesn't mean that to create a good OU design, we need to use one of the given designs. These models can be used as a guideline to design your own structure based on operation requirements. We can use a few models together and create our own model if required. This is called the "hybrid model." There is no limit on how many different models you can mix or which way they have to be used.

It's all up to engineers to decide based on their operation requirements. Because of this "freedom," engineers use this model widely.

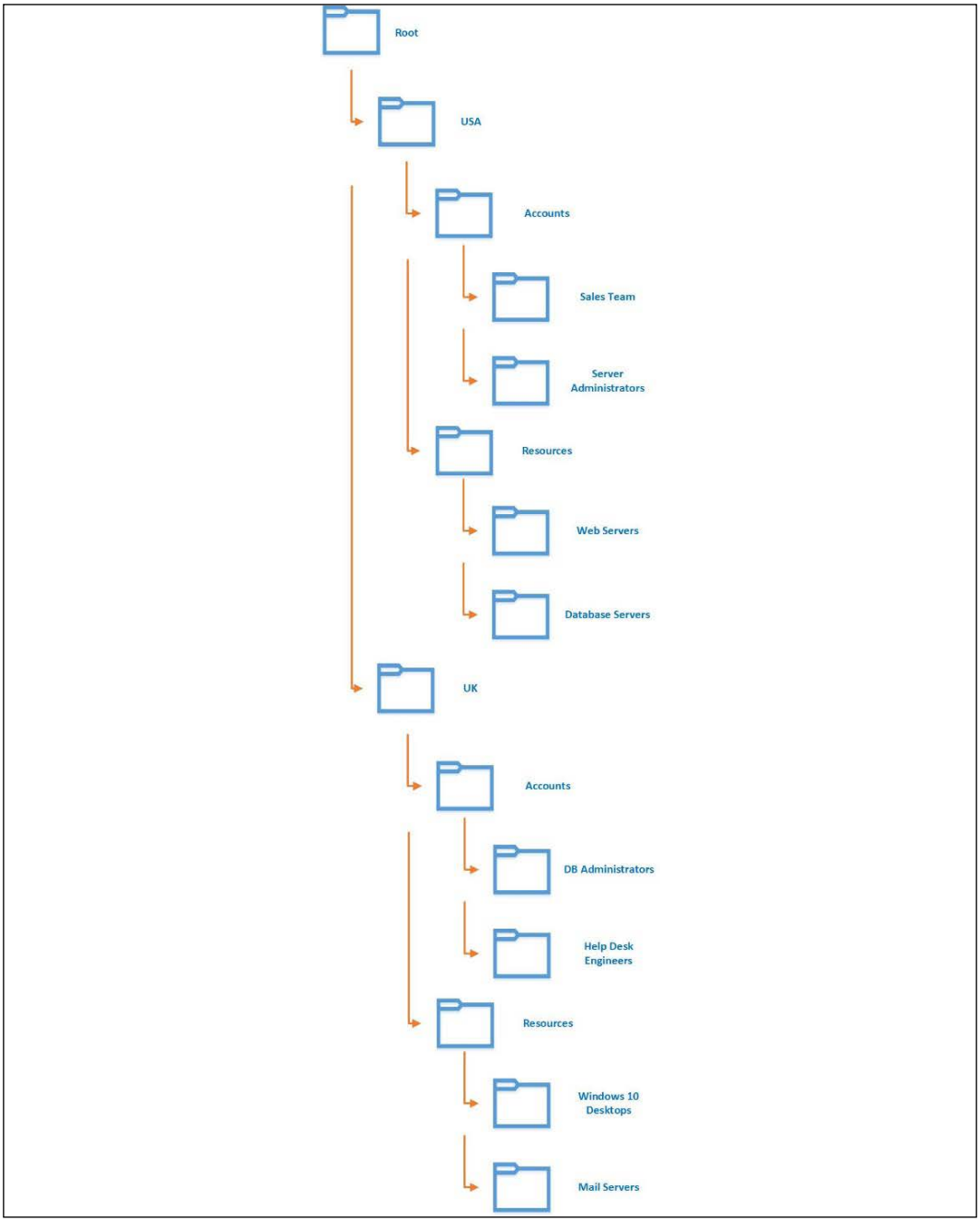


Figure 9.9: The hybrid model

In the preceding example, I am using two OU models, which are the geographical model and the functions model. At a high level, resources are categorized based on geographical level and then each location has Accounts and Resources OUs. Then, it is further broken down into different OUs based on the functions/responsibilities of the objects. As we can see, the hybrid model allows you to group objects in the most appropriate way.

Let's go ahead and evaluate the advantages and disadvantages of the hybrid model:

Advantages	Disadvantages
Flexibility: In this model, engineers have the freedom to mix different models together to create an OU structure. There are no rules in place to define what you can do or what you can't do. It is all up to engineers to plan and create their own model to address operation requirements.	**Can become over-complicated**: The freedom to use different models together can lead to a very complex OU hierarchy. There are no rules to follow and it can be purely up to the engineer's preferences.
Object grouping at a granular level: As this model allows you to use any number of OU design models, we can use it to group objects at a very granular level.	**Administrative overhead**: When an OU structure gets complex and objects are grouped at a very granular level, it takes a lot of resources to maintain/manage it.
Manage objects in an efficient way: As this model allows you to group objects at a granular level, we can use it to manage objects efficiently. As an example, if we use the department model, we place all the accounts and resources belonging to the department under one OU. But with the hybrid model, we can mix it with the functions model and group objects within a department by considering functions/responsibilities. This allows you to apply group policies or delegate control to smaller object groups.	

When it comes to the hybrid model, there is no right or wrong design. The freedom to mix different models even allows you to create your own model that is only valid for your organization. Also, even if there are design mistakes, we can change the OU structure with minimal impact on the business.

There is no limitation to how many sublevels OUs can have. Having more sublevels helps to categorize objects on a granular level, but at the same time, it will add to the complexity of managing the structure. This is especially effective in Group Policy management. Therefore, try to keep it under three sublevels.

Managing the OU structure

Similar to any other Active Directory object, the OU structure can be managed using **Active Directory Administrative Center** (**ADAC**), **ADUC MMC**, and PowerShell. In this section, I am going to demonstrate how to manage the OU structure using PowerShell.

Let's start this with a new OU. We can use the `New-ADOrganizationalUnit` cmdlet to create a new OU. The complete syntax can be reviewed using the following command:

```
Get-Command New-ADOrganizationalUnit -Syntax
```

As the first step, I am going to create a new OU called `Asia` to represent the Asia branch:

```
New-ADOrganizationalUnit -Name "Asia" -Description "Asia Branch"
```

In the preceding command, `-Description` defines the description for the new OU. When there is no path defined, it will create the OU under the root. We can review the details of the new OU using the following command:

```
Get-ADOrganizationalUnit -Identity "OU=Asia,DC=rebeladmin,DC=com"
```

We can add/change the values of OU attributes using the following command:

```
Get-ADOrganizationalUnit -Identity "OU=Asia,DC=rebeladmin,DC=com" |
Set-ADOrganizationalUnit -ManagedBy "Asia IT Team"
```

The preceding command will set the `ManagedBy` attribute to `Asia IT Team`.

When you use the ManagedBy attribute, make sure that you use an existing Active Directory object for the value. It can be an individual user object or a group object. If you don't use an existing object, the command will fail.

`ProtectedFromAccidentalDeletion` for the OU object is a nice safeguard we can apply. It will prevent accidental OU object deletion. This will be applied by default if you create an OU using ADAC, ADUC, or PowerShell:

```
Get-ADOrganizationalUnit -Identity "OU=Asia,DC=rebeladmin,DC=com" |
Set-ADOrganizationalUnit -ProtectedFromAccidentalDeletion $true
```

As the next step, I am going to create a sub-OU under the `Asia` OU called `Users`:

```
New-ADOrganizationalUnit -Name "Users" -Path
"OU=Asia,DC=rebeladmin,DC=com" -Description "Users in Asia Branch"
```

The preceding command will create an OU called `Users` under the `OU=Asia,DC=rebeladmin,DC=com` path. It is also protected from accidental deletion.

Now, we have the OU structure. The next step is to move objects to it. For that, we can use the `Move-ADObject` cmdlet:

```
Get-ADUser "tuser3" | Move-ADObject -TargetPath "OU=Users,OU=Asia,DC=re
beladmin,DC=com"
```

The preceding command will find the `tuser3` user and move the object to `OU=Users,OU=Asia,DC=rebeladmin,DC=com`.

We can also move multiple objects to the new OU:

```
Get-ADUser -Filter 'Name -like "Test*"' -SearchBase "OU=Users,OU=Europe
,DC=rebeladmin,DC=com" | Move-ADObject -TargetPath "OU=Users,OU=Asia,DC
=rebeladmin,DC=com"
```

The preceding command will first search all of the user accounts that start with `Test` in `OU=Users,OU=Europe,DC=rebeladmin,DC=com` and then move all of the objects it finds to the new OU path.

If you have **ProtectedFromAccidentalDeletion** enabled on the objects, it will not allow you to move the objects to a different OU. It needs to be disabled before the object is moved, because the move operation is actually a copy and a delete operation of the original object.

If we need to remove the OU object, it can be done using the `Remove-ADOrganizationalUnit` cmdlet:

```
Remove-ADOrganizationalUnit "OU=Laptops,OU=Europe,DC=rebeladmin,DC=com"
```

The preceding command will remove the `OU=Laptops,OU=Europe,DC=rebeladmin,DC=com` OU.

Delegating control

Administrators can delegate control based on OUs. This will provide control to individuals or groups to manage objects within OUs.

In my demonstration, I am going to provide delegated control for `Asia IT Team` members to manage objects under the `Asia` OU:

1. To do that, log in to the domain controller as the Domain Admin and open ADUC. Then, right-click on the relevant OU and click on **Delegate Control...**:

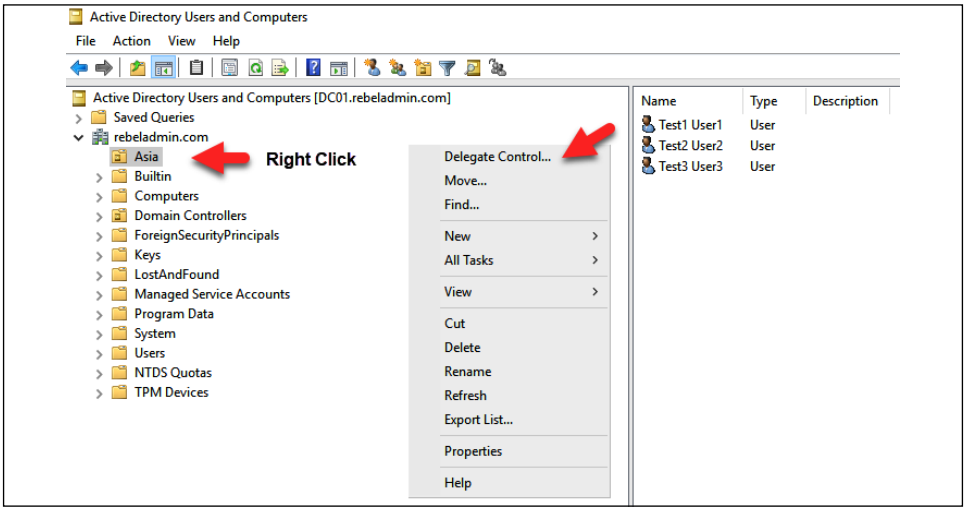


Figure 9.10: Delegate Control

2. Then, it will open up the wizard; there, select the individuals or groups that you'd like to provide delegated control to. In this demonstration, this is the **Asia IT Team (REBELADMIN\Asia IT Team)**:

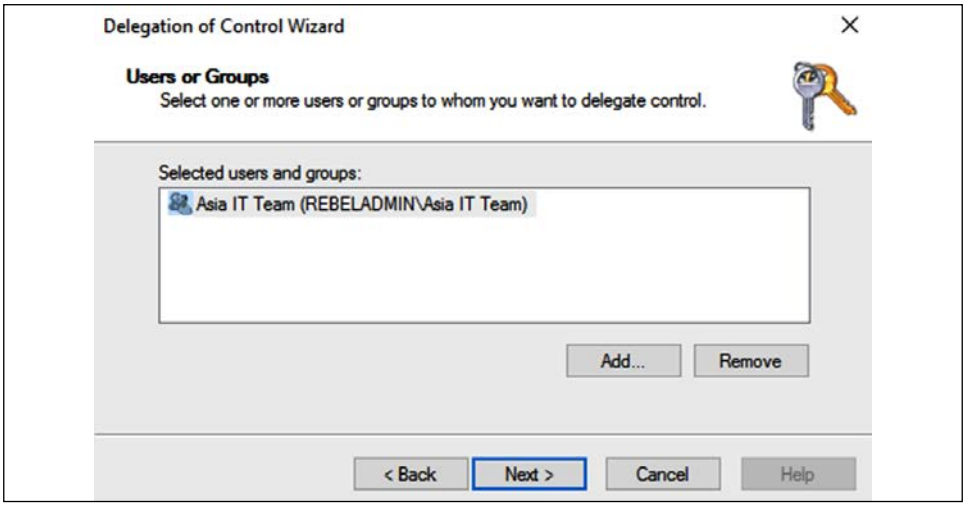


Figure 9.11: Delegate control – user/group selection

3. In the next window, the system will provide the option to select what kind of control to provide. These are sets of permissions predefined by Microsoft, and they cannot be changed. However, it provides the option to create custom tasks. After selecting the options you need, click **Next** to proceed:

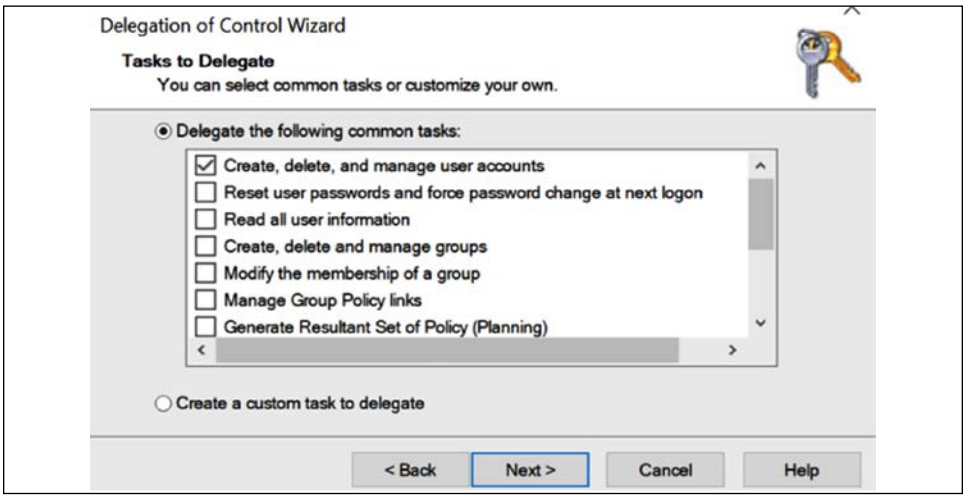


Figure 9.12: Delegate control – task selection

4. After the wizard completes the configuration, the team will have delegated control over the objects under `OU=Asia,DC=rebeladmin,DC=com`. We can review the delegated permission under the OU security settings:

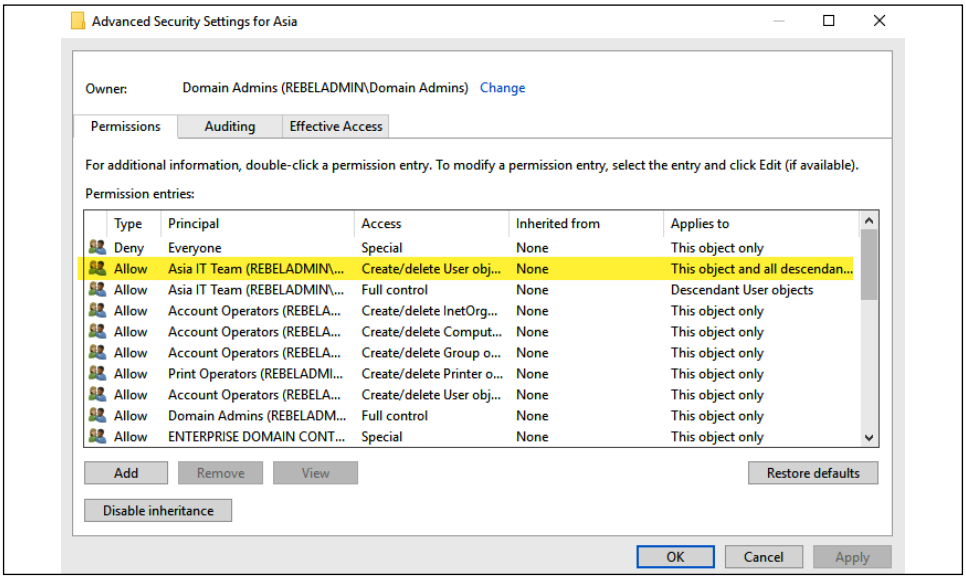


Figure 9.13: Advanced Security Settings

When required, delegated permission can be removed using the same window.

Note: Administrators can create a custom MMC for the Asia IT Team to manage the "Asia" OU. To do that, take the following steps:

1. Open the **Start** menu | select **Run** | type in `MMC` and hit *Enter*.
2. Then go to **File** | **Add-Remove Snap-in** | **Add ADUC**.
3. Expand the tree and select **Asia OU**.
4. Right-click on that OU and choose a new window from there.

This will open a new MMC window. Then go to **File** | **Save As** and save your MMC. After that, this new MMC can be used by a member of the Asia IT team to manage the "Asia" OU. It will only display the "Asia" OU in MMC.

Summary

OUs play a crucial role in Active Directory by allowing engineers to create a hierarchical structure within domain boundaries. This hierarchical structure should be created considering object management, delegating control, and applying group policies to manage applications, services, and security settings. In this chapter, we learned why OU design is important and what needs to be considered when designing the OU structure. After that, we moved on to different OU models that can be used as guidelines to design OU structures. At the end of the chapter, we learned how to manage OUs in the Active Directory infrastructure and how to delegate control for OUs.

In the next chapter, we will look at group policies, which are one of the core features of Active Directory.

10

Managing Group Policies

When I was preparing for this chapter, I was struggling somewhat as there is a lot to talk about in relation to group policies. Even though it was challenging, I tried my best to cover most of the key things related to group policies.

My council tax increased by 7% in April 2021. It is a rule, and whether I like it or not, I have to pay it every month. If not, I will have to face the consequences. This particular rule has a clear audience: it will only apply to houses under the Kingston council. We can consider Group Policy as an authority that executes a rule or set of rules against a clearly identified audience. This is similar to the council in my example.

It is impossible to describe the benefits of Active Directory (AD) without mentioning group policies. Group policies are one of the main reasons why Active Directory is so important in infrastructure management. Group policies are like a double-edged sword. They have lots of advantages as they help to manage various types of security, application, and system settings but, at the same time, if they have not been configured or used properly, this can affect things in many ways.

That said, in this chapter, we will cover the following topics:

- Benefits of group policies
- Understanding group policies and their capabilities
- Group Policy processing
- Group Policy inheritance
- Group Policy conflicts
- Group Policy filtering

- Loopback processing
- Guidelines for using group policies appropriately in the infrastructure
- Useful group policies

Before we dive into advanced topics about group policies, first we need to understand the benefits of group policies. Only then we can get the best out of it.

Benefits of group policies

A Group Policy has two types of settings: computer settings and user settings. Depending on the business and operation requirements, we will have to use both types of settings in policies. Let's go ahead and review what benefits group polices can deliver to a business.

Maintaining standards

I assume most of you have heard of **International Organization for Standardization** (**ISO**) standards. They allow organizations to run their operations in line with industry standards. Once an organization is complying with a relevant ISO standard, in return, a certification will be issued to prove the organization's commitment. Even though organizations have passed the ISO certification, the relevant authority will perform a yearly evaluation to make sure that they are *maintaining* standards continuously. Most companies follow these standards throughout the year, but for some, they come to attention only when the evaluation is due. This is because the implementation of these standards is easy, but maintaining them is challenging.

Our infrastructures are also subject to many standards. These standards can be based on the nature of the business, application and service best practices, business preferences, industry standards, and so on. Defining these standards within the organization is relatively easy when compared to the efforts required to maintain them. The best way to maintain the standards is to enforce them. As an example, a common security standard for a password is to use *complex* passwords. But whenever you're asked to define a password, by nature, people go for the easiest password they can remember. In the system, however, if we can *enforce* the practice of not accepting a non-complex password, users do not have a choice. They must comply with the password standards in order to define a password.

Group Policy allows us to enforce infrastructure standards. Group Policy can be based on user standards or computer standards. This policy ensures that companies will comply with standards with minimum maintenance efforts because once policy settings are applied, the target group of users or devices will not be allowed to opt out. It's almost like converting a standard into a *rule.*

Automating administration tasks

I started my career as a web developer and later moved on to system administration. My job title at that time was *associate system administrator*. One of the common service requests I used to receive from the development team manager was to deploy different software to engineers' computers. There were around 20 people in the company, and since I was a *junior* person, it was always my responsibility to install this software on computers. It was painful, but since it was just 20 computers, it was just about manageable. But imagine if it was hundreds of computers; I would be spending days doing boring and repetitive tasks.

From a company point of view, this is still at the cost of operations. This was just an example, but a regular helpdesk usually gets requests that are small but repetitive, such as mapping shared folders, printer installations, and customized application settings. Group policies allow engineers to automate these types of common and repetitive administration tasks. As an example, group policies can be used to push application installations, push new printer deployments, and map drives when the user logs in. This reduces the cost of operations and allows us to allocate IT resources for more important tasks.

Preventing users from changing system settings

By default, the Windows system has a software firewall to protect machines from threats, but sometimes, this prevents access to certain application traffic. Most of the time, the reaction to this will either be to disable the firewall or to allow application traffic via the custom rule. But if someone changes the firewall settings without the IT team's knowledge, it can possibly put the entire infrastructure at risk.

Group policies allow you to take control of these sensitive settings and prevent end users from making changes. As well as firewall settings, group policies can also be used to prevent the modification of services, prevent access to control panel features, prevent changes to applications, and much more.

Flexible targeting

At the beginning of this section, we learned about how group policies can be used to enforce standards. In a business, some of these standards apply to the entire organization, and others can be specific to departments, business units, or users/device groups. Group policies allow us to apply different policies based on Active Directory sites, domains, organization units, or groups.

As an example, we can use group policies to push two different firewall settings to IT department computers and Sales department computers.

We can create two group policies to cover the previous requirements and link them to a relevant organization unit. This flexible targeting allows us to apply different system settings for specific audiences.

No modifications to target

Group Policy also uses the server-client architecture. Active Directory domain controllers hold the group policies and process them when a client device's starts up or a user logs in. To apply Group Policy to a client, we do not need to do any system or profile modifications. If the target meets the Group Policy target criteria, it will process the Group Policy settings. This architecture simplifies the Group Policy processing as there are fewer dependencies.

Group Policy capabilities

Group Policy can be used to perform many different tasks in an infrastructure. Here, I have listed some of these capabilities:

- Group Policy can be linked to sites, domains, and organization units. This allows us to match the Group Policy requirements with the Active Directory structure. However, Group policies cannot be applied to default containers in Active Directory.
- Group Policy allows us to use security filtering to target specific groups, users, or computers.
- **Windows Management Instrumentation** (**WMI**) filters are capable of filtering the AD objects based on criteria such as the OS version, roles, and system configuration. Group Policy allows us to use WMI filters for targeting.
- The **Group Policy Object** (**GPO**) status can change based on operational requirements. If required, group policies can disable group policies completely, or disable user or computer settings individually.
- Group Policy management tasks can be delegated to individuals or groups.
- Group Policy can be used to install, redeploy, or remove programs from computers.
- Group Policy can be used to publish scripts to be executed upon computer startup, or to shut down a process.

- Group Policy can be used to deploy printers to computers.
- Group Policy is capable of applying different security policies, such as password policies, lock-out policies, Kerberos policies, firewall policies, and public key policies.
- Group Policy can be used to define system audit settings and enable/manage advanced system audit capabilities, which allow you to capture more data regarding the roles and their activities.
- Group Policy can be used to add registry keys to systems.
- Group Policy can be used to define software restriction policies and application control policies to control the application behaviors in computer systems.
- Group Policy can be used to set up policy-based QoS rules to define **Differentiated Services Code Point** (**DSCP**) and throttle values for the outgoing network traffic. It will allow you to prioritize/manage network traffic in a system.
- Group Policy administrative templates can be used to define the registry-based policies' targeting system, applications, and service settings.
- Group Policy can be used to apply preference settings to computers and users. For example, it can be used to define mapped drives, printers, power options, Internet Explorer settings, regional settings, local users and groups, and so on.
- Using Group Policy, it is possible to manage end user roaming profile settings, including folder redirection. It will automatically save user data to a network location instead of a local computer. It allows you to access the same profile data from any workstation in the domain.

Apart from the above listed capabilities, group policies have a lot more other capabilities that can be used to improve IT operations. Later on in this chapter, I will explain how to use some of these capabilities along with Group Policy examples.

Group Policy objects

When new Active Directory objects are added, the system saves the object data inside the Active Directory database. However, the way GPO is stored is different from the typical Active Directory object. GPO contents are stored in two locations (the Active Directory database and the `SYSVOL` folder) in the Active Directory environment.

The Group Policy container

As with any other object, the Active Directory database also holds GPO information. This information is more related to system settings and the path reference for the other dataset. When a GPO is created, as with any other Active Directory object, it will also have the **Globally Unique Identifier** (**GUID**) value. This is important as this value is used by both datasets to refer to each other. This value is used in the **Common Name** (**CN**), too. Before we look into datasets, we need to find the GUID value for the GPO. This can be done using the following command:

```
Get-GPO -name "Test Users"
```

The preceding PowerShell command will list the default properties of the `Test Users` GPO:

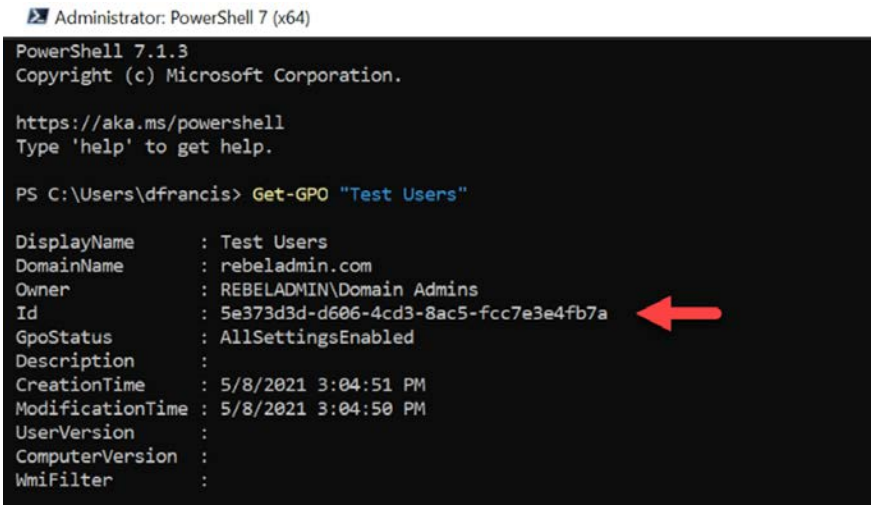


Figure 10.1: GUID value of GPO

In the preceding screenshot, the `Id` attribute represents the GUID value of the `Test Users` GPO.

Now that we have the GUID information, the next step is to review the **Group Policy Container** (**GPC**) information for the given GPO. This information can be accessed using ADSI Edit MMC or `Ldp.exe`. On this occasion, let's use `Ldp.exe` to review the data.

To open `Ldp.exe`, type `Ldp.exe` into the domain controller's **Run** box. Then, go to the **Connection** menu and select **Bind**. Select the default options if the logged-in account has relevant privileges (such as schema or Enterprise Admin). In the next step, click on **Tree** under **View** and select the **DN** domain.

As an example, in my demonstration, the **DN** value is `DC=rebeladmin,DC=com`. GPC values are located under `CN=Policies,CN=System,DC=rebeladmin,DC=com`:

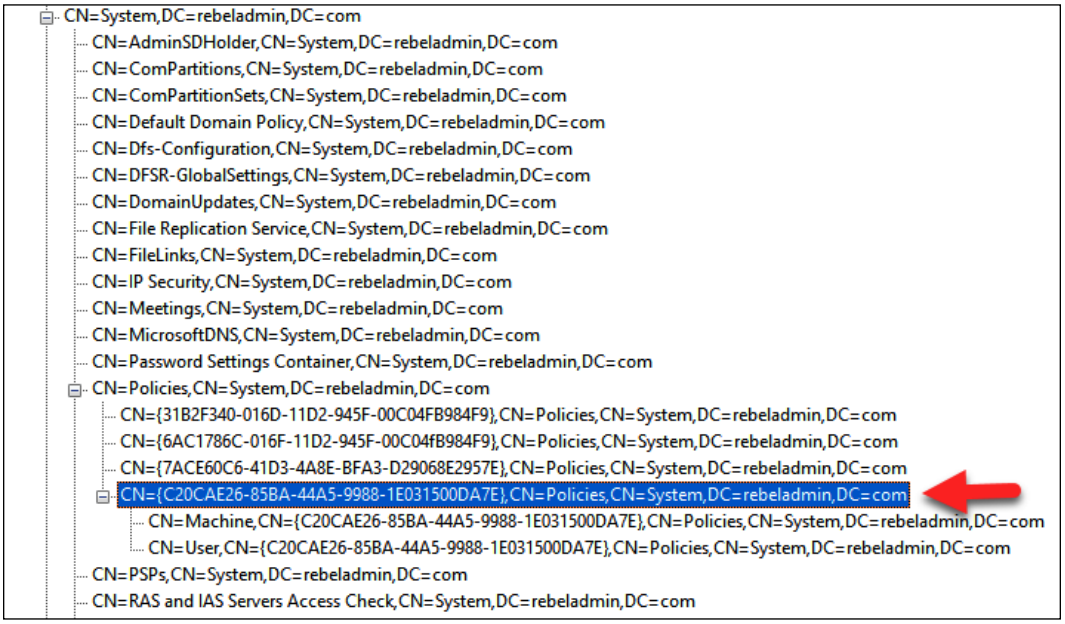


Figure 10.2: GPC information for policy

The policy object is further divided into two sections, which represent the computer configuration (machine) and the user configuration (user):

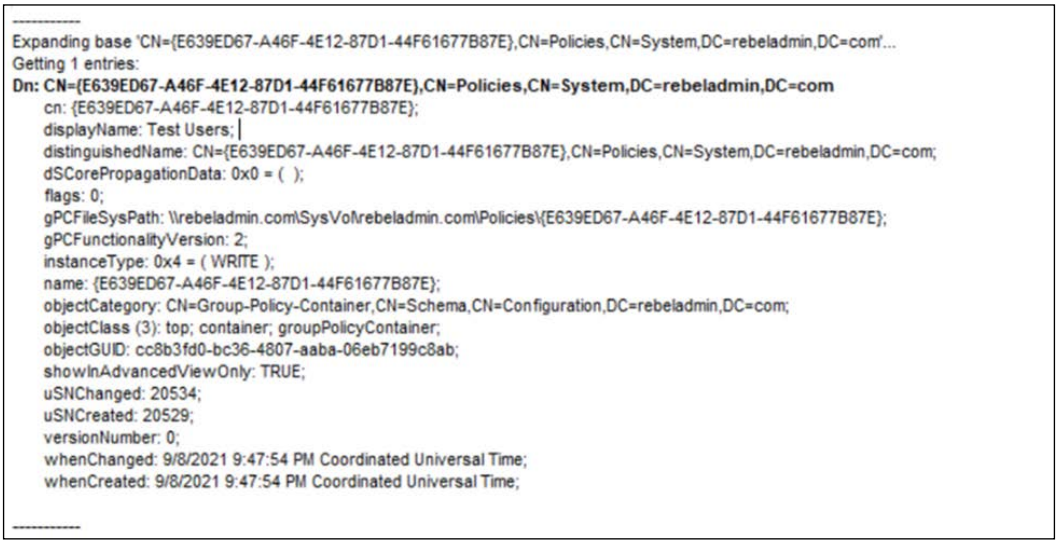


Figure 10.3: GPC information for policy – computer and user configurations

The preceding screenshot shows details about the GPO we selected, including certain attribute values:

- `displayName`: This attribute contains the name of the Group Policy, which is defined during the GPO setup process.
- `gPCFileSysPath`: This attribute value represents the path to the other dataset of the GPO. This is called the Group Policy template path. It is always available under the `SYSVOL` folder.
- `gPCMachineExtensionNames`: This attribute lists all of the **client-side extensions** (**CSEs**) that need to process the GPO computer settings. These are all listed with GUIDs, and `https://bit.ly/3l6IBWV` can be used as a reference to most of the known CSEs.

As mentioned at the beginning, GPO data is stored in two locations. In this section, we looked into data stored in the Active Directory database, and next, we are going to look into data saved in the `SYSVOL` folder.

The Group Policy template

When we looked at the Group Policy capabilities, we saw that policies can be used to publish applications, run start up/shutdown scripts, and more. From the Active Directory object's point of view, all of these policy settings are attributes and all scripts and files used in the policy need to be saved in a centralized location for processing. Instead of saving them in an Active Directory database, the system saves all of these policy-related files and settings in the `SYSVOL` folder. The default path for the **Group Policy template** (**GPT**) data is `\\rebeladmin.com\SYSVOL\rebeladmin.com\Policies`. Here, `rebeladmin.com` can be replaced with your domain's **fully qualified domain name** (**FQDN**).

Inside the policy folder, there are two subfolders called `Machine` and `User`. These folders contain files and settings related to the GPO's computer configuration and user configuration. There is another file called `GPT.INI`, which contains the version number of the group policy. In every Group Policy edition, the version number will increase automatically and it will be used as a reference to sync the Group Policy changes from one domain controller to another.

If healthy replication is in place, the version numbers should be equal:

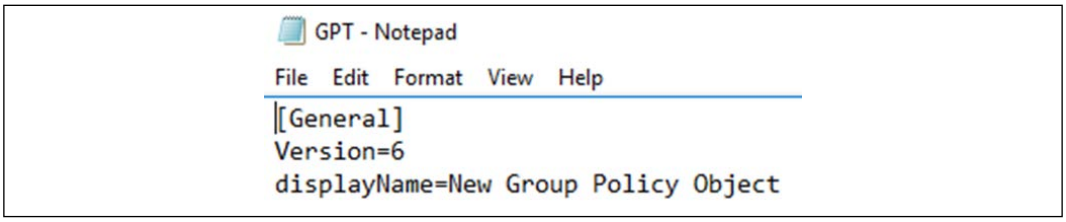


Figure 10.4: GPT.ini file

Inside the `Machine` and `User` folders, there are several folders that will include the data according to GPO configurations. As an example, the `Applications` folder will contain the software installation files if the GPO is set to publish applications, while the `Scripts` folder will contain any start up/shutdown scripts that are published via the GPO:

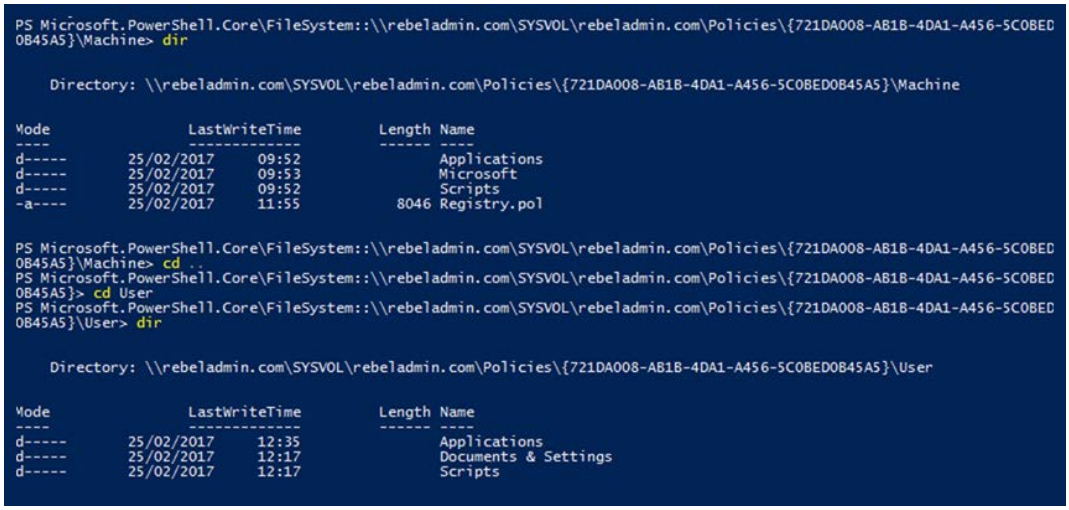


Figure 10.5: GPO folder structure

To process a Group Policy successfully, GPT data synchronization between domain controllers is crucial. `SYSVOL` replication issues will impact GPO policy processing. Before Windows Server 2008 domain services, `SYSVOL` replication used **File Replication Service** (**FRS**). After AD DS 2008, this was replaced by **Distributed File System** (**DFS**), which provides more efficiency and redundancy in terms of replication.

Group Policy processing

Evaluating Group Policy requirements may identify some settings that are common for objects in the entire domain. But at the same time, some settings may be unique to specific departments or groups. Any Group Policy that is applied at the root level will be inherited by other organization units by default. Therefore, organization units can have inherited group policies as well as directly linked group policies.

If multiple group policies are applying to an organization unit, in which order will it be processed? Will it prevent the processing of any Group Policy? If the same setting is applied to different policies, which one will be applied? To answer all of these questions, it's important to understand how Group Policy processing works.

There are two main types of policies in the Active Directory environment:

- **Local policies**: Windows systems are supported to set up local security policies. These policies are different from domain GPOs, and they contain limited features that are more focused on security settings. These are applied to any user who logs in to the system.
- **Non-local policies**: These policies are Active Directory-based policies that we will discuss throughout this chapter. These policies only apply to domain-joined computers and Active Directory users. These policies are feature-rich compared to local policies.

Group Policy can be applied to three levels in the Active Directory environment:

- **Site**: Group Policy can be linked to an Active Directory site. Any site-level Group Policy will apply to all domains on that site.
- **Domain**: Any Group Policy that is applied at the domain level will apply to all users' and computers' objects under that domain. By default, the system creates a Group Policy called **Default Domain Policy** at the domain level. Most of the time, domain-level policies will be used to publish security settings that apply to the entire infrastructure.
- **Organization units** (**OUs**): Group Policy at the OU level is applied to any user or computer object under it. By default, the system creates an OU-level Group Policy called **Default Domain Controllers Policy**, which is applied to the domain controller's OU. Group Policy settings applied at the OU level are more specific as the target audience is smaller compared to the site or domain.

The following diagram illustrates the policy levels in Active Directory:

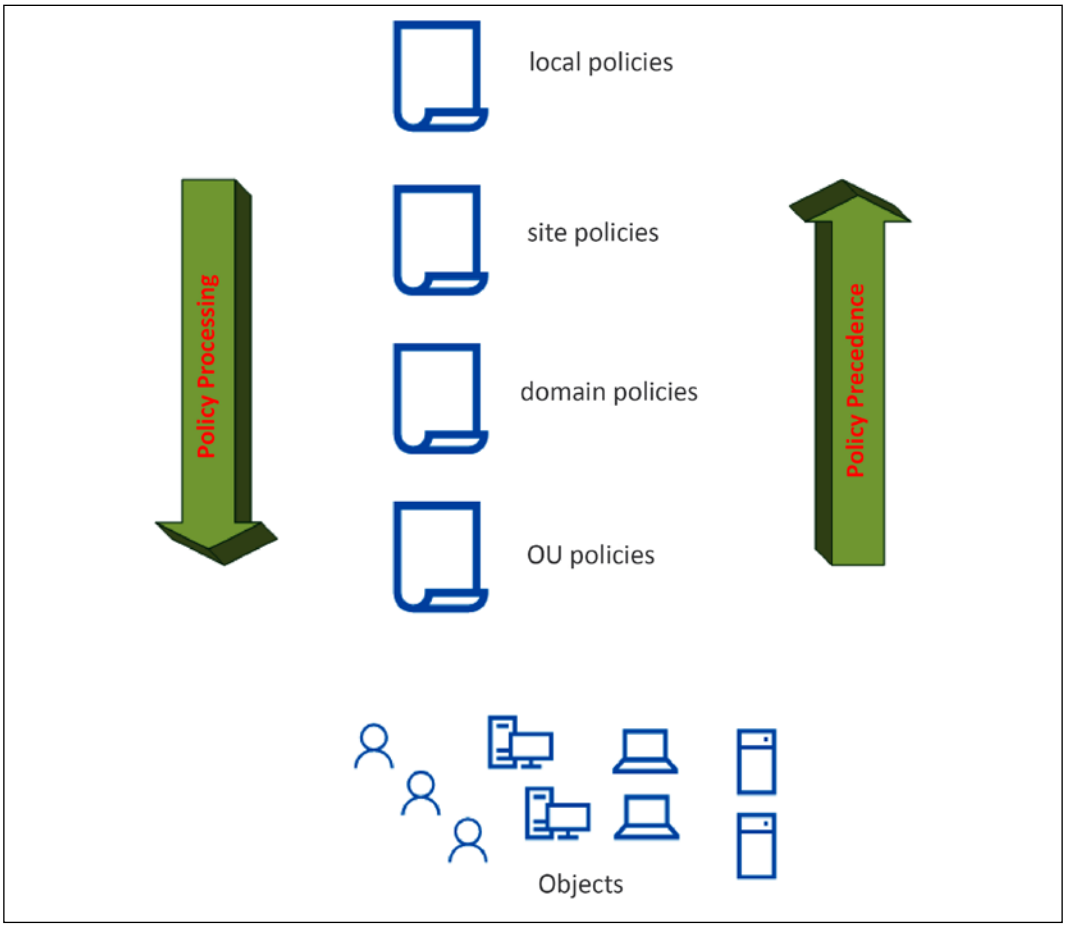


Figure 10.6: Group Policy processing order

In the Active Directory environment, group policies will be processed in the following order:

- Local policies
- Site policies
- Domain policies
- OU policies

In general, this order is called **LSDOU**. The first policy to be processed is the local policy, and the last one to be processed is the OU policy. This doesn't mean that the most preferred policy is the local policy. Of all four policies, the policy that holds the lowest precedence value will be the winning GPO, which is the OU policy. This processing order is important when policies have conflicting values because the policy that is closest to the object will be the winning policy. The policy at the OU level will be the *closest* policy to the object.

As I said earlier, OU-level policy settings are more specific according to organizational requirements. Therefore, they hold the most accurate policy settings for the targeted objects. This order is the default processing order and, based on the requirements, it is possible to change the precedence order. We will look into this later on in this chapter.

The processing of local policies can be completely turned off if required. This will prevent the application of non-recommended settings that administrators do not have control over or awareness of. This can be disabled by using a Group Policy setting. The policy setting is located at `Computer Configuration\Administrative Templates\System\Group Policy\Turn off Local Group Policy Objects Processing`. To disable policy processing, the value should be set to `Enable`.

To apply computer Group Policy settings successfully, the device should have a valid communication with a domain controller, and to apply user Group Policy settings, a user (the domain account) should log in to the domain-joined computer that can communicate with a domain controller.

Group policies are mainly processed in two different modes. By default, a Group Policy's computer settings will start to process during the OS startup. Once the user enters their username and credentials, the Group Policy's user settings start to process. This pre-processing mode is called **foreground processing**. During the foreground process, some policies will finish processing, but some will not. They will be processed in the **background** once login and network connections have been initialized. Also, once the user logs in, every 90 minutes, the group policies will run in the *background* by default. This is the second mode of Group Policy processing.

Foreground processing can be further divided into two sub-modes. Before Windows XP was introduced, all of the policy settings that ran in foreground mode will process before the user sees the desktop window. This is called **synchronous** mode. But after Windows XP, the default mode of processing is **asynchronous**, which means that the OS does not wait until Group Policy processing.

There are four policy settings that are defined by Microsoft, which are always processed in asynchronous mode. If any policy has folder redirection, software installation, disk quota, and drive mapping enabled, it will be processed in synchronous mode. Not only that, but all of the start up scripts will also run in the foreground in synchronous mode.

This default processing behavior provides a faster login time for users, but, at the same time, some policy settings can take up to two login cycles to apply the changes fully. This behavior impacts the security settings. As an example, if we want to block access to control panel settings, and if the policy settings are not processed in synchronous mode, when the user logs in, they can have access to the control panel. Then, it is not going to provide the expected results. This default mode is useful when users are connected through a *slow link*. Otherwise, it can take an awfully long time to finish the Group Policy processing. If there is no specific reason to use asynchronous mode, it is recommended that you always use synchronous mode. We can force this using Group Policy; the policy settings are located at `Computer Configuration\Administrative Templates\System\Logon\Always wait for the network at computer startup and logon`.

As we saw earlier, there are four Group Policy settings defined by Microsoft that always run in synchronous mode. Even if systems are connected via a slow link, they will process these Group Policy settings in synchronous mode. Therefore, it is recommended that you enable the following two Group Policy settings to force asynchronous mode when you log in via a slow link and remote desktop services. Asynchronous mode for slow links can be enabled via `Computer Configuration\Administrative Templates\System\Group Policy\Change Group Policy Processing to run asynchronously when a slow link is detected`. Asynchronous mode for remote desktop services can be enabled via `Computer Configuration\Administrative Templates\System\Group Policy\Allow asynchronous user Group Policy processing when logging on through Remote Desktop Services`.

Group policy processing has a direct impact on group policy inheritance. Based on group policy requirements, sometimes we have to disable the inheritance. Let's go ahead and explore Group Policy inheritance in more detail in the next section.

Group Policy inheritance

Any Group Policy that is applied to the upper level of the structure is inherited in the lower level. The order of the inherited policies is decided by the LSDOU model that we looked at in the *Group Policy processing* section. Group Policy inheritance for each OU can be reviewed using the **Group Policy Management MMC**.

To view the inheritance data, first, click on the OU and then click on the **Group Policy Inheritance** tab:

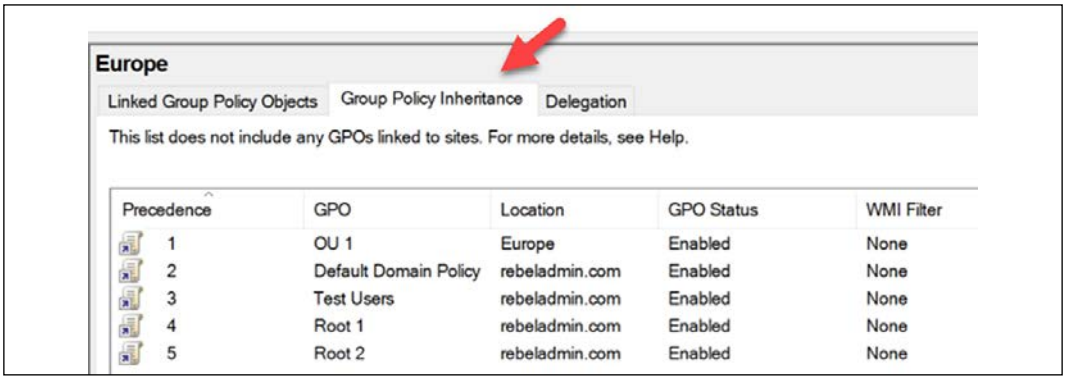


Figure 10.7: Group Policy Inheritance

The Group Policy inheritance details can also be viewed by using the `Get-GPInheritance` PowerShell cmdlet. As an example, the same information listed in the preceding screenshot can be viewed using the following command:

```
Get-GPInheritance -Target "OU=Users,OU=Europe,DC=rebeladmin,DC=com"
```

In this example, I have one site-linked Group Policy called **Site 1**. There are two domain-linked group policies called **Root 1** and **Root 2**. I also have an OU-linked Group Policy called **Test Users**:

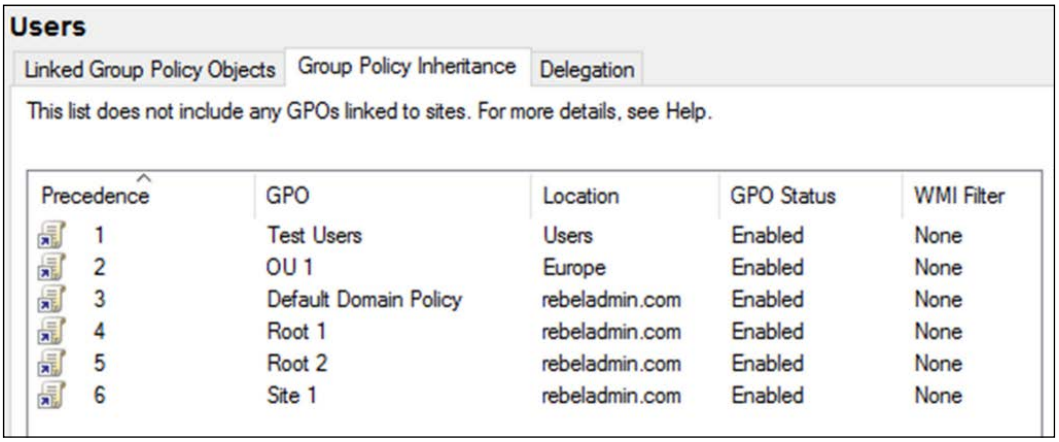


Figure 10.8: Group Policy precedence

In the list, the first precedence goes to the **Test Users Group Policy**, which is the OU-linked Group Policy. Then, precedence order **2** goes to the Group Policy linked to the parent OU. Precedence orders **3**, **4**, and **5** go to the domain-linked group policies.

The last in the order is the **Site 1 Group Policy**, which is linked to the Active Directory site.

This inheritance is not useful in every scenario. There can be situations where the OU needs to have very specific settings and should not be disturbed by any other inherited policy. When the number of group policies increases, the time they need for processing also increases. If my requirements can be achieved using one Group Policy that is linked to the OU, why should I still apply inherited policies that are not going to help me anyway? In a similar scenario, Group Policy inheritance can be blocked at the OU level. This can be done by using the **Group Policy Management MMC** or the `Set-GPinheritance` PowerShell cmdlet:

```
Set-GPInheritance -Target "OU=Users,OU=Europe,DC=rebeladmin,DC=com"
-IsBlocked Yes
```

The preceding command will block the Group Policy inheritance in `OU=Users,OU=Europe,DC=rebeladmin,DC=com`:

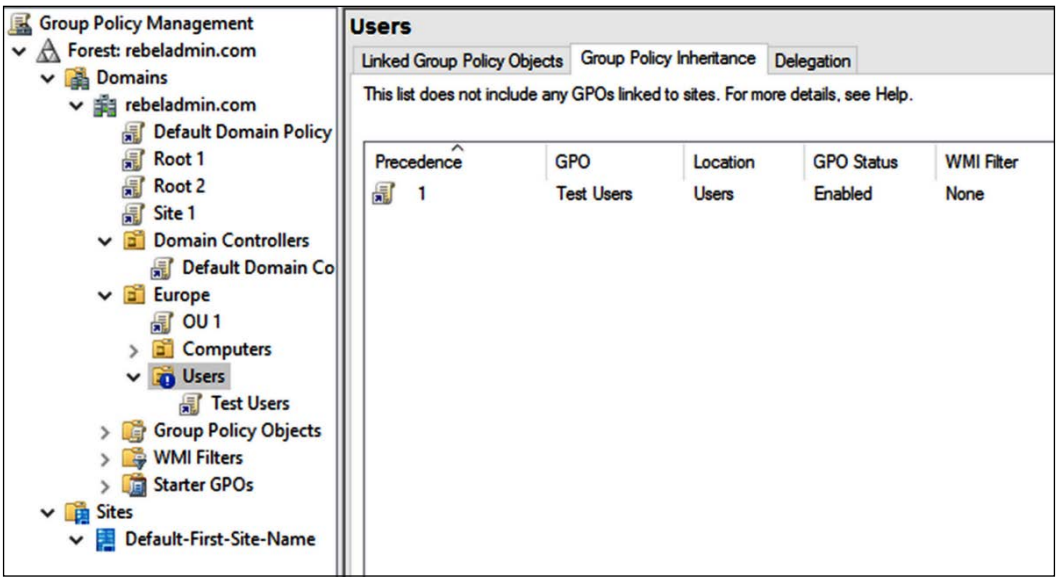


Figure 10.9: Block Group Policy Inheritance

After the Group Policy inheritance block, the only policy in the inheritance list is the policy that is linked to the OU. Also, when we block inheritance, we can see a blue exclamation mark on the OU. That's the easiest way to identify whether group policy inheritance is blocked for an OU.

Group Policy conflicts

The precedence order of group policies in LSDOU and Group Policy inheritance also decide which policy will win when we have some conflicting settings. Let's look at this further with the help of an example:

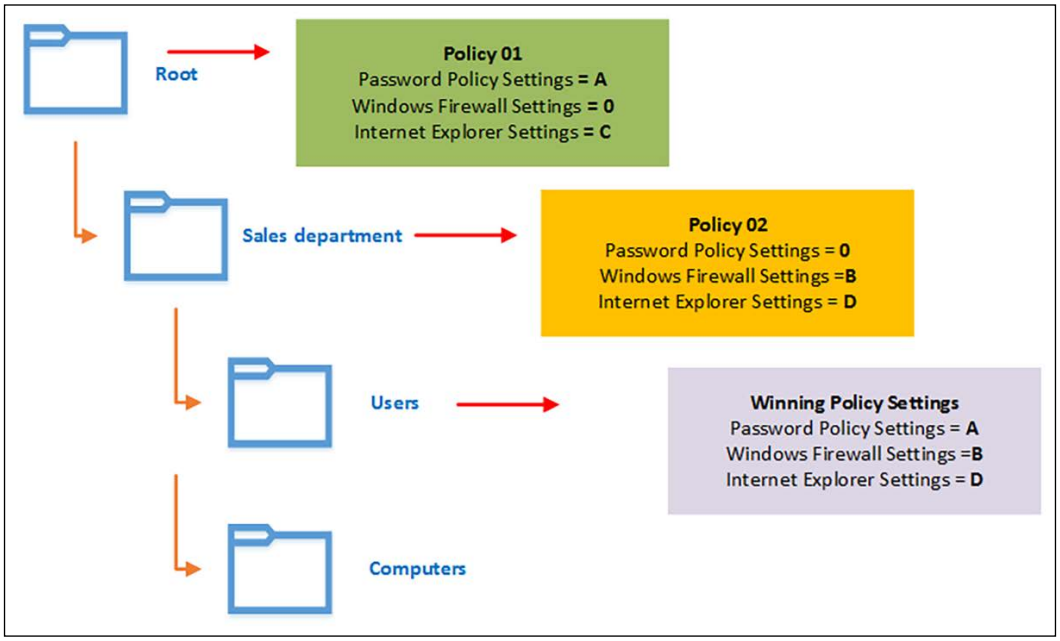


Figure 10.10: Group Policy conflicts example

As per the preceding example, we have two policies inherited by the **Users** OU. **Policy 01** is the domain-linked Group Policy, and **Policy 02** is the OU-linked Group Policy. Each Group Policy has its own values defined for the three selected settings. Based on the default Group Policy inheritance, the **Users** OU will have both policies applied. According to LSDOU, **Policy 02** will have the lowest precedence value as it is the closest policy to the **Users** OU. For **Password Policy Settings**, only **Policy 01** has a value defined. Therefore, even though it's the least preferred Group Policy, that value will apply to the **Users** OU. For **Windows Firewall Settings**, only **Policy 02** has a value. The same policy will also apply to the **Users** OU. When it comes to **Internet Explorer Settings**, both policies have values, which creates a conflict. The winning conflicting policy setting value will be decided based on LSDOU. Therefore, the winning value will be from **Policy 02**.

Microsoft allows you to change this default policy-winning procedure by *enforcing* policies. When a Group Policy has been enforced, it will have the lowest precedence value regardless of where it's been linked. Another advantage of the enforced policy is that it will apply even though the OU has blocked inheritance.

If the domain-linked policy is enforced, it will apply to any OU under the domain, and it will hold the lowest precedence. If multiple policies are enforced, all of them will take the lowest precedence numbers in order.

To enforce a policy, load **Group Policy Management Console** (**GPMC**), right-click on the selected Group Policy, and then select the **Enforced** option. This will enforce the policy and change the policy icon with a small padlock mark. This allows you to identify enforced policies quickly from the policy list:

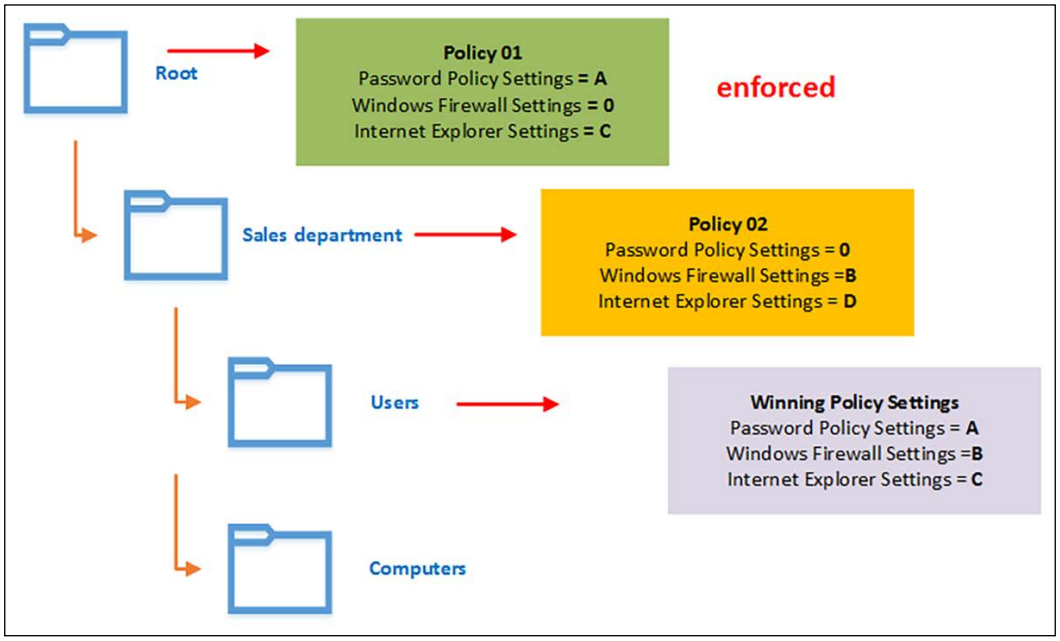


Figure 10.11: Enforced Group Policy

In the preceding example, **Policy 01** is **enforced**. It is the domain-linked Group Policy. Under normal circumstances, **Policy 02** would get the lowest precedence value (for **Users** OU). But when the policy is **enforced**, **Policy 01** will have the lowest precedence value. When we look into the winning policy values of the **Users** OU, for **Password Policy Settings**, the system will process the **Policy 01** value as it is the only one with any value defined. For **Windows Firewall Settings**, **Policy 01** does not have any value defined. So, even if a policy has been **enforced**, the winning policy setting will be from **Policy 02** as it's the only one with a value defined. **Policy 01** and **Policy 02** both have values for **Internet Explorer Settings**, but the **enforced Policy 01** is on top of the policy list and the winning policy setting will be from it.

So far, we have talked about conflicting policy settings from different levels in the domain structure. How will this work if it's at the same level? Policies at the same level also apply according to the precedence order.

When policies are at the same level, the LSDOU process is applicable. The winning policy will be decided based on its position in the policy list. The order of the list is decided based on the **Linked Group Policy Objects** list. This list can be viewed using the **Linked Group Policy Objects** tab in the OU detail window in GPMC:

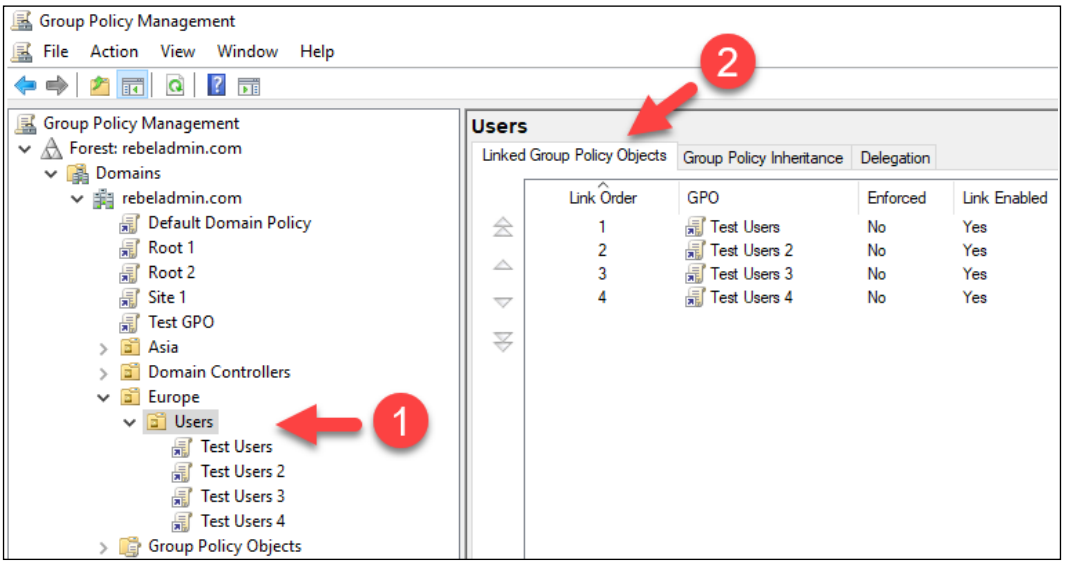


Figure 10.12: Group Policy processing at the same level

The order of policies in the same level can be changed by using two methods. One method is to enforce the policy. When the policy is enforced, it will take priority over the other policies at the same level, but it will not change the **link order** of the policy. The order of the list can be changed using the up and down buttons in the **Linked Group Policy Objects** tab. The link order will match the precedence order of the group policies:

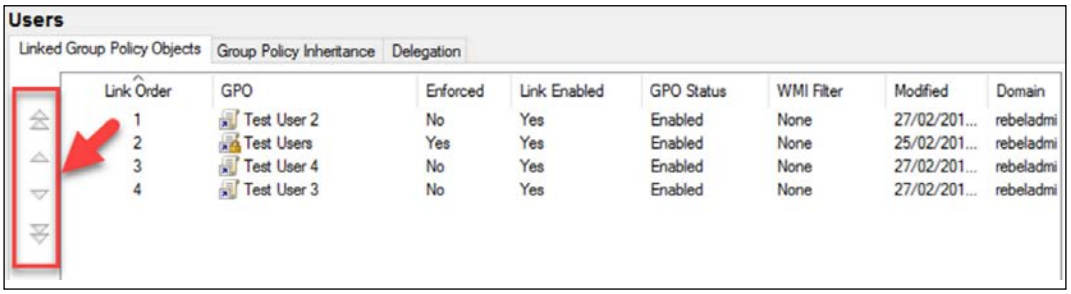


Figure 10.13: Group Policy Link Order

On rare occasions, you may also have conflicting settings between the computer configuration and the user configuration within the same policy. In such a situation, computer settings will always win.

Group Policy conflicts that occur in Active Directory environments are mostly due to unplanned GPO implementations and a lack of knowledge regarding the group policy processing order. Throughout this chapter, you can find relevant information that will help you to organize group policies properly and avoid conflicts.

Group Policy mapping and status

There are a few things we need to consider when we create and link a Group Policy object to a site, domain, or OU:

- If it's a new GPO, it can be created directly under the relevant OU or domain using GPMC.
- In sites, it's only allowed to link to an existing GPO. Therefore, if a new GPO needs to link to the site, first we need to add a new GPO using GPMC or a PowerShell cmdlet.
- An already added GPO can link to any OU, domain, or site. As an example, if policy A is created and linked under OU A, it can be reused in any other OU, domain, or site.

A new GPO object can be created using the `New-GPO` PowerShell cmdlet:

```
New-GPO -Name GPO-Test-A
```

The preceding command will create a GPO called `GPO-Test-A`. By default, it will not link to any OU, domain, or site. In GPMC, it can be viewed under the `Group Policy Objects` container.

Once an object is created, it can be linked to an OU, domain, or site by using the `New-GPLink` cmdlet:

```
New-GPLink -Name GPO-Test-A -Target "OU=Users,OU=Europe,DC=rebeladmin,
DC=com"
```

The preceding command links a GPO called `GPO-Test-A` to `OU=Users,OU=Europe,DC=rebeladmin,DC=com`.

Both cmdlets can be combined to create and link a GPO at the same time:

```
New-GPO -Name GPO-Test-B | New-GPLink -Target "OU=Users,OU=Europe,
DC=rebeladmin,DC=com"
```

The preceding command will create a new GPO called `GPO-Test-B` and link it to `OU=Users,OU=Europe,DC=rebeladmin,DC=com` at the same time.

There are occasions where a link to the group policies needs to be disabled. This is useful when you do Group Policy troubleshooting. When the link to the Group Policy is disabled, Group Policy will be removed from the Group Policy precedence list. However, it will not remove Group Policy from the Link Order list or from its location. This can be done via GPMC or by using the `Set-GPLink` PowerShell cmdlet:

```
Set-GPLink -Name GPO-Test-B -Target "OU=Users,OU=Europe,DC=rebeladmin,
DC=com" -LinkEnabled No
```

The preceding command will disable the link between the `GPO-Test-B` GPO and `OU=Users,OU=Europe,DC=rebeladmin,DC=com`. This is usually used to temporarily disable a policy. It can be enabled at any time by using the `-LinkEnabled Yes` option.

However, if this requirement is permanent, this GPO link can be completely removed by using the `Remove-GPLink` cmdlet. This will remove the link, but it will not delete the GPO. It will also not affect any other existing links in the GPO:

```
Remove-GPLink -Name GPO-Test-B -Target "OU=Users,OU=Europe,DC=rebeladmi
n,
DC=com"
```

This command will remove the `GPO-Test-B` policy from `OU=Users,OU=Europe,DC=rebeladmin,DC=com`. It will remove the GPO from the **Precedence** list, as well as the **Link Order** list.

If the GPO needs to be deleted completely, we can use the `Remove-GPO` cmdlet for that:

```
Remove-GPO -Name GPO-Test-A
```

The preceding command will delete the aforementioned GPO completely from the system. If the GPO was linked, it will forcefully remove it at the same time.

Without disabling links, removing links, or deleting the GPO, it is also possible to disable GPO settings only. This feature provides options to disable computer settings, user settings, or both.

Once the settings are disabled, it will not remove them from the **Link Order** or **Precedence** list. It will only disable the applied settings:

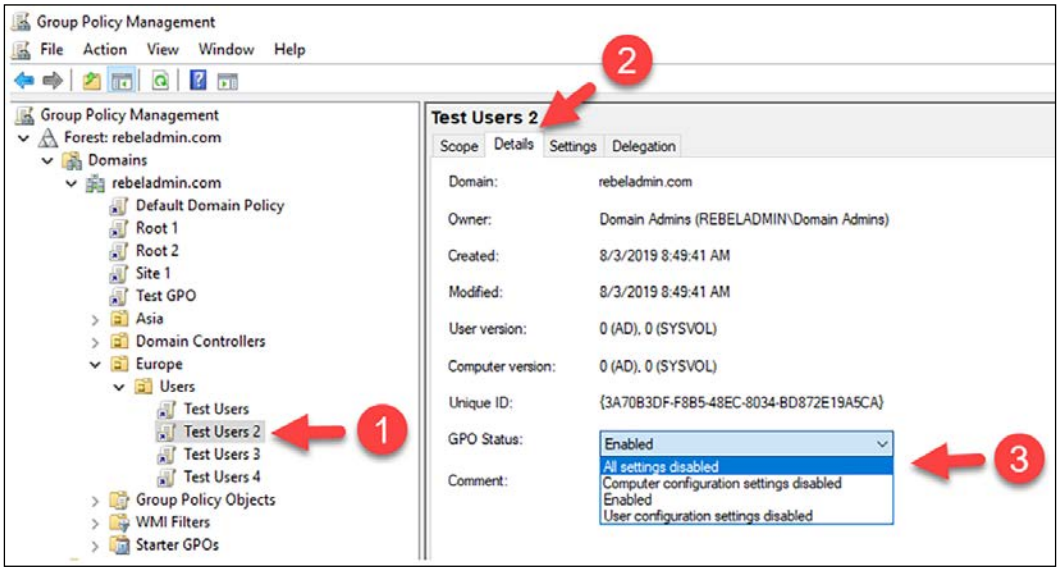


Figure 10.14: Group Policy status

In the next section, we are going to learn how group policies can be used to manage applications and service settings even if they have new updates.

Administrative templates

Group policies allow us to manage computer settings and user settings effectively. However, infrastructure operation requirements are changing frequently. As an example, this can be due to new application versions, new security requirements, new business policy requirements, and more. We know that group policies can be used to manage organization-wide settings, but not all of the requirements will be supported by default. By the time AD DS 2008 was released, it was not possible to have group policies that could manage settings in the Office 2016 application as it didn't exist.

So how can we manage new applications and new OS settings using group policies? Do we need to keep updating Active Directory? No, we don't. We can do this using administrative templates. Application vendors and developers can develop administrative templates and publish them via group policies to customize, manage, and optimize their products and services.

Administrative templates contain registry-based changes to the system. **Administrative Templates**, under **Computer Configuration** in the GPO, can be used to modify the registry entries under the `HKEY_LOCAL_MACHINE` key.

Administrative Templates, under **User Configuration** in the GPO, can be used to modify the registry entries under the `HKEY_CURRENT_USER` key. Microsoft also has pre-deployed administrative templates that can be used to edit more than 1,000 individual registry settings. Administrative templates are not just from the application vendors; if needed, we can also create custom administrative templates.

Before Windows Server 2008, administrative templates came as Unicode-formatted text files with the `.adm` file extension. These did have some drawbacks, however. It saves the `.adm` file as part of the Group Policy inside `SYSVOL`. If it is used in different GPOs, it will save a copy of `.adm` with every GPO. It increases the size of `SYSVOL`. Also, if you need to change a setting in the `.adm` file, it needs to be edited in each copy of `.adm`. One `.adm` file has support for only one language; if you need to use multiple languages, you need a copy of `.adm` for each language.

After Windows Server 2008, administrative templates were presented as two XML files. The file with the `.admx` extension is responsible for publishing registry settings and the file with the `.adml` extension is responsible for providing language-specific interface settings. This allows administrators to edit and manage policies in multiple languages. Unlike `.adm` files, if you need to edit the `.admx` file, it needs to be edited in one place only, and it will map the relevant `.adml` file to provide multi-language support if required.

The `.adm` file is part of the GPO, so it's always stored in the `SYSVOL` folder. However, by default, for ADMX/ADML, Group Policy holds the settings only for the policy, and when it needs editing, it will pull the ADMX and ADML files from the local workstation. It is possible to change this behavior and save these files in a central location. This provides easy access and better management for administrative templates.

We can move administrative templates to the `SYSVOL` folder by using the following steps:

1. Log in to the domain controller as the **Domain Admin** or higher.

2. Create a folder called `PolicyDefinitions` under `\\rebeladmin.com\SYSVOL\rebeladmin.com\Policies`. The `rebeladmin.com` domain can be replaced by your own domain FQDN:

```
mkdir \\rebeladmin.com\SYSVOL\rebeladmin.com\Policies
\PolicyDefinitions
```

3. After that, copy the policy definition data into this new folder:

```
Copy-Item C:\Windows\PolicyDefinitions\*
\\rebeladmin.com\SYSVOL\rebeladmin.com\Policies
\PolicyDefinitions -Recurse -Force
```

This will also move ADML files into `\\rebeladmin.com\SYSVOL\rebeladmin.com\Policies\PolicyDefinitions` with their language name. As an example, US English will be in `\rebeladmin.com\SYSVOL\rebeladmin.com\Policies\PolicyDefinitions\en-US`. The language to use in policy editing will be decided based on the language used in the workstation. The `"PolicyDefinitions"` folder in `SYSVOL` is also referred to as "Central Store."

Group Policy filtering

A Group Policy can map to sites, domains, and OUs. If a Group Policy is mapped to the OU, by default, it will apply to any object under it. But within an OU, domain, or site, there are lots of objects. Sometimes, we may have to target certain objects in the OU, domain, or site without changing the current Active Directory structure. Group Policy filtering capabilities allow us to further narrow down the Group Policy targets to security groups or individual objects.

There are a few different ways to perform filtering in Group Policy:

- Security filtering
- WMI filtering

Both of these methods have their own characteristics. It is up to engineers to choose the best method based on the filtering requirement.

Security filtering

Before you apply security filtering, the first thing to check is whether the Group Policy is mapped correctly to the site, domain, or OU. The security group or the objects you are going to target should be at the same level as where the Group Policy is mapped.

We can use the GPMC or PowerShell cmdlets to add security filtering to a GPO:

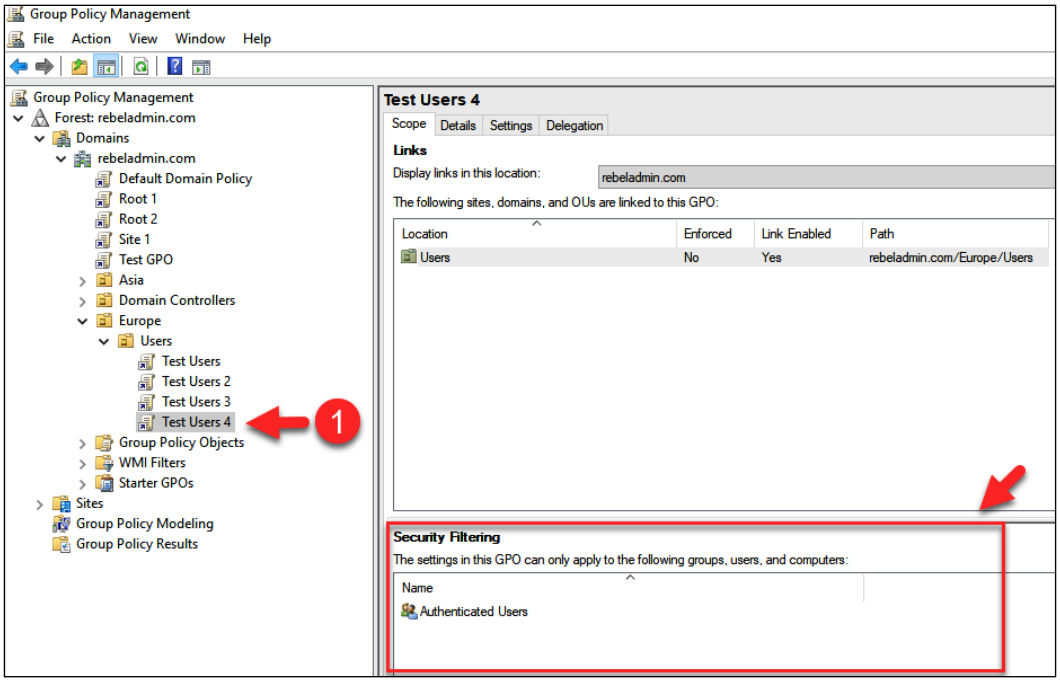


Figure 10.15: Security Filtering – default settings

As you can see, by default, every policy has an "Authenticated Users" group under the **Security Filtering** section. This means that, by default, the policy will apply to any authenticated user in that OU. When we add any group or object to security filtering, it also creates an entry under delegation. To apply a Group Policy to an object, it needs a minimum of the following permissions:

- Read
- Apply group policy

Any object added to the **Security Filtering** section will have both of these permissions set by default. In the same way, if an object is added directly to the delegation section of the policy and applies both permissions, that object will appear in the **Security Filtering** section.

Before we add custom objects to the **Security Filtering** section, we need to change the default behavior of the security filtering with authenticated users. Otherwise, it doesn't matter what security group or object we add—Group Policy settings will still be applied to any authenticated user. Before Microsoft released security patch MS16-072 in 2016, we could simply remove the **Authenticated Users** group and add the required objects to it.

With these new security patch changes, group policies will now run within the computer's security context. Earlier, group policies were executed within the user's security context. To accommodate these new security requirements, one of the following permissions must be available on the Group Policy's **Delegation** tab:

- Authenticated Users: Read
- Domain Computers: Read

To edit these changes:

1. Go to the Group Policy.
2. Then, go to the **Delegation** tab.
3. Click on **Advanced**, select **Authenticated Users**, and then remove the **Apply group policy** permissions:

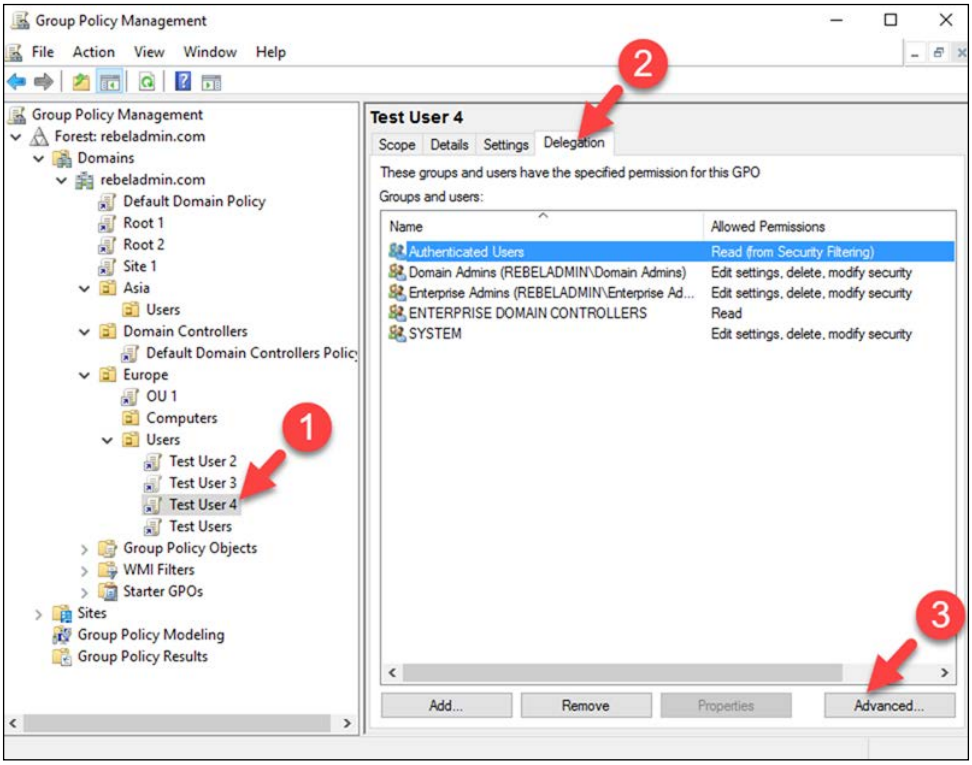


Figure 10.16: Group Policy delegation

4. Click on the **Scope** tab.

5. Add the required security group or objects to the **Security Filtering** section.

   It will automatically add the relevant **Read** and **Apply group policy** permissions:

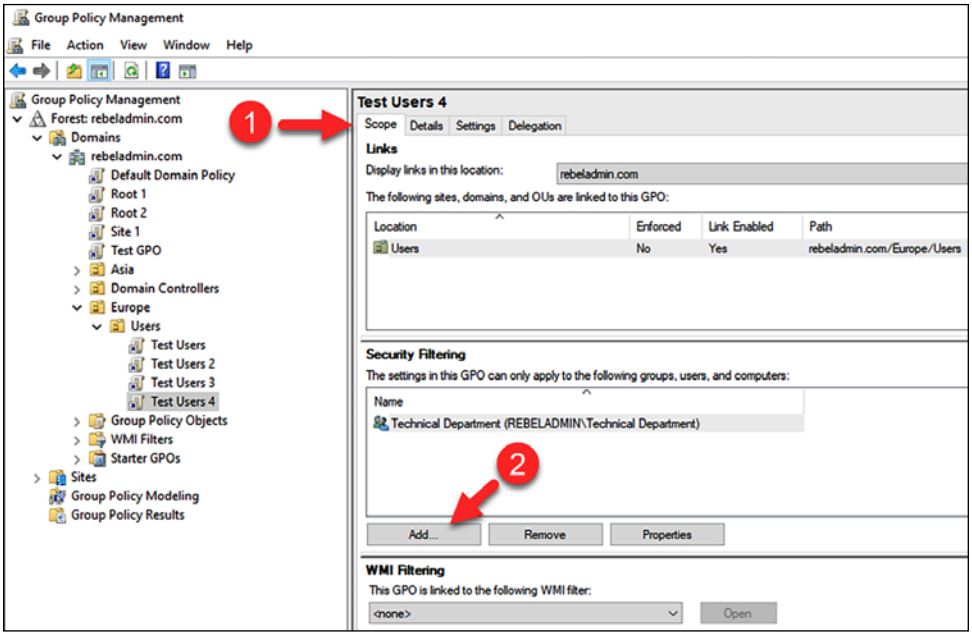


Figure 10.17: Group Policy Security Filtering with security groups

Here, although we are looking at how to apply group policies to a specific target, we are also allowed to explicitly apply policies to a large number of objects and then *block* groups or objects. As an example, let's assume we have an OU with a few hundred objects from different classes. From all of these, we have 10 computer objects that we do not need to apply to a given Group Policy. Which one is the easiest? Go and add every security group and object to **Security Filtering**, or allow the Group Policy for everyone and only block access for one security group.

Microsoft also allows you to use the second method in filtering. To do that, Group Policy should have default security filtering, which is Authenticated Users with the **Read** and **Apply group policy** permissions. Then, go to the **Delegation** tab and click on the **Advanced** option. In the next window, click on the **Add** button and select the group or object that you need to block access to:

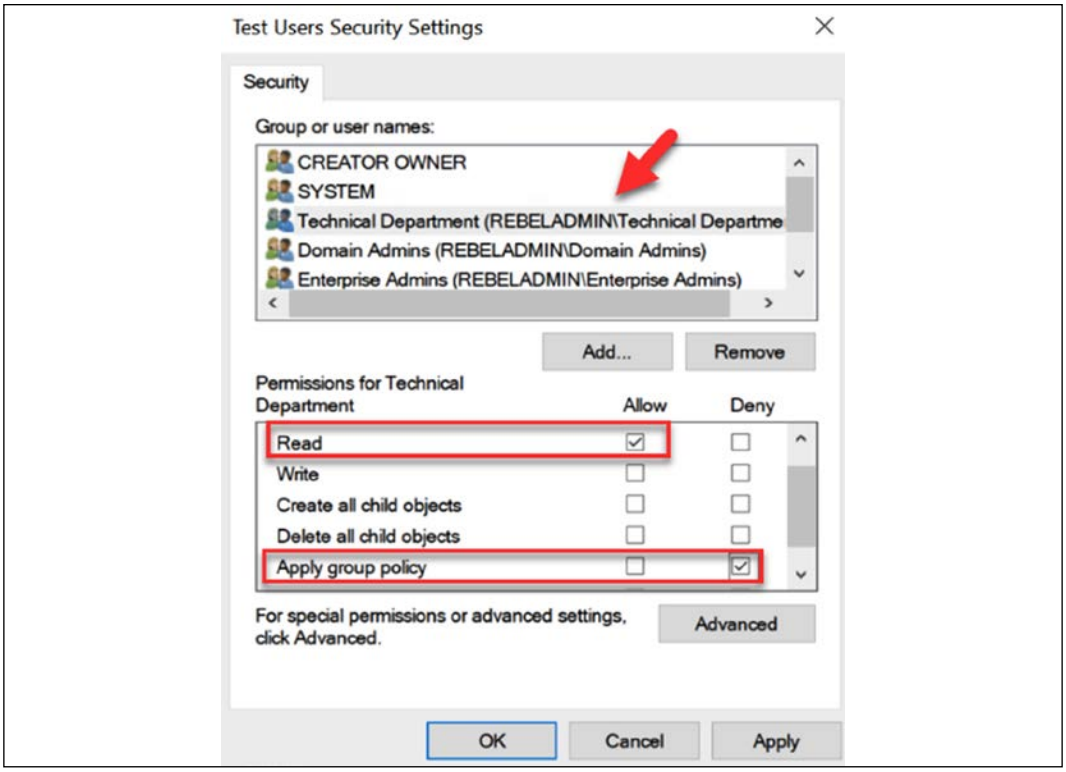


Figure 10.18: Group Policy permissions

Here, we are denying **Apply group policy** permissions to an object, while all other objects under that OU will still be able to read and apply the Group Policy.

WMI filtering

WMI filters are another method that we can use to filter the Group Policy target. This method can be used to filter computer objects only and is based on computer attribute values. As an example, WMI filters can be used to filter different OS versions, processor architecture (32 bit/64 bit), Windows Server roles, registry settings, event IDs, and more. WMI filters run against the WMI data of the computer and decide whether it should apply the policy. If it matches the WMI query, Group Policy will be processed. This method was first introduced with Windows Server 2003.

We can use GPMC to create/manage WMI filters. Before applying a filter to a GPO, first, we need to create it. A single WMI filter can be attached to many GPOs, but a GPO can only have a single WMI filter attached.

To create a WMI filter, open **GPMC**:

1. Right-click on **WMI Filters**.
2. Click on **New...**:

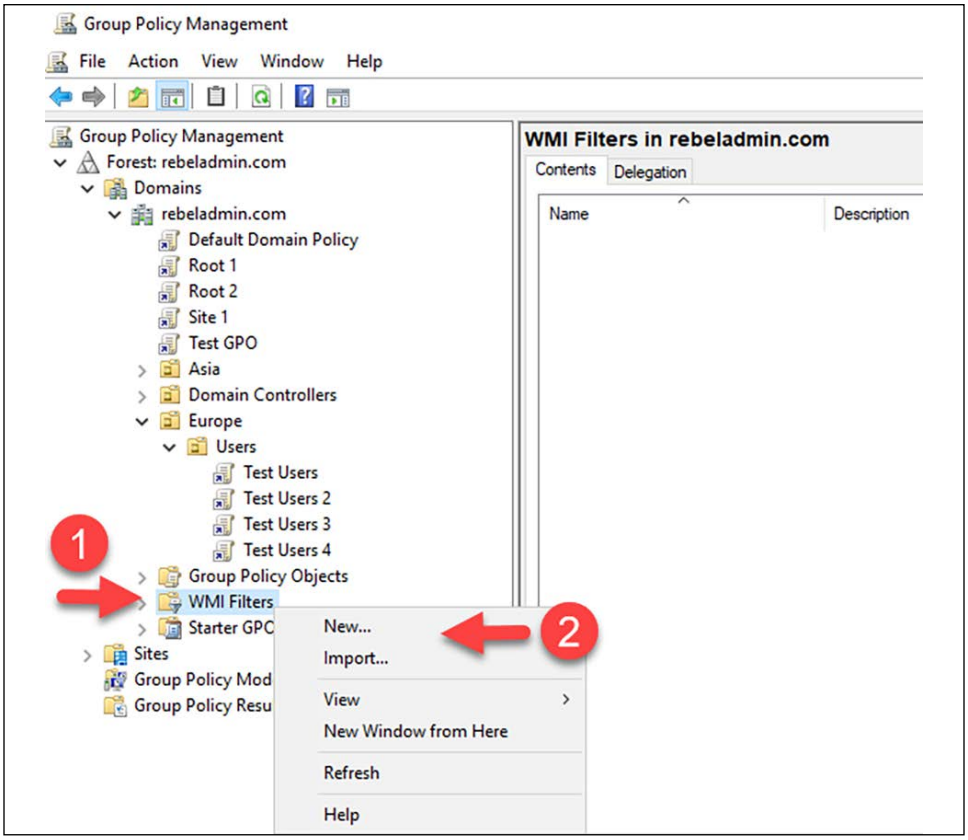


Figure 10.19: Creating WMI filters

This will open up a new window. There, we can define the WMI query:

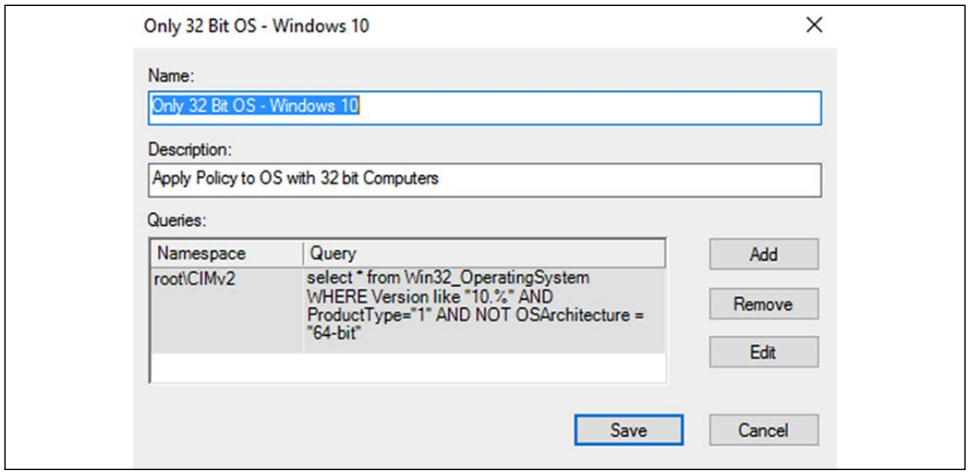


Figure 10.20: WMI filter

By clicking on the **Add** button, we can define **Namespace** and **WMI Query**. As an example, I have created a WMI query to filter the Windows 10 OS that is running the 32-bit version:

```
select * from Win32_OperatingSystem WHERE Version like "10.%"
AND ProductType="1" AND NOT OSArchitecture = "64-bit"
```

In the following commands, you can find a few examples of commonly used WMI queries:

- To filter a Windows 8 64-bit OS, use the following command:

```
select * from Win32_OperatingSystem WHERE Version like
"6.2%" AND ProductType="1" AND OSArchitecture = "64-bit"
```

- To filter a Windows 8 32-bit OS, use the following command:

```
select * from Win32_OperatingSystem WHERE Version like
"6.2%" AND ProductType="1" AND NOT OSArchitecture = "64-
bit"
```

- To filter any Windows Server 64-bit OS, use the following command:

```
select * from Win32_OperatingSystem where (ProductType =
"2") OR (ProductType = "3") AND  OSArchitecture = "64-bit"
```

- To apply a policy to a selected day of the week, use the following command:

```
select DayOfWeek from Win32_LocalTime where DayOfWeek = 1
```

  In the preceding command, day 1 is Monday.

We can use Microsoft **WMI Code Creator** to create WMI code: `https://bit.ly/3HR1sz1`.

Once a WMI filter has been created, it needs to be attached to the GPO. To do that:

1. Go to GPMC and select the required GPO.
2. Make sure to select the **Scope** tab
3. Then, in the **WMI Filtering** section, select the required WMI filter from the drop-down box:

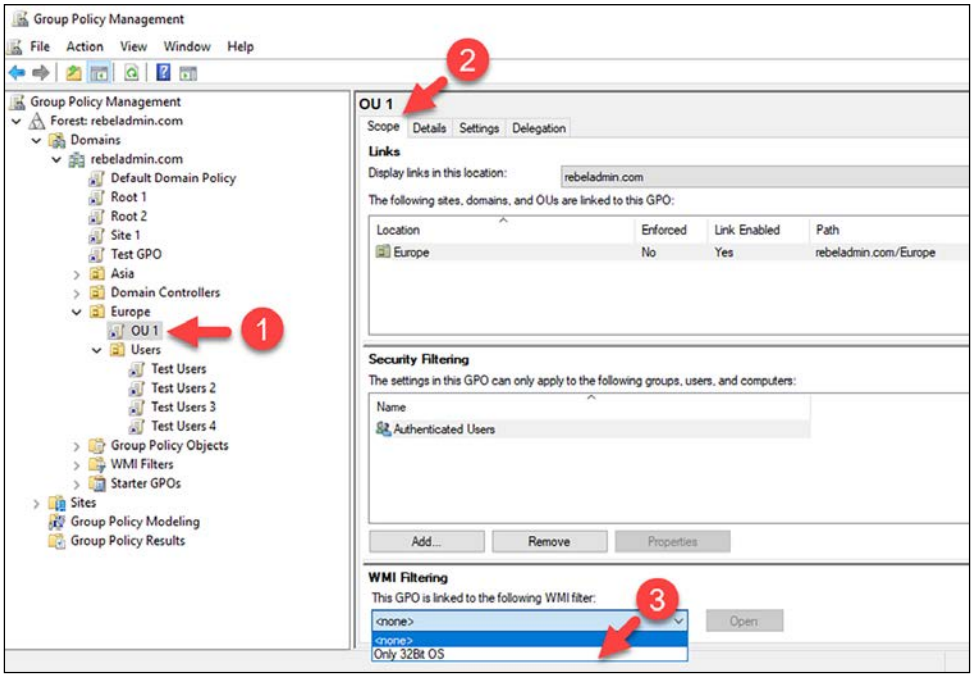


Figure 10.21: Selecting the WMI Filter

Now it is time for testing. Our test query is to target the Windows 10 32-bit OS. If we try to run the query over a 64-bit OS, it should not apply. We can check and confirm the result by running `gpresult /r`:

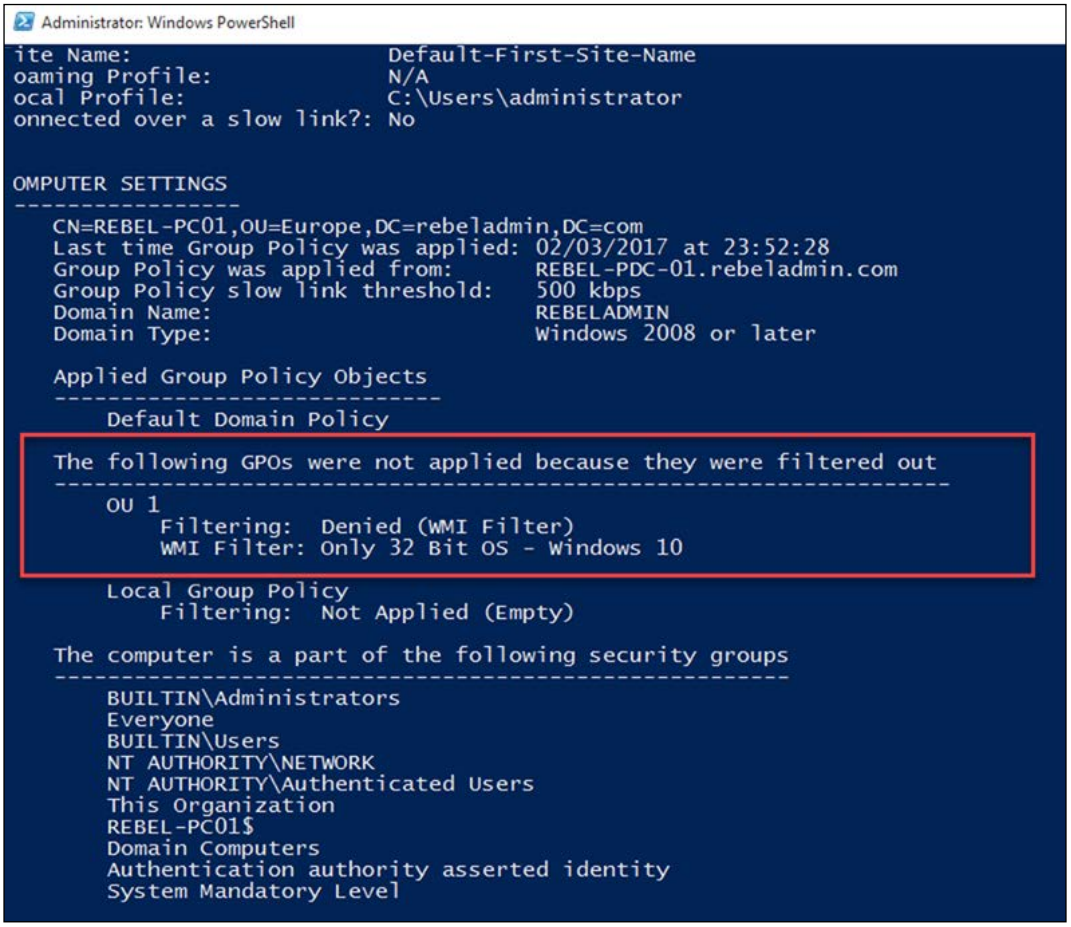


Figure 10.22: gpresult /r output

The test was successful and the policy was blocked as we were running a Windows 10 64-bit OS.

Now we know how we can apply these different filtering options to target specific objects for a GPO. But in what order will all these apply? The following list explains the order of processing:

1. **LSDOU**: The first filtering option will be based on the order in which policies are mapped in the domain structure. This has been covered in detail in an earlier section of this chapter.

2. **WMI filters**: The next on the list is WMI filtering, which we looked at in this section. If the query/condition is true, the Group Policy will process.
3. **Security settings**: The last on the list is security filtering. If the user or device is in filtering, the group policy will be processed.

Group Policy preferences

Group Policy preferences were introduced with Windows 2008 to publish administrative preference settings to Windows desktop OS and server OS. These preference settings can apply only to domain-join computers. Group Policy preferences provide granular-level targeting and provide easy management via enhanced GUI. Group Policy preferences have replaced many Group Policy settings that required registry edits or complex logon scripts. Group Policy preferences are capable of adding, updating, and removing settings such as the following:

- Drive maps
- Internet Explorer settings
- Registry entries
- Printer deployment
- Start menu items
- Power management
- Local users and groups
- File replication
- Managing VPN connections
- Schedule tasks

Group Policy settings and Group Policy preferences are processed in two different ways. Group Policy settings are applied during the boot-up process and the user logon process. After that, settings are refreshed every 90 minutes (default). Once the Group Policy setting is applied, its values cannot change easily. It is required to push new values via Group Policy. Otherwise, even if it's changed, it will be overwritten in the next policy refresh cycle. But Group Policy preferences will not be enforced. They allow users to alter them if required. This also allows you to configure applications that are not Group Policy-aware.

Group Policy preferences are also divided into computer configuration and user configuration. We can use GPMC to manage preference settings.

To access preference settings, select the Group Policy, right-click on **Edit...**, and expand the computer configuration or user configuration. There, we can see the `Preferences` container:

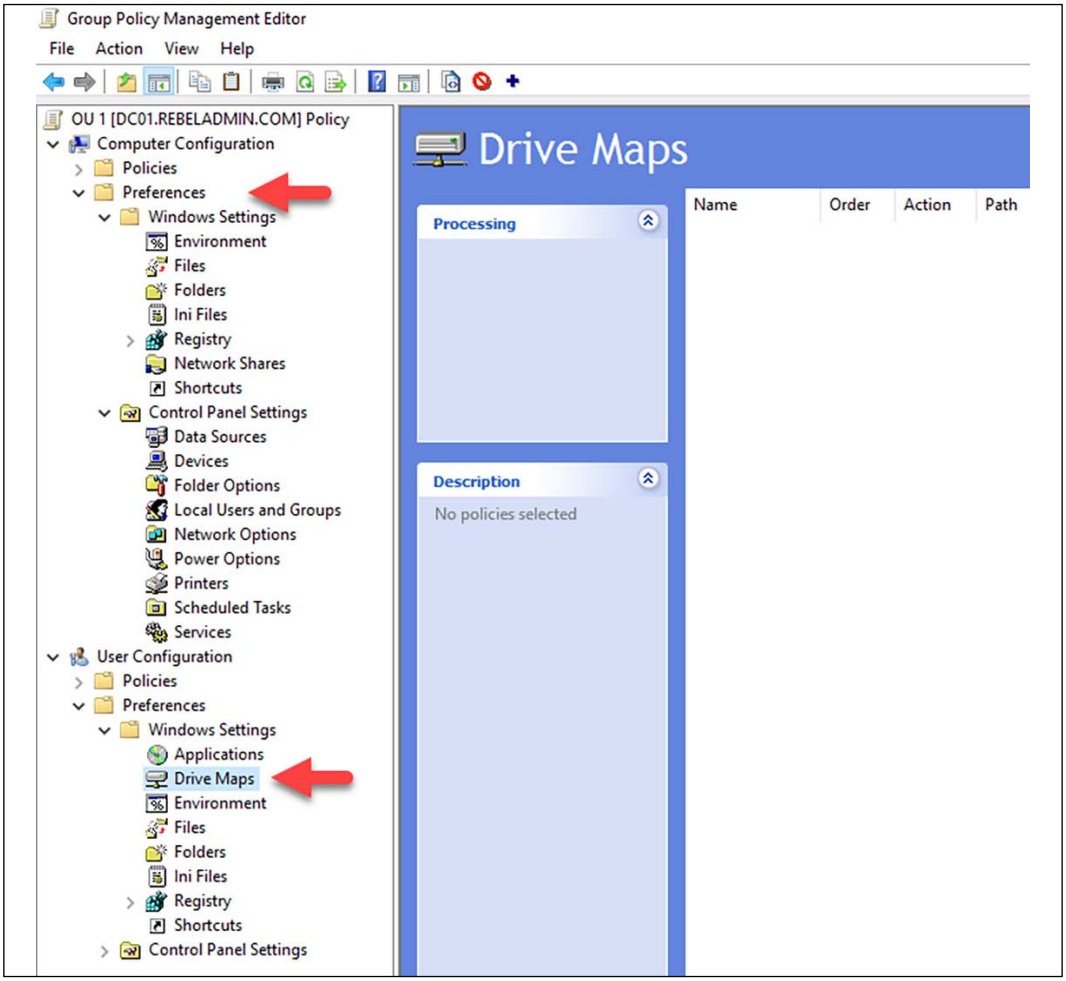


Figure 10.23: Drive Maps

As an example, we can see how we can configure Internet Explorer settings. This is one of the most commonly used preference settings in organizations, specifically to publish proxy settings. Before Internet Explorer 10, Internet Explorer settings were managed by using **Internet Explorer Maintenance** (**IEM**) in their Group Policy. If your organization has IE settings published using IEM, this will no longer apply to IE 10 and IE 11. Internet Explorer settings can also be applied via registry edits. Group Policy preferences made this easy as it can use a GUI similar to an actual IE settings window.

To configure the settings, open the Group Policy settings and then go to **User Configuration | Preferences | Control Panel Settings | Internet Settings**. Then, right-click and select **New**. There, we can select the settings based on the IE version. There is no option for IE 11, but any setting that applies to IE 10 will apply to IE 11 too:

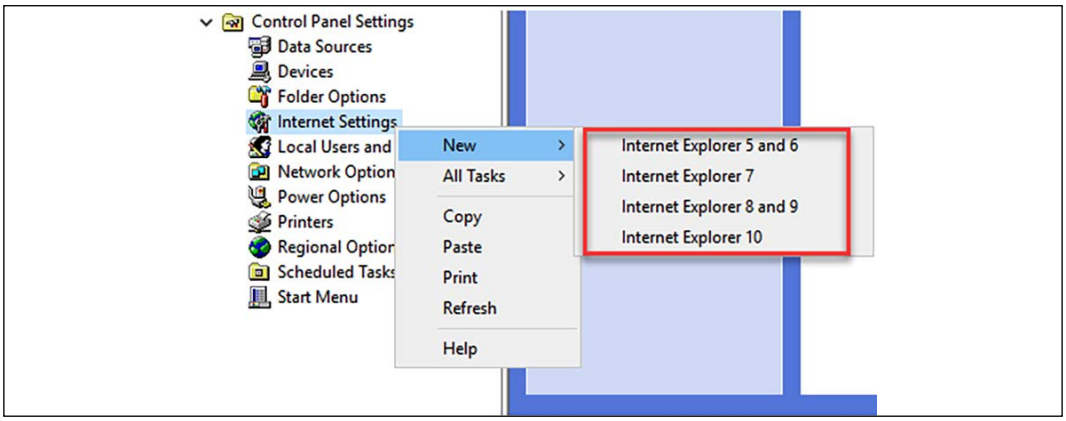


Figure 10.24: Internet Explorer settings

Once you open the relevant configuration, you will see that it is very similar to the settings window we see in the application itself. Complex registry entries to configure IE settings are no longer required.

One thing you need to make sure of is that once you input the changes, press the *F6* key to apply the changes. If they work fine, the *red* dotted line will change to a *green* dotted line. It doesn't matter what changes you make; if you do not activate them by pressing the *F6* key, they will not publish:

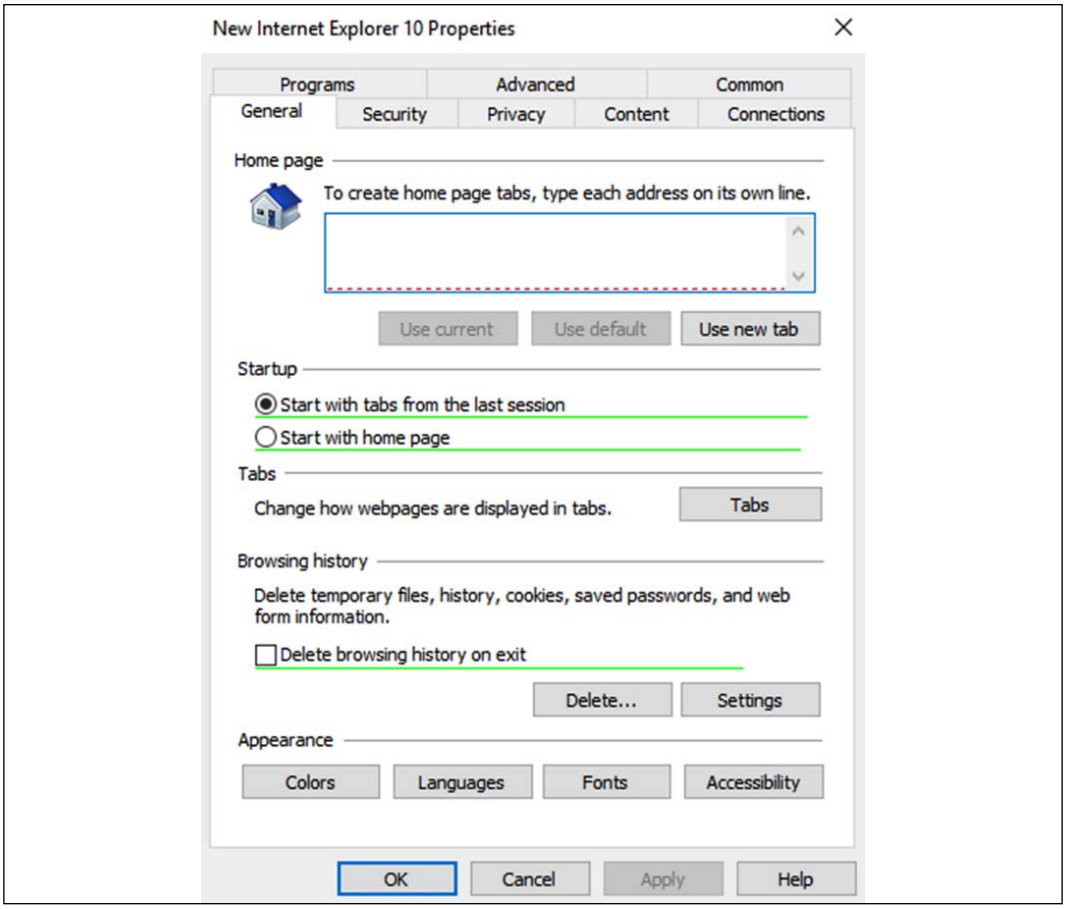


Figure 10.25: Applying Internet Explorer settings

These preference settings will not prevent users from changing them. If it is necessary to prevent users from changing preference settings, then we need to use Group Policy settings for it.

Item-level targeting

In the previous section, we looked at how we can use WMI filters for granular-level Group Policy targets. Similarly, item-level targeting can be used to target Group Policy preference settings based on application settings and properties of users and computers.

We can use multiple targeting items in preference settings and make selections based on logical operators (such as *AND*, *OR*, *IS*, and *IS NOT*).

Item-level targeting in Group Policy preferences can be set up/managed using GPMC. To do that, open the Group Policy settings, go to the relevant preference settings, and then right-click and select **Properties**.

As per the previous example (IE 10 settings), the path should be **User Configuration | Preferences | Internet Settings | Internet Explorer 10**. Then, right-click and select **Properties**.

From the **Properties** window, perform the following steps:

1. Select the **Common** tab.
2. Tick **Item-level targeting**.
3. Then, click on the **Targeting...** button:

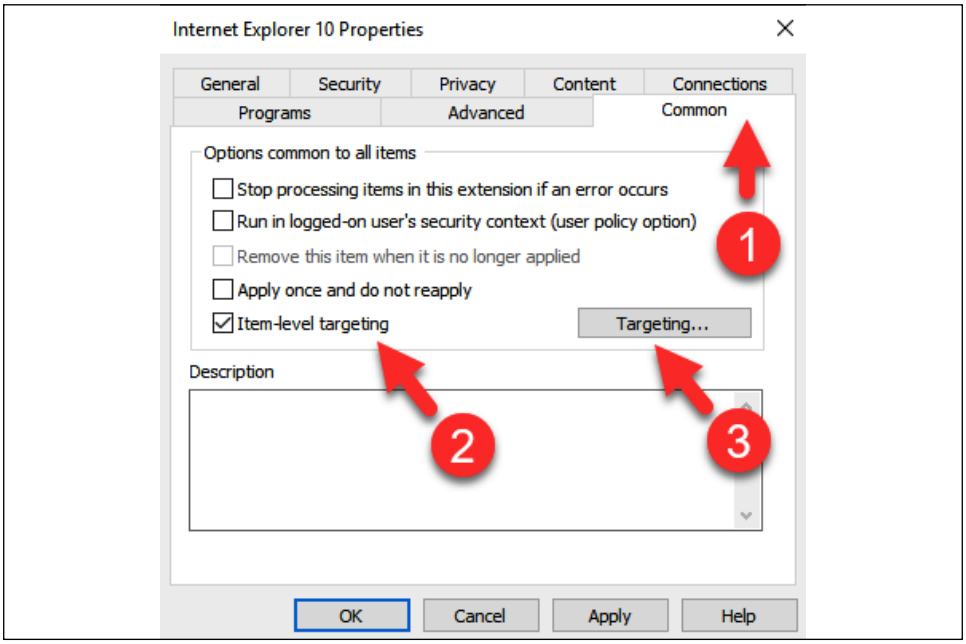


Figure 10.26: Item-level targeting

In the next window, we can build granular-level targeting based on one item or multiple items with logical operators:

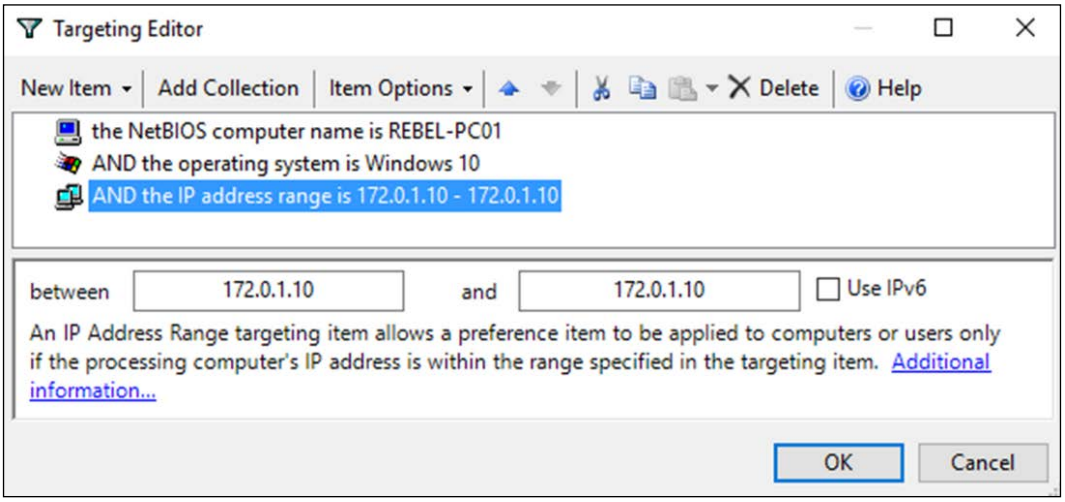


Figure 10.27: Targeting Editor

In the preceding example, I built a query based on three settings, which are the NetBIOS name, the OS, and the IP address. To apply the preference settings, all three statements should provide the `true` value as a result as I used the *AND* logical operator. If it's the *OR* logical operator, the result can have either `true` or `false` values.

In the preceding screenshot, the **New Item** menu contains items we can use for targeting. **Add Collection** allows you to create a parenthetical grouping. The **Item Options** menu is responsible for defining logical operators.

Loopback processing

Group Policy has two main configurations. One is targeted computer settings, and the other is targeted user configuration settings. When we apply user configuration to a user located in the OU, it doesn't matter which computer they log in to; their policy settings will follow them. As an example, let's assume that the user Peter is located under the `Sales` OU. The computer he usually logs in to is also located under the same OU. However, he occasionally logs in to the meeting room laptop that is located under the `IT operations` OU. The `IT operations` OU has its own computer configuration and user configuration policies assigned. But when Peter logs in to it, he still has the same settings he had in the `Sales` OU PC. This is the normal behavior of group policies. However, there are situations where it needs to apply user policy settings based on the computer the user logs in to.

Remote Desktop Services (**RDS**) and Citrix Xenapp/XenDesktop solutions are one of the greatest examples of this scenario. These solutions are mostly open for login from remote networks. Therefore, their security and operation requirements are different from a computer in LAN. If users who log in from different OUs are going to have different settings, it's hard to maintain the system with the required level of protection. Using *loopback processing*, we can force users to only have user policy settings that are linked to the OU where computers are located.

There are two modes of loopback processing:

- **Replace mode**: In replace mode, user settings attached to the user from the original OU will be replaced by the user settings attached to the destination OU. If loopback processing replaces the mode enabled, when Peter logs in to the meeting room laptop, he will get the same settings as the user in the `IT operations` OU.
- **Merge mode**: If merge mode is enabled, in my example, Peter's sales user settings will apply when he logs in to the meeting room laptop first. Once the Group Policy settings are processed, it will also add the user settings from the `IT operations` OU. If there are any conflicting settings, the `IT operations` OU user policy settings will prevail.

To enable loopback processing for Group Policy, go to the Group Policy edit mode and browse to **Computer Configuration | Policies | Administrative Templates | System | Group Policy | Configure user Group Policy loopback processing mode**:

1. Click on the **Enabled** option.
2. Then, select **Merge** or **Replace** mode:

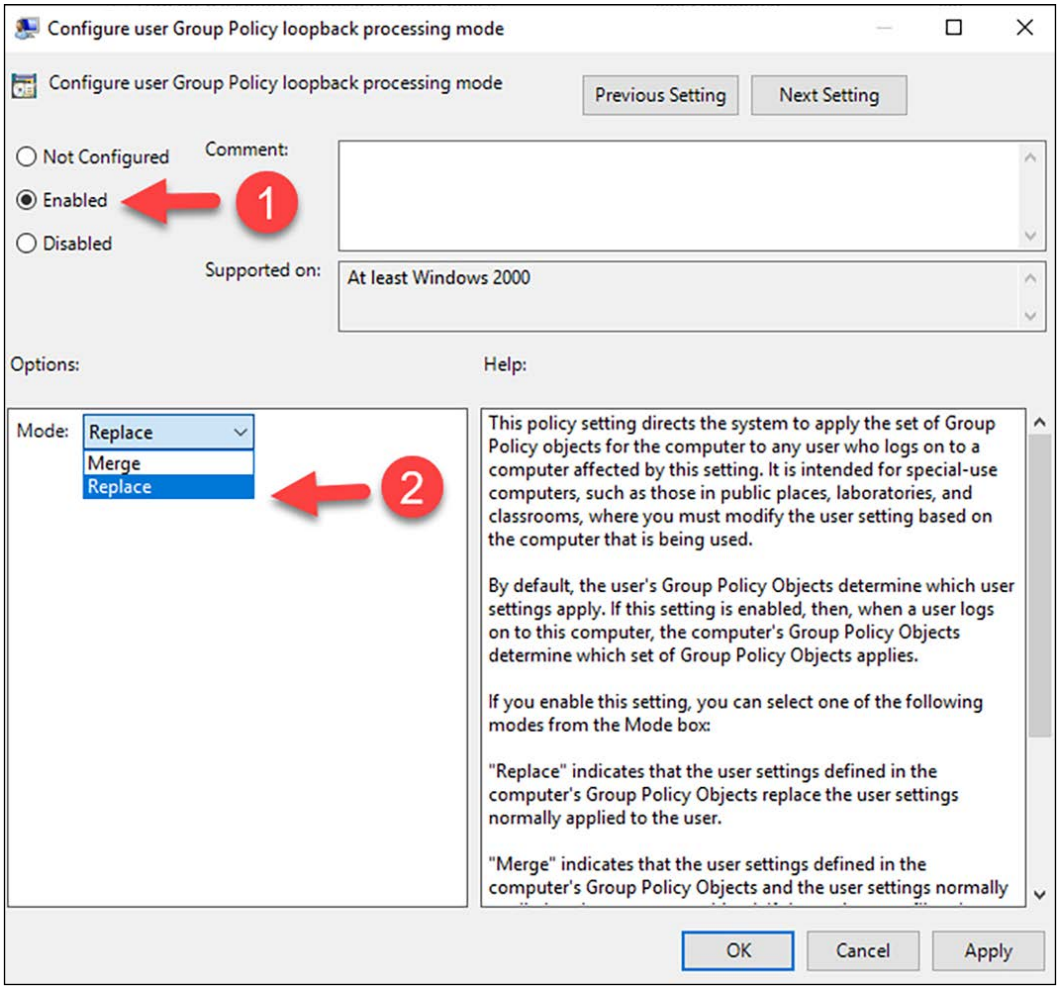


Figure 10.28: Loopback policy settings

Group Policy best practices

In Sri Lanka, there is a common saying for explaining risky action: *eating curd from a knife*. Curd with honey is amazing, but if you have to eat it using a sharpened knife, there is a risk that you may cut your tongue. But it's still worth taking the risk (if you have ever tasted curd and honey before). Group policies are also like that; they can do so many useful things, but only if you use them correctly. In the Active Directory environment, Group Policy-related issues are the most painful and time-consuming troubleshooting tasks as there are so many things that can go wrong.

Here, I have listed a few tips that will be useful for designing group policies:

- **Identify the best place to link the Group Policy**: The Group Policy can be linked to the site, domain, or OU. Organizations have different Group Policy requirements that can also map with the aforementioned components. It is important to understand the best place in the hierarchy to publish each Group Policy setting. This will prevent repetition and Group Policy conflicts. As an example, password complexity settings are common for all of the objects under the domain, so the best place to link the policy for the password settings is the domain root, not the OU.
- **Standardize settings**: Today, infrastructure requirements are complex. Each business unit has its own operation and security requirements to address via group policies. When designing group policies, always try to summarize the changes as much as possible. If two business units have almost the same user settings requirements, discuss it with the relevant authorized people (such as the line manager and team leads) and try to use the standard settings. This will allow you to use one Group Policy and link it to two business units instead of creating two separate group policies. Always try to reduce the number of group policies that will be used in the system, since when the number of group policies increases, it also increases the time taken for the login process.
- **Use meaningful names**: This is a small suggestion, but I have seen people use Group Policy with names that don't explain anything. In such a scenario, when you are troubleshooting, it can take an awfully long time to find the relevant Group Policy. Make sure that you name the Group Policy to reflect the settings in there. It doesn't need to have details, but at least have something that you and your colleagues can understand.
- **Avoid mixing user settings and computer settings**: Group Policy has two main configuration sets that will apply to users and computers. Always try to avoid mixing these settings in the same Group Policy. Try to use separate group policies for computer settings and user settings. This will make it easy to organize policies.

After that, disable the unused configuration section in the Group Policy to avoid processing. This can be done by using GPMC. To do that, perform the following steps:

1. Click on the relevant group policy.
2. Go to the **Details** tab.
3. Select the relevant mode under **GPO Status**:

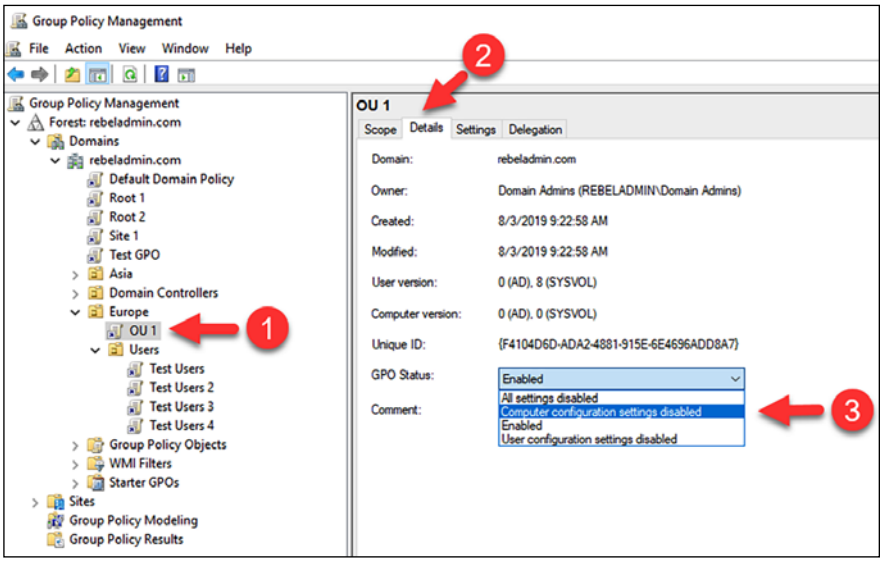


Figure 10.29: Changing the GPO status

- **Use Group Policy enforcing and block inheritance appropriately**: These features are a good way to manage the default Group Policy inheritance, but use them carefully as they can prevent users/systems from applying critical policy settings from the top of the hierarchy. In the same way, by enforcing policies, you may enforce settings on users and computers that should not be targets. Therefore, always measure the impact before using enforce or blocking inheritance features.
- **Never edit default policies**: There are two default group policies: Default Domain Policy and Default Domain Controllers Policy. It is highly recommended that you do not use these to publish the Group Policy settings. These can be used as the baseline when you design the group policies, and they allow you to revert to the original settings with minimum effect. These can also be used as a reference to troubleshoot inheritance issues and replication issues.

- **Be careful when using loopback processing**: Issues related to loopback processing are due to a lack of knowledge, especially when they involve replace and merge modes. In this chapter, we covered both of these modes. If you are enabling loopback processing, always use a separate Group Policy and name it correctly so that everyone understands it when it comes to troubleshooting. My recommendation is that you use it in a situation where you can use replace mode. Merge mode can create lots of hassles with combined group policies plus conflicted settings. Loopback settings are also recommended to use only with **Virtual Desktop Infrastructure** (**VDI**) solutions such as Citrix and RDS, as they help to maintain consistency.
- **Win or lose**: Earlier in this chapter, we looked at which policy settings will win when there is a settings conflict. But in theory, there should not be any conflicts at all if the design is done correctly. Therefore, in design, always avoid repeating values in a hierarchical structure. If different values are used for the same setting, use filtering options and block inheritance to prevent the policy conflicts.
- **Housekeeping:** Last but not least, it is important that you review the group policies periodically and remove the policies that have not been used. Also, remove the legacy policy settings if they have been replaced by new policies or methods. Audit and make sure that the hierarchical order is maintained and objects are getting the expected group policies applied to them.

Useful group policies

So far in this chapter, we have talked about the benefits of group policies and how to use group policies appropriately. As the final section of this chapter, I choose to talk about some useful group polices that can be used in environments. It doesn't mean that every environment should have similar policies, but we can use this as an example and grow from there:

1. **Policy Name**: Password Policy.

   Policy Location: **Computer Configuration | Policies | Windows Settings | Security Settings | Account Policies | Password Policy**.

   Description: This is one of the most commonly used group policies in an Active Directory environment. Passwords are no longer the best method for securing an account, but passwords are still widely used as the primary authentication method. When a system asks to set a password, as humans, we tend to use the easiest password we can remember. These passwords can also be easy for intruders to break. Using password policy, we can enforce complexity and other different standards on the user passwords. Under this policy, we have seven settings.

Enforce Password History: This setting defines the number of unique passwords that need to be used before an old password can be reused. The value for this setting can be between 0 and 24.

Maximum Password Age: This setting will define the validity period of a password before the system forces the user to change it. We can define any value between 0 and 999 here, but the default value is 42 days. It is recommended to use a value between 30 and 90.

Minimum Password Age: This policy setting defines the number of days that must go by before a user can change their computer password. The default value for this is 1 day. If you are enabling the enforce password history setting, this value must be more than 0. Also, the value should be below the maximum password age value.

Minimum Password Length: This setting defines the minimum number of characters a password should have. This can be any number between 0 and 14. It is recommended to use at least 8 characters for improved security.

Minimum Password Length Audit: This setting can have a value between 1 and 128. When the value of this setting is greater than the Minimum Password Length value and the new user account password length is less than the value of this setting, an audit event will be issued. This is normally used when engineers want to evaluate the impact of using minimum password length settings.

Password must meet complexity requirements: This policy setting determines the complexity of the new password. If this is enabled, it should meet the following minimum requirements:

- The password cannot contain the username or part of the user's full name that exceeds two consecutive characters.
- The password must be at least six characters (this will change if the minimum password length policy setting is specified).
- The password must have at least three of the following:
- English uppercase characters
- English lowercase characters
- Base 10 digits (0-9)
- Non-alphanumeric characters

Store passwords using reversible encryption: This policy determines whether passwords need to be stored with reversible encryption. Some applications need to know the user's password for authentication. This policy should not be enabled unless it is an application requirement to encrypt the password.

In the following table, I have listed a sample policy that we can use:

Setting	Value
Enforce Password History	24
Maximum Password Age	30
Minimum Password Age	1
Minimum Password Length	8
Password must meet complexity requirements	Enabled

2. **Policy Name**: Removable Storage Access.

   Policy Location: **Computer Configuration | Policies | Administrative Templates | System | Removeable Storage Access**.

   Description: USB devices can be a source of malware and viruses. On corporate networks, they can also be used to steal sensitive data. Therefore, it is common security practice to disable USB access on corporate computers. There are a few different methods we can use to do this. We can use the registry entry to block it or we can use third-party software to block USB access. We also can use group policy to do this.

   In this policy, we have the following settings that can be used to block USB access:

   All Removable Storage classes: **Deny all access**: When you enable the policy setting, the system will block USB access to the device. If we disable the policy or set it to the **Not configured** value, USB access will be allowed and the system will have read/write permissions to the USB device.

3. **Policy Name**: Prohibit access to Control Panel and PC settings.

   Policy Location: **User Configuration | Policies | Administrative Templates | Control Panel**.

   Description: We use a control panel on the PC to manage the hardware and OS settings. In a corporate environment, if users change the settings of the PC as they wish, it is hard to maintain standards, especially if the device is used by multiple users. As an example, if users are connecting to a WVD (Windows Virtual Desktop) or Citrix solution, we need to maintain configuration standards for every user. To do so, it's best to disable access to the control panel settings.

   We can control access to control panel and PC settings by using the following policy setting:

   Prohibit access to Control Panel and PC settings: To block access to control panel and PC settings, we need to enable this policy setting.

Once this policy setting is enabled, control panel settings will be removed from the **Start screen** and **File Explorer**. Also, PC settings will be removed from **Start screen, Settings charm, Account picture**, and **Search result**.

4. **Policy Name**: Folder Redirection.

   Policy Location: **User Configuration | Policies | Windows Settings | Folder Redirection**.

   Description: In a PC, user settings and user files are, by default, saved inside the **Users** folder on the C:\ drive. This works well with standalone computers, but if users are logging in to different systems, it will be a nightmare to maintain settings and access to user files. Also, if a solution such as WVD or Citrix is in place, large numbers of users will appear on each session host server. If every user is saving files in their profile, it will be difficult to manage the storage requirements of session host servers. To address these sorts of issues, we have two options. We can use roaming profiles for users and redirect user profile data to a network location or else we can use **Folder Redirection** to redirect the path of known folder to a network location or specific folder on a local computer. We can also use both of these solutions together. By using both settings together, we can reduce the size of the roaming profile. To enable folder redirection, we have to use policy settings located under **User Configuration | Policies | Windows Settings | Folder Redirection**.

   We can enable folder redirection for the following list of folders.

   - AppData/Roaming
   - Contacts
   - Desktop
   - Documents
   - Downloads
   - Favorites
   - Links
   - Music
   - Pictures
   - Saved Games
   - Searches
   - Start Menu
   - Videos

Each folder in the preceding list has the following settings when it comes to the target configuration.

Basic—Redirect everyone's folder to the same location: By using this policy setting, we can force all redirected folders to save in the same location. These folders can be saved in a network location or a specific path in the hard drive.

Advanced—Specify locations for various user groups: This policy setting helps to redirect folders to different locations based on security group memberships. As an example, we can say that Sales Group users should use the `\\fileserver\salesfolders` network share as the target, and Tech Group should use `\\fileserver\techfolders` as the target.

Create a folder for each user under the root path: When choosing this setting for the target folder location, each user will have their own folder under the root folder.

Redirect to the following location: By using this policy setting, we can force the system to use an explicit path for folder redirection. This way, users will share the same path for the redirection.

Redirect to the local user profile location: This policy setting can be used to force the system to save data under the C:\Users folder.

It is best to redirect folders to the network share and use **Create a folder for each user under the root path** to create a unique folder for each user. This way, it will be easy to manage as well as easy to troubleshoot if something goes wrong.

5. **Policy Name**: Turn off Windows Installer.

   Policy Location: **Computer Configuration | Administrative Templates | Windows Components | Windows Installer**.

   Description: On a PC, Windows Installer is responsible for installing, updating, and uninstalling applications. Users do not always need admin rights to install software; some applications can be installed in the current user context. In a corporate environment, if users have the ability to install applications on a PC as they wish, it will be difficult for engineers to maintain. Also, this can create security risks. Therefore, it is recommended to turn off the windows installer and this will prevent users from installing `.msi` packages. However, this will not prevent the installation of applications using other methods.

   To turn off the windows installer, we can use the following setting:

   Turn off Windows Installer: To enable this policy setting, click on **Enable** in the settings window and then select **Always** under **Disable windows installer option**.

6. **Policy Name**: Do not store the LAN Manager hash value at the next password change.

   Policy Location: **Computer Configuration | Windows Settings | Security Settings | Local Policies | Security Options**.

   Description: Windows doesn't keep user passwords in cleartext format. It uses hash values instead. When the user changes their password, and provided it is less than 15 characters, the system generates an LM hash as well as an NT hash. Then, this hash value will be saved in the local SAM database or Active Directory.

   Note: If the password is longer than 15 characters, the LM hash will be set to a fixed value and it is equivalent to a null password. Therefore, any attempts to break the hash will be in vain.

   An LM hash is weak compared to an NT hash. As a security best practice, it is recommended to not save the LM hash in the local SAM database. We can do this using GPO. For that, we need to use the following policy setting:

   Network security: Do not store the LAN Manager hash value at the next password change: To enable this policy setting, click on **Enable** in the policy settings window.

   More information about hash values and Active Directory authentication is available in *Chapter 15, Active Directory Rights Management Services*.

7. **Policy Name**: Rename administrator account.

   Policy Location: **Computer Configuration | Windows Settings | Security Settings | Local Policies | Security Options**.

   Description: In the Active Directory environment, we should not only consider protecting Active Directory accounts, but we also need to consider the protection of local administrator accounts. A compromised local administrator account will allow attackers to later move within the network and gain access to more privileged Active Directory accounts. In computers, we have a default administrator account called Administrator. If we can change the name of this account, it will be slightly difficult for attackers to guess the user name of the local account. We can change the default name of the account by using Group Policy.

   Accounts: **Rename administrator account**: By using this policy setting, we can rename the administrator account. To enable this policy setting, click on **Define this policy setting** and then specify the new name for the administrator account. As a best practice, do not use a name that can be easily guessed, such as *Admin* or *LocalAdmin*.

In this section, we looked at some useful group policies. However, there are a lot more policies out there that can be used based on your requirements. When you test a new policy, make sure to use at least a separate OU with a few objects before introducing it to production. This will reduce the risk.

Summary

Group policies are one of the core values of the AD environment. As long as they are designed and maintained properly, they can be used to manage computer and user settings effectively. In this chapter, we learned about Group Policy components and their capabilities. This was followed by explaining features that can be used to design and maintain a healthy Group Policy structure. We also learned about Group Policy designing best practices. In the last section, we went through some useful group policies.

In the next chapter, we are moving on to the third part of this book, which will focus on Active Directory server roles. In this part, you will learn about the advanced features of AD DS, including schemas, replication, the **read-only domain controller** (**RODC**), and Active Directory recovery.

11

Active Directory Services – Part 01

With this chapter, we are moving into the third section of this book, which focuses on the **Active Directory** (**AD**) server roles. There are five main AD server roles:

- **Active Directory Domain Services** (**AD DS**)
- **Active Directory Lightweight Directory Services** (**AD LDS**)
- **Active Directory Federation Services** (**AD FS**)
- **Active Directory Rights Management Services** (**AD RMS**)
- **Active Directory Certificate Services** (**AD CS**)

We have already looked into many AD components, features, and capabilities, but we are not quite done yet. AD services are attached to many different components, such as **Domain Name System** (**DNS**), **Distributed File System Replication** (**DFSR**), and group policies. To maintain a healthy AD environment, we need to manage each of these components properly and make sure they do what they are supposed to do. However, in some scenarios, IT professionals and software developers are only interested in AD authentication and authorization capabilities. This can be due to application developments, directory migrations (from different vendors), and application authentication requirements. In such situations, we can use AD LDS to provide **Lightweight Directory Access Protocol** (**LDAP**) services. As the name suggests, it's a cut-down version of AD DS and does not depend on many other components. It also has fewer maintenance requirements. In this chapter, we will learn about AD LDS in detail.

There are no healthy AD environments without healthy AD replication. With Windows Server 2008, Microsoft introduced a **Distributed File System** (**DFS**) for `SYSVOL` folder replication. It is better than its predecessor, **File Replication Service** (**FRS**), in many ways. In this chapter, we will learn about the differences between FRS and DFSR. Apart from replication services, AD sites also have a direct impact on AD replication. An AD site is a logical representation of another remote network that has domain controllers and other resources. In this chapter, we will learn about AD sites, site links, subnets, and replication intervals in detail. Toward the end of the chapter, we will learn about how intra-site and inter-site AD replication work. Due to the amount of content, I have divided the topic into two chapters as it will help readers to absorb the information effectively.

In this chapter, we will cover the following topics:

- Overview of AD LDS
- AD replication
- AD sites

We are going to start this chapter by looking into AD LDS. There are a lot of corporate applications that support integration with an existing AD environment. However, there are some applications that only require data stores to save user authentication data and application data. In such scenarios, we can use AD LDS, which is capable of providing data stores and relevant services to access them.

Overview of AD LDS

When we talk about AD, we refer to it as a single service; however, AD DS is a collection of many other components such as DNS, group policies, `SYSVOL` folder replication, and so on. Each of these components needs to operate well in order to run a healthy AD environment. Managing these components isn't easy; it requires investments in resources, time, and skills. It is not just about service uptime and performance; security also plays a crucial role in this. The failure or compromise of these components/services can have a potential impact on the entire AD infrastructure.

Microsoft Windows Core is also count as operating systems. It doesnt have fancy GUIs or lots of applications running, but still do the job of an operating system. It allow users to build systems from scratch according to their requirements. This also increases the server uptime (fewer updates), reliability, performance, and security. Soon after Microsoft released the first AD version, IT engineers, application developers, and IT professionals started requesting a cut-down version of AD DS with pure LDAP capabilities.

They wanted to eliminate all these dependencies and management requirements so they could focus application development upon core AD functions. After Windows Server 2003, Microsoft released **Active Directory Application Mode** (**ADAM**), which allowed administrators to run a cut-down version of AD without things such as group policies and SYSVOL replication. It can run on a desktop computer or a member server similar to any other Windows service. With Windows Server 2008, Microsoft renamed it AD LDS and allowed users to install this role using Server Manager. This version provided administrators with more control and visibility to deploy and manage LDS instances. This was continued with all the AD DS releases after that version, and was included in Windows Server 2022 too.

Where to use LDS

If we are already using AD DS, then the question will be why we need AD LDS. In the following section, we will look at several scenarios where we can use LDS.

Application development

This is the area that has benefited most from AD LDS capabilities. Application development involves lots of research and testing. If these applications are AD-integrated, it is obvious that they need to be developed and tested within an AD environment. During the development process, you may be required to build many test environments. If they're full-blown AD DS instances, it will take resources, time, and effort to deploy and maintain them. AD LDS allows you to run multiple instances of it within the same environment, independently. Each instance will have its own schema, and engineers can maintain the instance for each application test environment. Even if it looks like a cut-down version, it provides the same AD DS authentication and management capabilities, allowing engineers to easily adopt it.

Since AD LDS instances can even be run on a desktop operating system, it only has a few prerequisites. Therefore, applications can also be released with integrated LDS. For example, not every business runs AD. Even though application functions are based on AD features, it is still not easy to convince everyone that they should have an AD environment to run the application. Instead, the application installation can have an integrated LDS instance that can be installed in the guest system as part of the installation process.

Hosted applications

With digital transformation, businesses are using applications more and more for their operations. Most of these applications are web-based applications as they are more convenient to use.

End users do not have to worry about local software installations and all they need to do is access the application via a web browser.

Most of the time these internet-facing applications are deployed in the perimeter or in a public network. These applications can also have authentication requirements, but it is not recommended to install AD DS in the perimeter or on a public network. In such a situation, the most common method is to deploy AD FS to provide federated access; however, if complete isolation is required for the application, we can set up the AD LDS instance inside the perimeter/public network and provide the directory-enabled authentication service to applications. This will ensure the application doesn't have any connection with corporate LAN or the other LDS instances in the perimeter network. This provides a secure environment by design.

Distributed data stores for AD-integrated applications

Most AD-integrated applications also require schema modifications. With schema modifications, the application will store certain datasets in the AD database. When the number of AD-integrated applications increases, AD's schema and data will continue to grow too. This will result in a significant impact on AD replication, especially if replication happens via slow links. Instead of storing data in the AD database, additional datasets of applications can be stored in the LDS instance. The LDS instance will still use AD DS for authentication. Additional datasets stored in the LDS instance will not replicate to any other domain controllers.

Migrating from other directory services

There are environments and applications that use legacy X500-based directory services that like to migrate to AD DS. In such scenarios, AD LDS can be used as a middleman that can also support X500-based applications. It also cleans up the junk and only moves filtered data to AD DS. AD LDS allows the instances to run alongside AD DS, and privileged identity solutions such as MIM can sync data between LDS and AD DS instances if required.

Azure AD Connect is a Microsoft tool that allows us to synchronize user identities from on-prem AD to Azure AD. Even though AD LDS has directory service capabilities, Azure AD Connect does not support synchronization from AD LDS instances to Azure AD.

The LDS installation

In the Windows Server 2022 operating system, LDS can be installed using Server Manager. In order to install LDS, a user needs to log in to the selected systems with *local administrator* privileges.

Once logged in, launch **Server Manager** and click on **Add Roles and Features**. Then, follow the wizard, select **Active Directory Lightweight Directory Services** under **Server Roles**, and proceed:

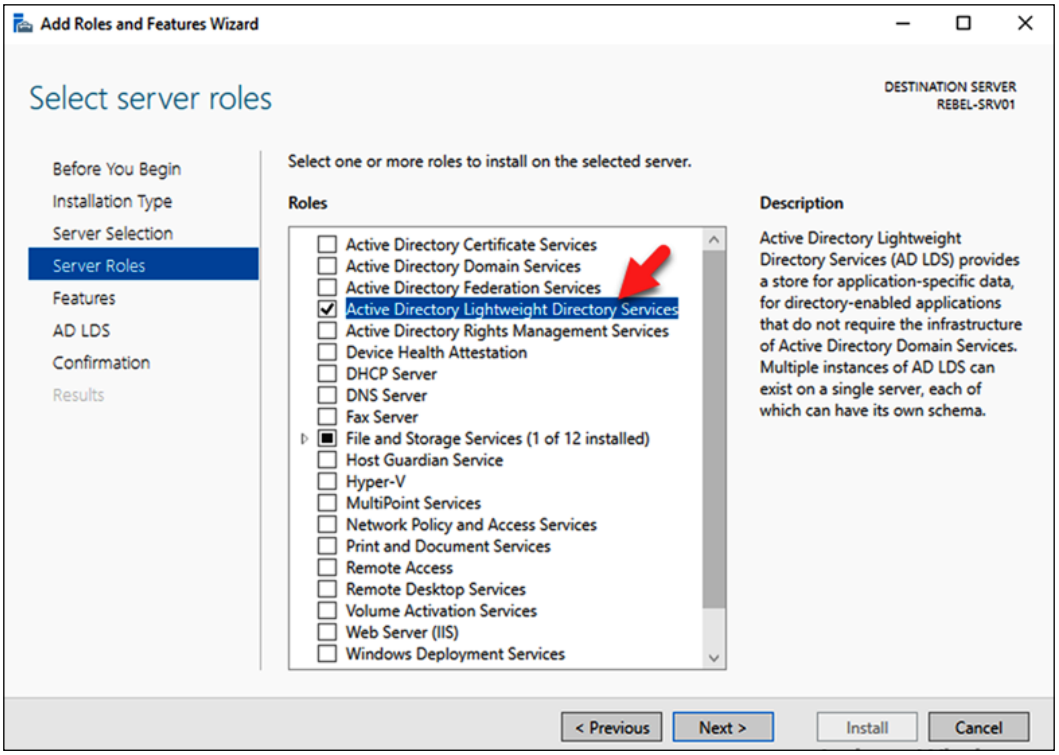


Figure 11.1: AD LDS role

Once the role is installed, click on the **Post-Deployment Configuration** wizard in **Server Manager**. LDS can be set up in two ways: one is by using **a unique instance** and the other one is by using **a replica of an existing instance**. The replica option is similar to using the cloned copy of an existing instance.

This is useful, especially in an application development environment where engineers need to maintain a number of application versions:

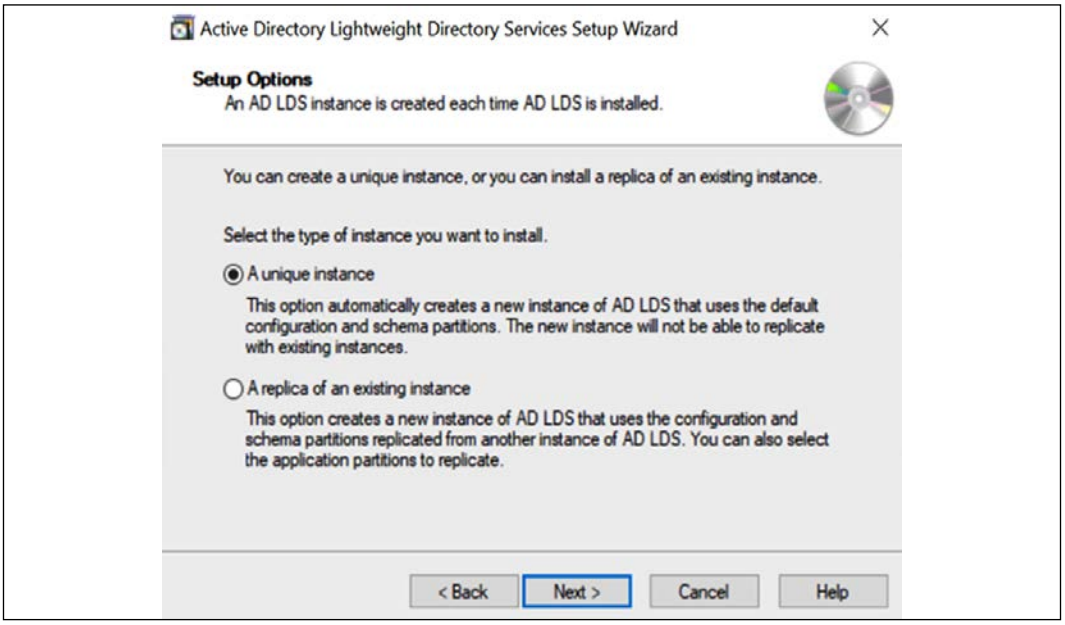


Figure 11.2: AD LDS instance type

In the next window, we can define the name and description of the LDS instance:

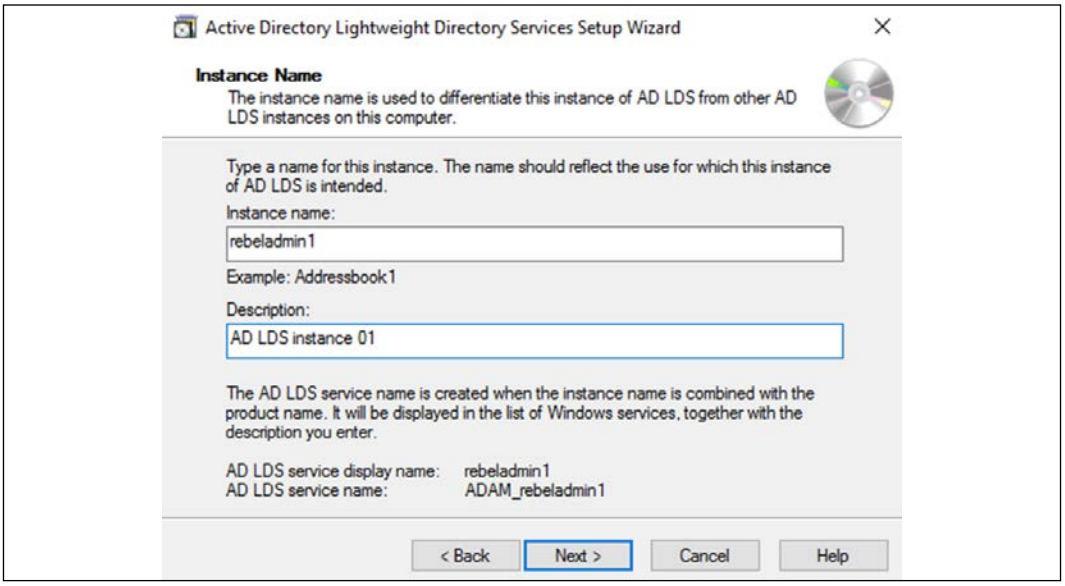


Figure 11.3: AD LDS Instance Name

Then in the next window, we can define the LDS port. By default, the LDAP port is set to `389` and the SSL port is set to `636`. If you run multiple instances, these need to be changed accordingly.

After this, we can create the application directory partition. This allows applications to use a specific partition as a data repository to store the application-related data. If the application is capable of creating a partition, this step is not necessary and we can create a relevant partition during the application deployment process. When defining the application's **partition name**, we need to provide it in a **Distinguished Name** (**DN**) format:

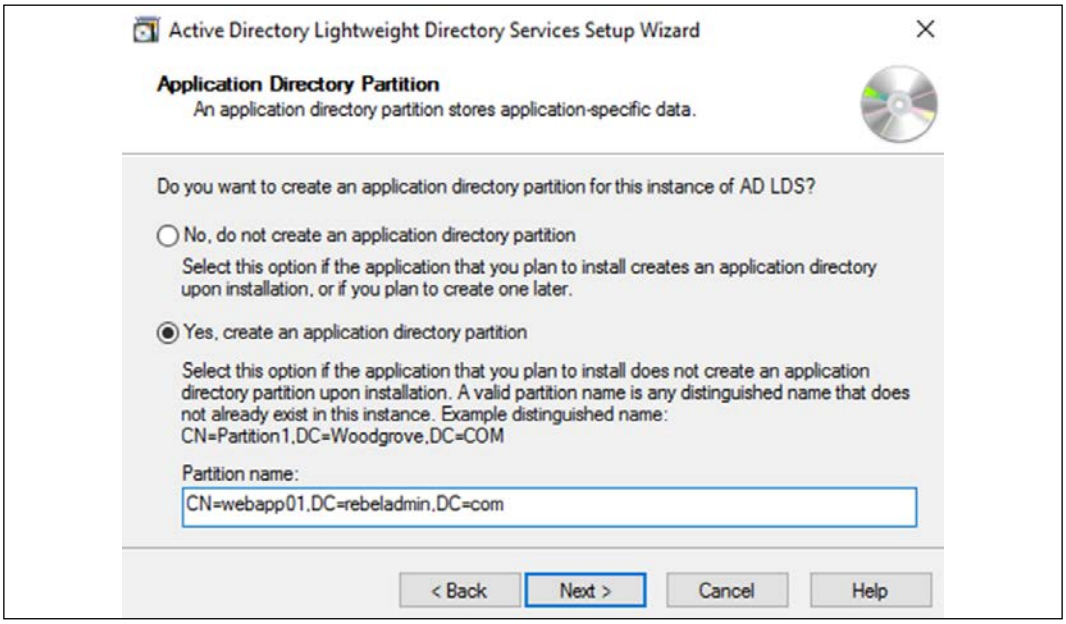


Figure 11.4: AD LDS application directory partition

The next step is to define a location in which to store the LDS data files. After, it gives us the option to specify a service account for LDS. If it's a workgroup environment, you can use the **network service account** or a local user account for it.

If it's a domain environment, it can be any AD user account:

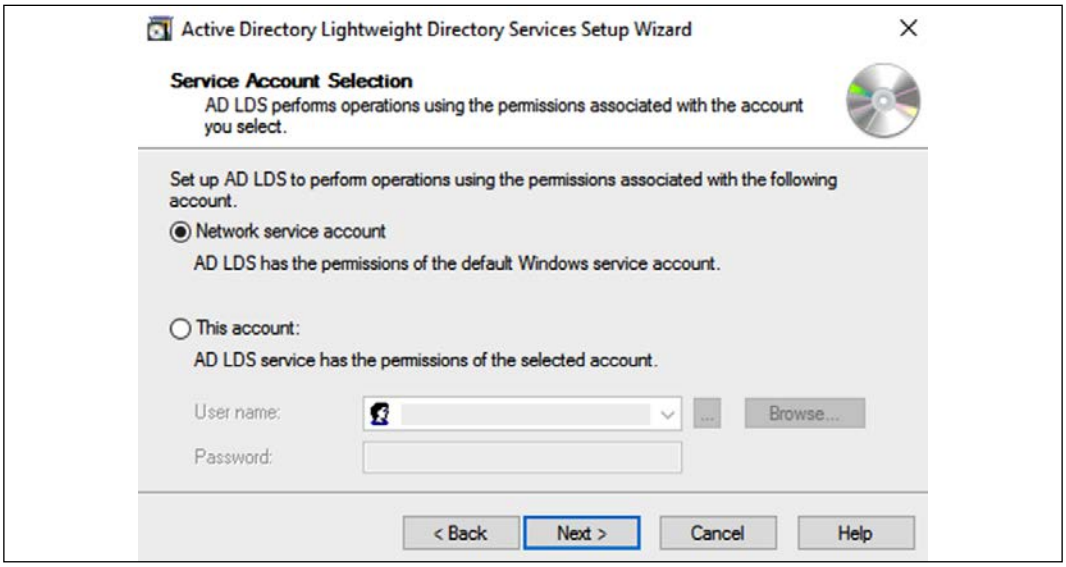


Figure 11.5: AD LDS service account

After that, we need to define the AD LDS administrator account. By default, it selects the user account that was used for the installation. If required, it can be changed to a different account or group.

Once we define the administrator account, the next step is to define which LDIF file to import. This is a text file that represents the data and commands that will be used by the LDAP instance. The text file can contain one or more LDIF files; these files depend on application requirements. For example, for user account functionalities, the relevant LDIF file will be `MSUser`:

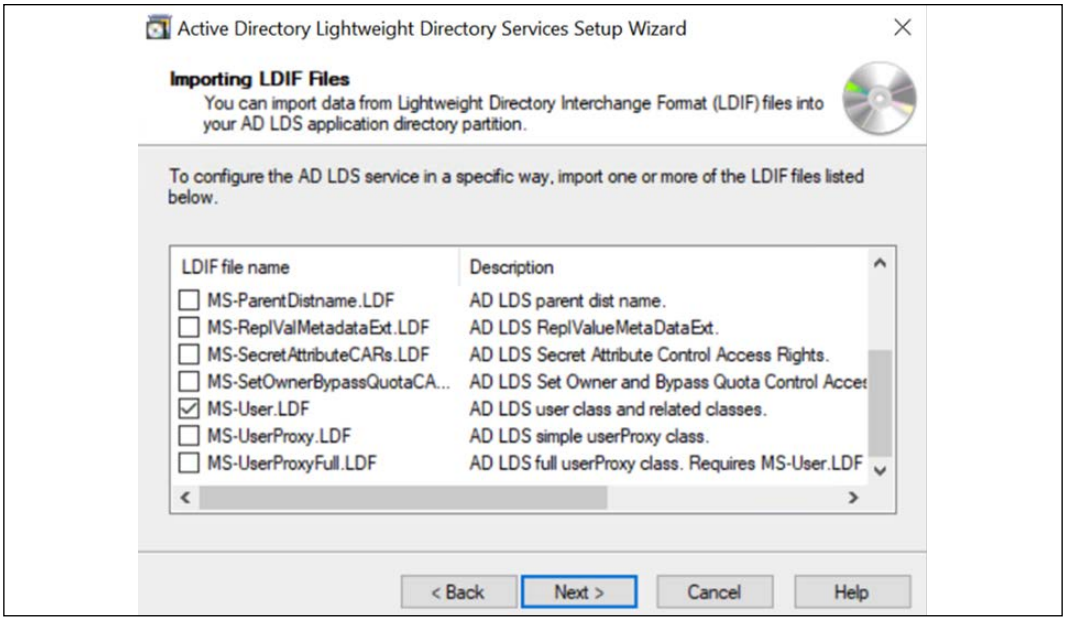


Figure 11.6: AD LDS – Importing LDIF Files

This step completes the AD LDS installation, and once it is completed, we can create the relevant objects and manage them. There are two methods we can use to connect to it. One way is to connect using the **ADSI Edit** tool:

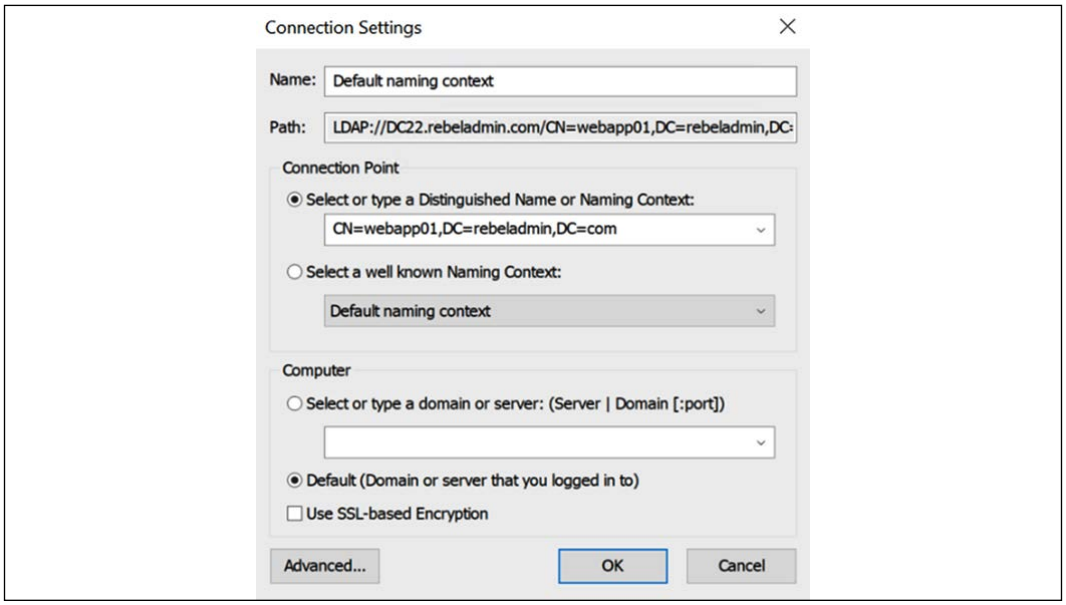


Figure 11.7: AD LDS Connection Settings

The other method is by using PowerShell cmdlets. These are the same commands that we use for AD DS user object management, with the only difference being the need to define the **DN** and server:

```
New-ADUser -name "tidris" -Displayname "Talib Idris" -server
'localhost:389' -path "CN=webapp01,DC=rebeladmin,DC=com"
```

The preceding command creates a user account called `tidris` on the local LDS instance that runs on `389`. Its DNS path is `CN=webapp01,DC=rebeladmin,DC=com`:

```
Get-ADUser -Filter * -SearchBase "CN=webapp01,DC=rebeladmin,DC=com"
-server 'localhost:389'
```

The preceding command lists all the user accounts in the LDS instance, `CN=webapp01,DC=rebeladmin,DC=com`. If you'd like to learn about more commands, please refer to *Chapter 7, Managing Active Directory Objects*.

AD LDS can be installed on a desktop operating system using the **Windows features** option under **Program and Features**. The installation steps are similar to the server version. Once AD LDS is enabled, the setup wizard can be found under **Administrative Tools**:

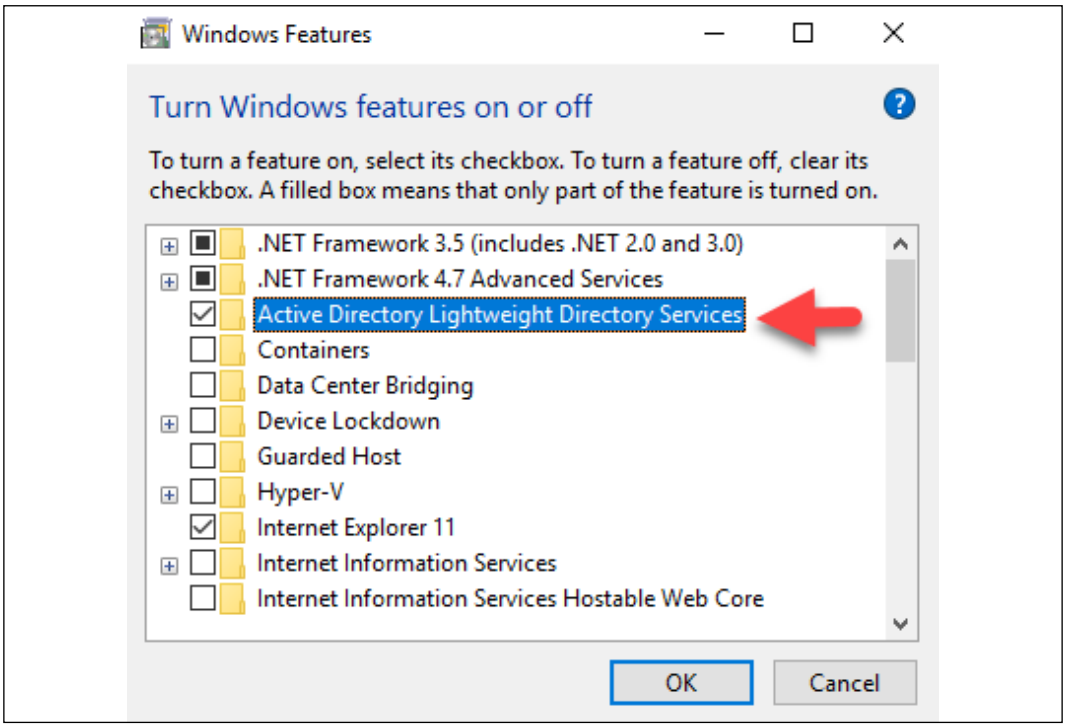


Figure 11.8: AD LDS Windows Features

This option is available on all desktop operating systems after Windows Vista. If it's Windows 7, it needs to be downloaded from the Microsoft site. In order to manage objects, you need to install **Remote Server Administration Tools** (**RSAT**) or use PowerShell to manage them.

AD replication

Healthy replication is a must for an AD environment. AD uses a multi-master database, so every domain controller in the environment should be aware of every change in an AD database. As well as this, domain controllers should also know about changes in group policies, startup scripts, preference settings, and more. When it comes to replication, it is not only the replication service that is responsible for it. There should be stable network connectivity between domain controllers. This communication media can be copper cables, fiber cables, or even a **Software-Defined Network** (**SDN**). In this section, we are going to look at how we can use the AD-integrated features to maintain healthy replication.

FRS versus DFSR

Windows Server 2000 and 2003 use FRS to replicate the `SYSVOL` folder content between domain controllers. With Windows Server 2008, FRS was deprecated and Microsoft introduced DFSR for `SYSVOL` folder replication:

FRS	**DFSR**
FRS is an outdated protocol and no development or investment was put into it after the release of Windows Server 2003 R2. No bug fixes or updates were released. Outdated protocols can lead to security threats to systems, as they cannot be tested against modern security threats.	Continuous improvements and investments were made toward the DFSR protocol and it was continuously tested against emerging threats.
FRS uses the last-write-wins algorithm. When the system detects a file change in one of the `SYSVOL` folders, it becomes the authoritative server and will replicate the *entire* file to other domain controllers. It will not merge the changes. It doesn't matter how small the change is, it always copies the entire file, which can cause performance issues, especially if the domain controllers are connected via slow links.	DFSR allows you to replicate partial file changes using block-level replication. It also supports asynchronous file replication via slow links. If you are running Enterprise Edition, it can use cross-file **Remote Differential Compression** (**RDC**) to create files on the client side using common blocks used by similar files. It will reduce the amount of data that needs to be transferred via the links.

FRS uses NTFS file compression in the staging folder.	File compression can be controlled based on the file type.
There is no interface for monitoring (API or WMI), and the GUI tools for managing services are very limited. These GUI tools are no longer available after Windows Server 2003. They do not support monitoring using the System Center Operations Manager.	There is an enhanced GUI tool for managing related services, and the GUI can use WMI to monitor the health of the service. It also has management packs developed for monitoring through the System Center Operations Manager.
FRS does not have a reporting system to generate diagnostic reports if required. Diagnostics will only be based on events. It also has limited counters to review performances using PerfMon.	DFSR provides support for generating health reports in XML or HTML format. It also includes many counters to review the performance stats using PerfMon.
FRS does not have a system for auto-healing if there is file corruption.	DFSR is capable of auto-healing when it detects data corruption.
FRS does not fully support the RODC environment, and it can have data synchronization issues.	DFSR is fully supported for RODC replication.
FRS does not have advanced audit capabilities, as it only contains limited event logs and debug logs.	DFSR generates data-rich events and logs that can be used to audit, troubleshoot, and debug service failures.

Even though FRS is deprecated with Windows Server 2008, it can still be used for replication if you have migrated from Windows Server 2000 or 2003. Most of the time, engineers forget to migrate to DFSR as part of the upgrade projects. FRS to DFSR migration cannot happen automatically and a few manual steps are involved. This can be done using the `Dfsrmig.exe` utility. In order to perform the migration from FRS to DFSR, your domain and forest functional levels should be at a minimum of Windows Server 2008.

Windows Server 2022 cannot promote as a domain controller if the existing domain is still using FRS for SYSVOL replication. If we try to set up Windows Server 2022 as an additional domain controller, we will get the following error: *The specified domain %1 is still using the File Replication Service (FRS) to replicate the SYSVOL share. FRS is deprecated. The server being promoted does not support FRS and cannot be promoted as a replica into the specified domain. You MUST migrate the specified domain to use DFS Replication using the DFSRMIG command before continuing. For more information, see* `https://bit.ly/30WjqPl`.

The only workaround for this is to migrate an existing domain from FRS to DFSR.

A step-by-step guide for FRS to DFSR migration is available on `https://bit.ly/2Zj1huT`.

AD sites and replication

AD components represent the physical and logical structure of a business. AD forests, domains, organization units, and objects, such as computers and users, represent the logical structure of the business. AD roles and features, such as AD CS, AD RMS, and AD FS, can be used to represent the organization's operational and security requirements. In an infrastructure, all these components are communicating with each other using physical connections such as copper or fiber. Based on the business requirements, AD may have to be extended to remote geographical locations. This can vary from different buildings in the same location to locations on different continents. These remote networks may use various connection methods to maintain communication between each other. It can be VPN, copper leased lines, fiber connections, or even satellite connections. By default, AD will not understand the network topology underneath it. If we consider the **Open Systems Interconnection** (**OSI**) model, AD is operated in the application layer and physical connections are represented by the network layer. There are three main reasons why AD should also be aware of this physical network topology, which are as follows:

- Replication
- Authentication
- Service location

From the preceding list, I would say replication is the most important thing to consider.

Replication

The success of an AD largely depends on healthy replication. Every domain controller in the network should be aware of every change in configuration. When a domain controller triggers a sync, it passes the data through the physical network to the destination. It consumes the *bandwidth* for the data. Depending on the used media and available bandwidth, the impact made by this replication traffic will vary. If it has high-speed links, such as 40 Gbps, 10 Gbps, 1 Gbps, or 100 Mbps, then the impact made by replication traffic will be very low. But in slow links, such as 128 Kbps or 256 Kbps, the impact will be significantly higher.

Most of the time, links between remote networks are slow links. In such situations, there should be a way to optimize the replication based on the available bandwidth. Later on in this chapter, we will be looking into this in detail.

Authentication

When a user tries to authenticate to a system, the request should be processed by a domain controller. If all the domain controllers are located in one geographical location, it doesn't matter which domain controller processes the request. But if it's between remote networks, the time it takes to process the request will depend on the availability of the network links and the number of hops it needs to travel through to reach a domain controller.

As an example, let's assume that Rebeladmin Corp. has an AD infrastructure and it is stretched between two offices in London and Seattle. It has domain controllers located in both locations. If a user logs in to a PC in the London office, it doesn't make sense to have the authentication request processed by a domain controller in the Seattle office. This is because the request needs to pass through a few network hops and a slow link. If large numbers of requests are processed, the majority of the slow link bandwidth will be used by these requests. Ideally, it should be processed by the closest domain controller, which is located at the same location. Then, there are no additional hops to pass and no bandwidth limitations. Also, it will not depend on the availability of the link between the two locations in order to process the requests. Therefore, AD should force identities located in remote networks to authenticate via its closest domain controllers.

Service locations

In a remote network, there can be different server roles and AD-integrated applications running. Similar to authentication requests, when users or computers try to use a service or application, it should be processed by the closest application server. This will improve the user experience and the reliability of the application or the services. To do this, there should be an automated way to process these service requests and point users to the closest servers.

The answer to all the aforementioned requirements is the AD site and related components. These allow us to form the physical network topology within the AD environment. Then, AD is aware of where its components are located and how they're connected with each other. Based on this data, AD will allow us to control the replication over slow links and point the authentication and service requests to the closest servers.

So, let's go ahead and look further into AD sites and their related components.

Sites

Sites can be explained as physical locations that contain various AD objects. We should be able to describe these objects using their boundaries. As an example, users, computers, and network devices located in an office location in London can be treated as a site, and these can be identified as unique from similar objects located in the Seattle office. The AD site topology can be divided into four different designs:

- **Single domain-single site**: This is the most common setup for small- and medium-sized businesses. In this setup, there is one site and one domain. When we set up the first domain controller in the infrastructure, it is set up as a single domain-single site by default. This is easy to maintain.
- **Single domain-multiple sites**: In this setup, the infrastructure has only one domain, and it's extended to multiple sites. These sites can be different buildings on the same campus or different geographical locations. Sites are interconnected using physical network links.
- **Multiple domains-single site**: In one physical site, there can be multiple AD domains. Replications between domains will depend on the logical topology. The replication bandwidth impact is minimal, as domain controllers communicate with each other using fast LAN connections.
- **Multiple domains-multiple sites**: In this setup, multiple domains will be placed in multiple sites. In such an environment, replication will depend on the logical topology as well as the physical topology. In some environments, domains will be limited to certain sites, and in others, domains will be extended to multiple sites.

Once we decide on the site topology, the next step is to configure subnets.

Subnets

Subnets represent IP address ranges in each site. These can represent the subnets allocated in network devices, but do not need to have the same CIDR notation. For example, if a site uses 10-20 class C subnets, instead of adding all these as the AD site's subnet, we can summarize them all into a class B subnet and use it. Based on this subnet information, AD helps objects to locate the closest domain controller. When physical subnets are added or removed, that information should reflect in the AD site configuration as well; otherwise, AD will pass unmanaged traffic via slow links.

Site links

Site links represent the physical connection between sites; however, site links don't control the network level routing or connectivity between sites. It's still being handled by the underlying WAN links and network devices. Site links allow replication jobs to be scheduled and bandwidth to be controlled (link cost).

Site link bridges

Site link bridges contain multiple site links. These allow transitive communication between each site link under the bridge. By default, all site links are treated as bridges. In some cases, not all site links need to talk to each other. They are controlled by the routing rules in the network devices; if they're set up that way, the default behavior of the link bridges needs to be modified, and make sure to disable **Bridge all site links**:

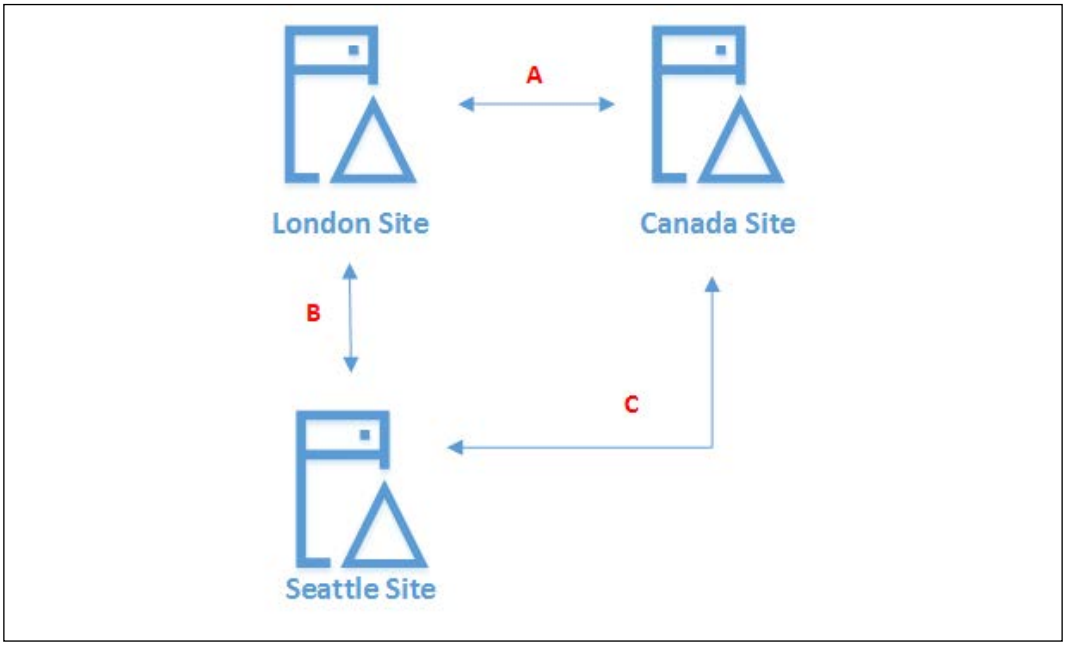


Figure 11.9: Site-link bridges example

The best way to explain site link bridges is to use this simple example. According to the preceding diagram, for **London Site** to reach **Canada Site**, there are two paths: one is using **A** and the other one is using **B** and **C**. Therefore, on **Seattle Site**, we can create a site link bridge that includes site links **B** and **C**, and present it as an alternative path for reaching **Canada Site**.

Managing AD sites and other components

There are two ways to manage AD sites and related components. One option is to use the AD Sites and Services MMC, and the other one is to use PowerShell cmdlets. To add/edit/remove sites and related configurations, we need to have Domain Admin/Enterprise Admin privileges.

The AD Sites and Services MMC will be available in any server that has the AD DS service enabled or any server/computer that has RSAT installed.

The AD PowerShell module will also be available in any server that has the AD DS role enabled or has the RSAT tools installed.

Managing sites

When the first domain controller is introduced into the infrastructure, the system creates its first site as `Default-First-Site-Name`. This can be changed based on the requirements. We can review the existing site's configuration using the following PowerShell cmdlet:

```
Get-ADReplicationSite -Filter *
```

It will list the site's information for the AD environment.

Our example only has the default AD site. As this is the first step, we need to change it to a meaningful name so we can assign objects and configurations accordingly. To do that, we can use the `Rename-ADObject` cmdlet:

```
Rename-ADObject -Identity "CN=Default-First-Site-Name,CN=Sites,CN=Confi
guration,DC=rebeladmin,DC=com" -NewName "LondonSite"
```

The preceding command renames the `Default-First-Site-Name` site to `LondonSite`. We also can change the site configuration values using the `Set-ADReplicationSite` cmdlet:

```
Get-ADReplicationSite -Identity LondonSite | Set-ADReplicationSite
-Description "UK AD Site"
```

The preceding command changed the site description to `UK AD Site`.

We can create a new AD site using the `New-ADReplicationSite` cmdlet. The full description of the command can be viewed using `Get-Command New-ADReplicationSite -Syntax`:

```
New-ADReplicationSite -Name "CanadaSite" -Description "Canada AD Site"
```

The preceding command creates a new AD site called `CanadaSite`.

Once the sites are created, we need to move the domain controllers to the relevant sites. By default, all the domain controllers are placed under the default site, `Default-First-Site-Name`.

In the following command, we are listing all the domain controllers in the AD environment with filtered data to show the `Name,ComputerObjectDN,Site` attribute values:

```
Get-ADDomainController -Filter * | select Name,ComputerObjectDN,Site |
fl
```

Now we have the list of domain controllers; in the next step, we can move the domain controller to the relevant site:

```
Move-ADDirectoryServer -Identity "REBEL-SDC-02" -Site "CanadaSite"
```

The preceding command will move the `REBEL-SDC-02` domain controller to `CanadaSite`.

During the additional domain controller setup, we can define which site it will be assigned to. If the site already has domain controllers, it will do the initial replication from the site local domain controller. If it doesn't, it will replicate from any selected domain controller or, if not, from any available domain controller. If the link bandwidth is an issue, it's recommended to promote the domain controller from a site that has fast links, and then move the domain controller to the relevant site.

Managing site links

Now that we have the sites set up, the next step is to create site links. Using site links, we can manage the replication schedule and the bandwidth.

The site link cost

The site link cost defines the nearest resources if the on-site resource is not available. In a physical network, the quality of inter-site links is measured based on link speed, latency, and availability. Then we can use that information to decide on the site link cost for each connection. For example, let's assume site A and site B are connected via a 100 Mbps link. Site A and site C are connected via a 512 Kbps link. If we only consider the bandwidth, site A will prefer site B as the closest site. But this link had a few failures last month, and so 512 Kbps is more reliable. By changing the site link cost, we can force site A to use site C as the preferred closest resource site.

In the following example, we have three sites, and each site has two site links to connect to each other:

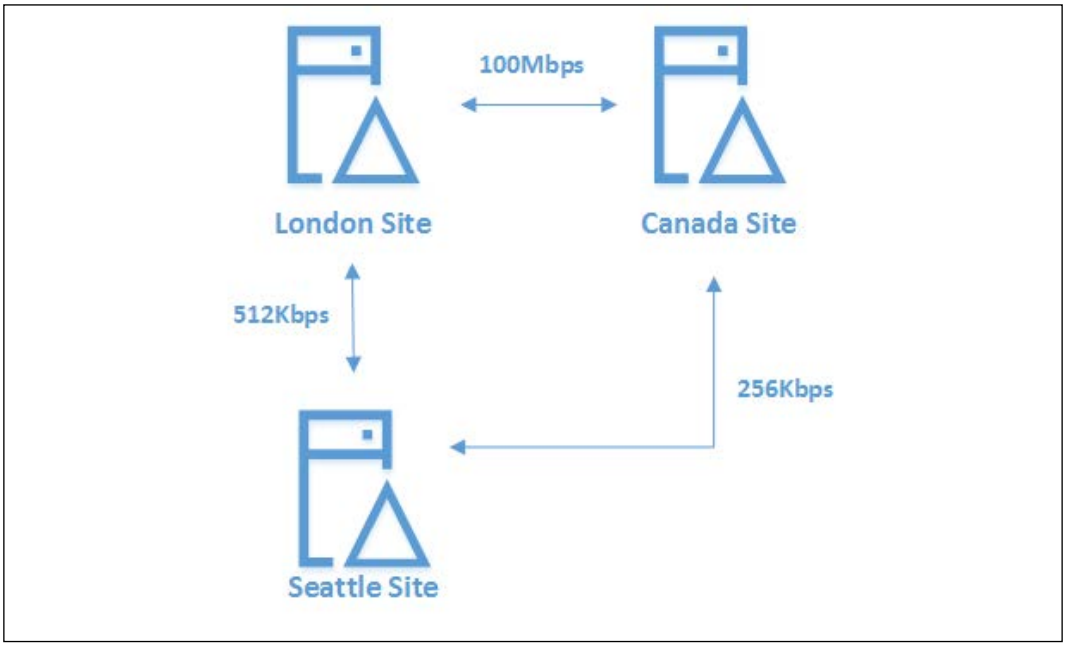


Figure 11.10: Site link cost example

If the line latency and availability are not a problem, then for **London Site**, the first preferred site should be **Canada Site**. If that link fails, it still uses **Seattle Site**. For **Canada Site**, **London Site** is the first preference and the next is **Seattle Site**. For **Seattle Site**, the first preference is **London Site** and the next is **Canada Site**. But here, we are only considering the bandwidth. The link preferences can be modified by changing the site cost. By default, every site link gets the cost of 100, which is just a number, but we'll use it to map the site link speeds later.

The site link that holds the lowest site cost value will be the first preference. When the system determines the code to a destination, it does not consider the direct links. In the same way, a network topology will find its best route. In the preceding example, if **London Site** wants to reach **Canada Site**, there are two paths: one is the direct link and the other one is via **Seattle Site**. So, when it considers the best path, it will calculate the cost value of the direct link against the cost value of **London Site | Seattle Site | Canada Site**.

The following list includes the preferred site cost value based on the bandwidth. This cost value is calculated by dividing 1,024 by the log of the available bandwidth (Kbps).

As an example, the logarithm of 512 is 2.709. When we divide 1,024 by 2.709, we get 377.999. So, the link cost of a 512 Kbps line is 378:

Available bandwidth	Cost
9.6 Kbps	1,042
19.2 Kbps	798
38.4 Kbps	644
56 Kbps	586
64 Kbps	567
128 Kbps	486
256 Kbps	425
512 Kbps	378
1,024 Kbps	340
2,048 Kbps	309
4,096 Kbps	283
10 Mbps	257
100 Mbps	205
1,000 Mbps	171

Inter-site transport protocols

There are two transport protocols that can be used for replication via site links. The default protocol used in a site link is IP, and it performs synchronous replication between available domain controllers. The SMTP method can only be used between sites, if replicating domain controllers in different domains. Domain controllers in the same domain should replicate using the IP method.

Replication intervals

By default, in a default site link, a replication occurs *every 180 minutes*. Based on the requirements, this can be changed to the value we need. If required, it also allows us to disable the replication schedules completely and rely on manually triggered replications.

Replication schedules

By default, site replication is happening 24/7. Based on the site bandwidth usage, this can be changed. For example, if it's a slow link, it is best to set the replication after operating hours and during lunch hours. This will minimize the replication traffic impact on slow links and allow the organization to use the link bandwidth for other mission-critical traffic. Before the actual replication schedule changes, it's important to evaluate the consequences. If we add/modify objects and policies, they will not replicate between sites unless they match the replication schedule or forcefully initiate replication.

In order to set up the site links, we can use the `New-ADReplicationSiteLink` cmdlet:

```
New-ADReplicationSiteLink -Name "London-Canada" -SitesIncluded
LondonSite,CanadaSite -Cost 205 -ReplicationFrequencyInMinutes 30
-InterSiteTransportProtocol IP
```

The preceding command creates a new site link called `London-Canada`, which includes `LondonSite` and `CanadaSite`. The site cost is set to `205`, and the replication intervals are set to every `30` minutes. Its transport protocol is set to `IP`.

We can do the same configuration by using the **AD Sites and Services** MMC:

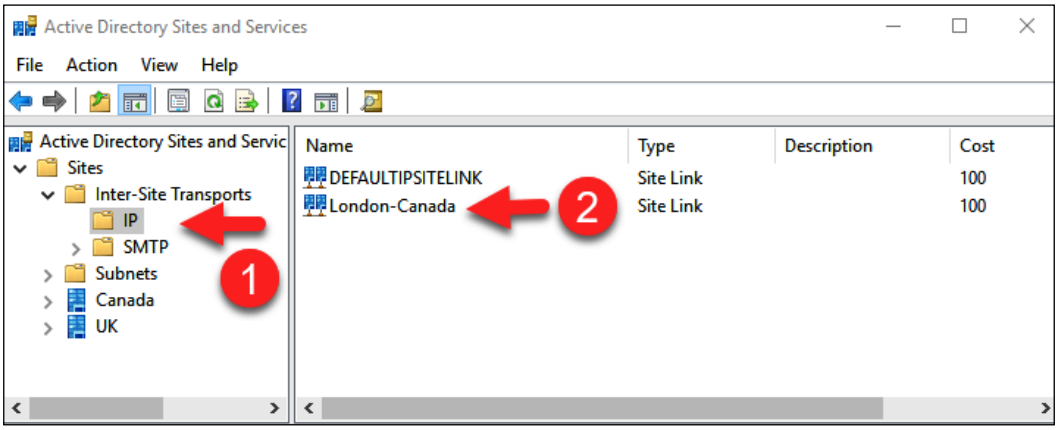


Figure 11.11: Site link

We can change the replication schedule using the `-ReplicationSchedule` option or the GUI. In order to change it using the GUI, click on the **Change Schedule...** button. Then, in the window, you can change the replication schedule.

In this demonstration, I changed the replication to happen from Monday to Friday, from 6:00 a.m. to 10:00 a.m:

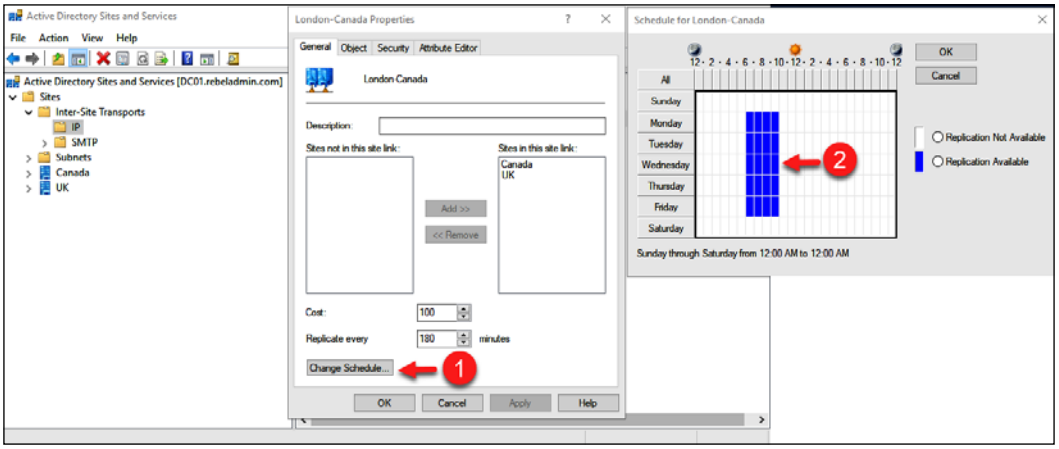


Figure 11.12: Site link replication schedule

The site link bridge

We can create the site link bridge using the `New-ADReplicationSiteLinkBridge` cmdlet:

```
New-ADReplicationSiteLinkBridge -Name "London-Canada-Bridge"
-SiteLinksIncluded "London-Canada","London-CanadaDRLink"
```

The preceding command creates a new site link bridge called `London-Canada-Bridge` using two site links: `London-Canada` and `London-CanadaDRLink`.

Using the `Set-ADReplicationSiteLinkBridge` cmdlet, the existing site link bridge value can change:

```
Set-ADReplicationSiteLinkBridge -Identity "London-Canada-Bridge"
-SiteLinksIncluded @{Remove='London-CanadaDRLink'}
```

The preceding command removes the `London-CanadaDRLink` site link from the existing site link bridge, `London-Canada-Bridge`:

```
Set-ADReplicationSiteLinkBridge -Identity "London-Canada-Bridge"
-SiteLinksIncluded @{Add='London-CanadaDRLink'}
```

The preceding command adds the given site link to the existing site link bridge.

Bridgehead servers

The **Knowledge Consistency Checker** (**KCC**) is a built-in process that runs on domain controllers and is responsible for generating replication topology. It will configure the replication connection between domain controllers. When it comes to replication between sites, the KCC selects a domain controller as a *bridgehead server*, which sends and receives replication traffic for its site. If you have multiple domains in multiple sites, each domain should have its own bridgehead server. A site can have multiple bridgehead servers for the same domain, but at a given time, only one will be active. This is decided based on the domain controller's lowest GUID value. If AD involves an intra-site replication, AD automatically selects the bridgehead servers. However, there are situations where you may prefer a specific server to act as a bridgehead server.

By opening the properties of the domain controller, you can choose what you want to set as a bridgehead server. The best practice is to set the most reliable domain controller as the bridgehead server:

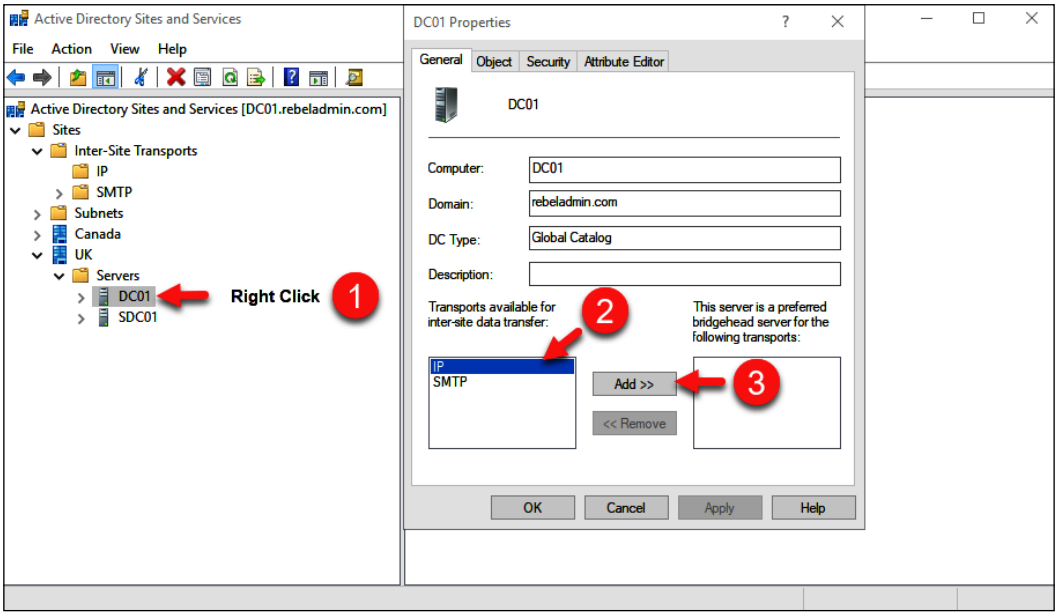


Figure 11.13: Bridgehead server selection

Managing subnets

Now that we have the sites and site links set up, the next step is to assign the subnets to each site. These can be set up using the `New-ADReplicationSubnet` cmdlet:

```
New-ADReplicationSubnet -Name "192.168.0.0/24" -Site LondonSite
```

The preceding PowerShell command creates a new subnet, `192.168.0.0/24`, and assigns it to `LondonSite`.

Using `Set-ADReplicationSubnet`, we can change the value of the existing subnet:

```
Set-ADReplicationSubnet -Identity "192.168.0.0/24" -Site CanadaSite
```

The preceding command changes the site of the `192.168.0.0/24` subnet to `CanadaSite`.

We can use the `Get-ADReplicationSubnet` cmdlet to find the subnet data:

```
Get-ADReplicationSubnet -Filter {Site -Eq "CanadaSite"}
```

The preceding command lists all the subnets under `CanadaSite`.

How does replication work?

By now, we know the logical and physical components involved in the AD replication process. Now, it's time to put all these together and understand exactly how AD replication works.

In the AD environment, there are two main types of replication:

- Intra-site replication
- Inter-site replication

In any given AD environment, there will be intra-site replication. So let's go ahead and understand how this default replication works.

Intra-site replication

As the name implies, this covers the replications happening within the AD site. By default (according to Microsoft), any domain controller will be aware of any directory update within 15 seconds. Within the site, despite the number of domain controllers, any directory update will be replicated in less than a minute:

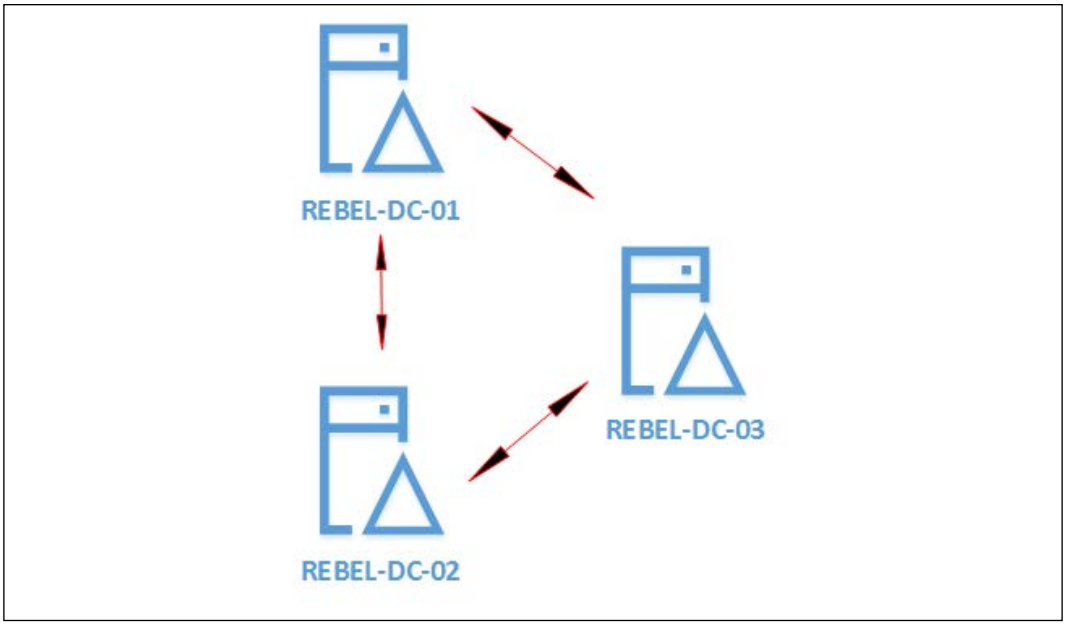


Figure 11.14: Intra-site replication

Within the site, the replication connections are being performed in a ring topology, which means that any given domain controllers have two replication links (of course, if there are a minimum of three domain controllers). This architecture will prevent domain controllers from having endless replication loops.

For example, if there are five domain controllers and if all are connected to each other with one-to-one connections, each domain controller will have four connections, and when there is an update to one of the domain controllers, it will need to advertise this to four domain controllers. Then, the first one to receive the update will advertise to its four connected domain controllers, and it goes on and on. There are too many replication processes to advertise, listen, and sort out the conflicts.

But in a ring topology, despite the number of domain controllers in the site, any given domain controller only needs to advertise or listen to two domain controllers at any given time. With this replication topology, there is no need for manual configuration, and AD will automatically determine the connections it needs to make. When the number of domain controllers grows, the replication time can grow as well as it's in a ring topology. But to avoid latency, AD creates additional connections. This is also determined automatically, and we do not need to worry about these replication connections.

Inter-site replication

If the AD environment contains more than one site, a change in one site needs to be replicated over to the other sites. This is called an **inter-site replication**, and its topology is different from intra-site replication. Replication within the site always benefits from high-speed links. But when it comes to the connection between sites, things such as bandwidth, latency, and reliability all create a direct impact.

In the previous section, we discussed site links, site costs, and replication schedules, which we can use to control intra-site replications:

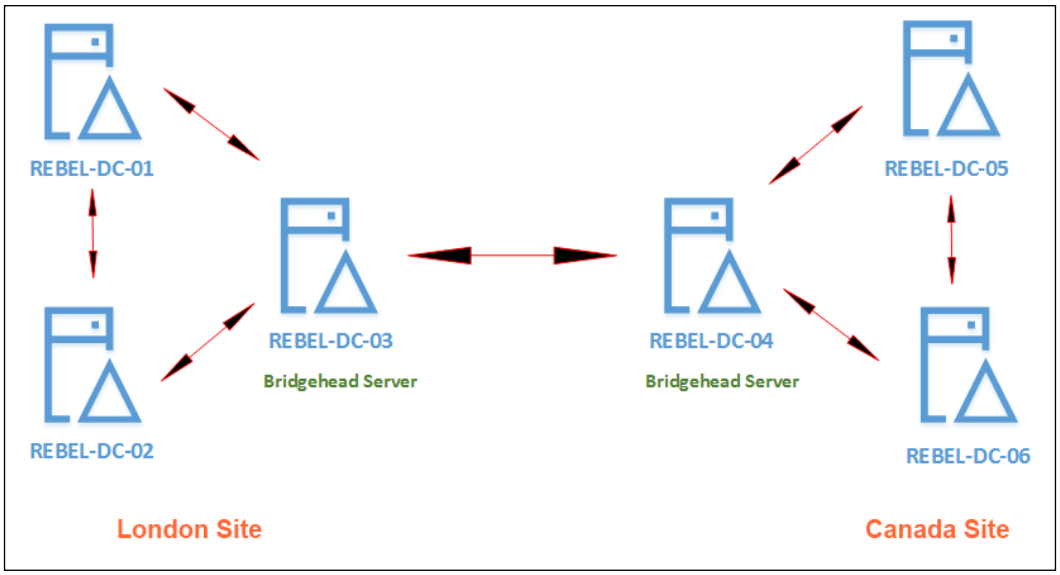


Figure 11.15: Inter-site replication

Here, replication happens via *site links*. The replication within each site still uses the ring topology. In the preceding example, let's assume an object has been added to **REBEL-DC-02** in **London Site**. Now, based on the topology, it will be advertised to **REBEL-DC-03** too. But apart from being the domain controller, this particular domain controller is a bridgehead server as well. So, it is this server's responsibility to advertise the updates it received into the bridge server in **Canada Site**, which is **REBEL-DC-04**. Once it receives the update, it will advertise to the other domain controllers in the site. The replication between sites still needs to follow the replication schedules.

AD DS automatically selects the bridgehead server for a site. But there are situations where engineers want to select a server, but they prefer it to be the bridgehead server. In the previous section, we looked at how we can force a domain controller to become the *preferred bridgehead server*.

The KCC

When I discussed AD replication, I mentioned that AD is automatically creating replication links and selecting bridgehead servers. But how does it really do that?

The KCC is a built-in service in AD domain controllers and it is responsible for generating and maintaining the replication topology for intra- and inter-site replications. Every 15 minutes, the KCC will revalidate its existing replication topology and make topology changes if required. It gives enough time for domain controllers to replicate the changes if the existing replication topology is valid.

When it comes to inter-site replication, the KCC selects a single KCC holder in a remote site to act as the **Intersite Topology Generator** (**ISTG**), and the ISTG's responsibility is to select the bridgehead servers for replication. The ISTG creates the view of the replication topology for all the sites it is connected to. The ISTG is responsible for deciding the topology for the site, and individual domain controllers (such as the KCC) are responsible for making topology decisions locally.

The best way to understand the KCC is to compare it with a network routing protocol. A network routing protocol is responsible for maintaining a routing path for connected networks. If network A needs to communicate with network B, the routing table will tell it what path to go to. In the same way, the topology created by the KCC will tell us how domain controller A can replicate the changes in domain controller B. When I have worked on AD projects, I have seen engineers create manual replication links between domain controllers. But I really doubt whether someone can be smarter than the KCC when it comes to deciding the replication topology.

By running the command `Repadmin /kcc`, we can force the KCC to immediately recalculate the inbound replication topology.

How do updates occur?

We now know how the replication topology works, but how exactly does a domain controller know when an update has occurred? And how do the connected domain controllers know when to trigger replications? Let's go ahead and explore the technology behind it.

The Update Sequence Number (USN)

The USN is a 64-bit number that is allocated to the domain controller during the **DCPromo** process. When there is any object update, the USN allocated to the domain controller will be increased. As an example, let's assume that domain controller A had an initial USN value of 2,000 assigned to it.

If we add 5 user objects, the new USN will be 2,005. This number can only increase; it cannot decrease. The USN is only valid for its own domain controller. It is not technically possible for two domain controllers in the site to have the same USN assigned.

The Directory Service Agent (DSA) GUID and invocation ID

Domain controllers involved in the replication process are identified using two unique identifiers. The first one is the DSA GUID. It is generated during the **DCPromo** process, and it will never change during the lifetime of the domain controller. The next one is the invocation ID. In a restore process, it will change; otherwise, the existing domain controllers will identify it as an existing domain controller and will not replicate data over.

The High Watermark Vector (HWMV) table

The HWMV table is maintained locally by each domain controller in order to keep track of the last change from its replication partner for the given **Naming Context** (**NC**). Domain controllers have three NCs: schema NCs, configuration NCs, and domain NCs. There is an HWMV table for each NC. The table contains the latest USN value it received from its replication partner for a given NC. Based on that, the domain controller decides where to start the replication process.

The Up-To-Dateness Vector (UTDV) table

The UTDV table is maintained locally by each domain controller to prevent unnecessary replications. UTDV also—per NC and domain controller—has a minimum of three UTDV tables. The UTDV table contains the highest UPN value it learned from any connected domain controller per NC basis. This prevents domain controllers from replicating the same changes over and over. For example, if domain controller A received a domain NC from domain controller B, it would update the UTDV table and update the UPN value for it. Based on that, it would not retrieve the same update from the other connected domain controllers. Because of UTDV, domain controllers will not send any data to their replication partners if they have already received it from someone else. This is called **propagation dampening**:

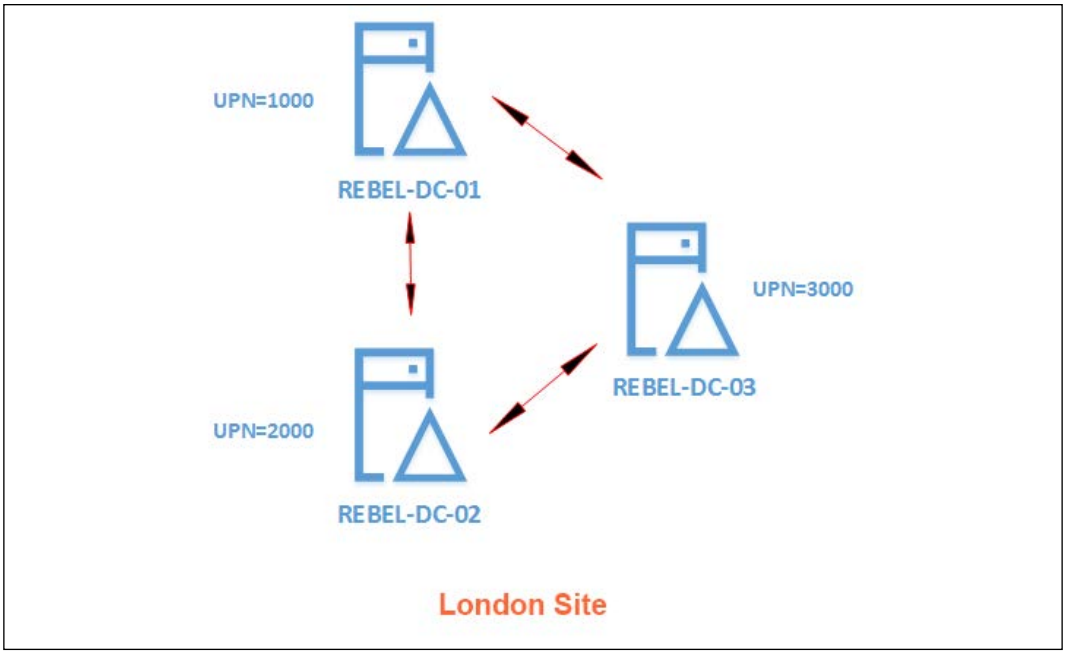


Figure 11.16: UPN value

Let's go ahead and summarize all these theories with an example. In the preceding example, we have three connected domain controllers. Each has an initial UPN value assigned. Now, let's assume, a new user object has been added in **REBEL-DC-01**. So its UPN has increased to **1,001**. At this point, the domain controller knows about its connected replication partners based on the DSA GUID and invocation ID. **REBEL-DC-02** is already aware of the last UPN value it received from **REBEL-DC-01** as it is stored on the HWMV table. Before it updates, it checks the UTDV table to make sure it didn't receive the same update via **REBEL-DC-03**. If not, then it replicates and increases its UPN and the values in the HWMV and UTDV tables.

Summary

We started the chapter by looking into AD LDS and its capabilities. Then, we moved on to AD replication. In that section, we focused on the physical and logical components involved in AD replication and how they can be used to optimize complex replication requirements. More importantly, we also looked into how AD replication happens behind the scenes.

12

Active Directory Services – Part 02

During the COVID-19 pandemic, lots of businesses have started to collaborate with each other. Sometimes these businesses have had to share resources among them. For example, one company may want to access an Active Directory integrated web application of another company. In such a scenario, how can we grant access to the application with minimal effect? An Active Directory trust allows you to connect two different Active Directory domains/forests together and allows users to share resources among them. In this chapter, we will look into Active Directory trusts in detail.

In an Active Directory environment, each and every domain controller holds sensitive information about identities. Therefore, the security of domain controllers is crucial. With Windows Server 2008, Microsoft introduced **read-only domain controllers** (**RODCs**), which are ideal for sites where we can't guarantee physical security. In this chapter, we will learn about how RODCs work and how to configure them. Last but not least, we will learn about Active Directory database maintenance, including database defragmentation, backup, and recovery.

In this chapter, we will cover the following topics:

- Active Directory trusts
- Active Directory database maintenance
- RODCs in action
- AD DS backup and recovery

Active Directory trusts simplify resource sharing between Active Directory forests. Let's start this chapter with this important topic.

Active Directory trusts

I bought a new bicycle for my daughter on her last birthday. It's almost summer here in the UK and the weather is getting better. So, on a sunny Sunday evening, we wanted to go to Richmond Park so she could ride her new bike.

She asked if her friend Georgina could join us. I agreed and we all went to the park. Georgina liked my daughter's new bike very much. My daughter went ahead and asked her if she wanted to ride it. Once Georgina agreed, my daughter let her ride it. Georgina is her friend and she has known her for years. She trusts her and she was happy to share the bike with her. In the same way, modern businesses collaborate with each other more than ever. The rapid digital transformation of businesses due to the pandemic has opened up new areas of opportunities. As part of the collaboration process, sometimes it is required to share resources between organizations. This can be in the form of access to an application, access to data shares, or even access to servers.

Active Directory trusts allow users in one domain to access resources in another domain. It is not a must to use Active Directory in both domains to perform trust. It is possible to create a trust between Active Directory and a Kerberos V5 realm that uses a third-party directory service. In this section, we will look into the characteristics of different types of trust and how we can use them for collaboration.

Now, if we go back to the story of my daughter's bicycle, my daughter has known Georgina for years and she already trusts her. This is similar to establishing an initial trust between two domains. Georgina got the chance to try my daughter's bike. This was a one-way agreement and my daughter did not expect anything in return. This is similar to a **one-way trust** between domains. In a one-way trust, one domain trusts another domain and allows users of the second domain to use resources in the first domain. Georgina had a good time, and afterward, her parents asked me about the bike as they wanted to buy a similar bike for her. A few weeks later, we went back to the park, but this time Georgina also had a bicycle with her. After riding the bikes for a while, the girls wanted to swap their bikes and try each other's. This is similar to a **two-way trust** between domains. In a two-way trust, both domain users have access to resources in each other's domains.

Trust direction

In a two-way trust, trust direction is irrelevant as both domains trust each other. But when it comes to a one-way trust, direction is important as it decides which domain is going to be the **trusted domain** and which is going to be the **trusting domain**.

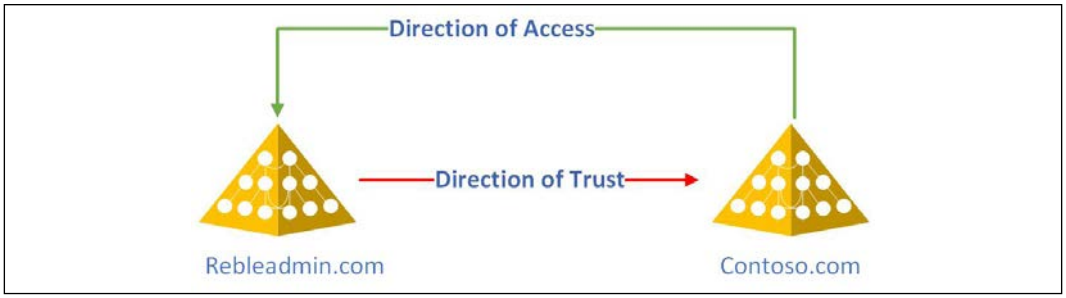


Figure 12.1: Direction of Trust vs Direction of Access

In an Active Directory trust direction configuration, we have three options to choose from:

1. **A two-way trust –** In a two-way trust, both domains trust each other. So, there is no real difference in trust direction.
2. **A one-way incoming trust –** In the above example, the `Contoso.com` domain has a one-way incoming trust from the Rebeladmin.com domain. So, the users of `Contoso.com` can be authenticated in the `Rebeladmin.com` domain.
3. **A one-way outgoing trust –** In the above example, the `Rebeladmin.com` domain has a one-way outgoing trust to `Contoso.com`. This means users of the Contoso.com domain can be authenticated in the `Rebeladmin.com` domain.

The key takeaway from this is an incoming trust allows outgoing access and an outgoing trust allows an incoming trust. In the above example, the `Rebeladmin.com` domain can also be called a **trusting domain** as it is allowing access to its resources from users in a remote domain. We can also call this remote domain – `Contoso.com` – a **trusted domain**.

Transitive trusts vs Non-Transitive trusts

An Active Directory forest can contain multiple domains. All domains in an Active Directory forest trust each other by default. Let's assume that we have two Active Directory forests called `rebeladmin.com` and `contoso.com`. The rebeladmin.com forest has two more domains called `toronto.rebeladmin.com` and `london.rebeladmin.com`. When we create a forest trust, contoso.com will also trust the `toronto.rebeladmin.com` and `london.rebeladmin.com` domains. This is because forest trusts are also **transitive trusts** by default. A transitive trust extends the trust beyond the original trusting domain. Non-transitive trusts only allow trust between the original domains. Active Directory external trusts are **non-transitive** by default.

Active Directory trust types

There are six different types of Active Directory trusts. Some of those trusts are created automatically and some need manual intervention.

1. **Tree-Root Trusts**

   This type of trust will be created automatically when a new domain tree is added to an Active Directory forest. These trusts are created in the root domain of each tree and are also two-way transitive trusts.

2. **Parent-Child Trusts**

   When a new child domain is added to an existing Active Directory environment, a new two-way transitive trust will automatically be established between the child domain and its parent domain.

3. **Forest Trusts**

   Forest trusts are created between two Active Directory forests. These need to be created manually. They are transitive trusts by default but, based on business requirements, they can be one-way or two-way trusts.

4. **External Trusts**

   External trusts are created between domains in different Active Directory forests. These trusts need to be created manually and by default will be non-transitive trusts.

5. **Shortcut Trusts**

   A shortcut trust is explicitly created between two domains in the same Active Directory forest or different forests to improve authentication times by shortening the authentication path. These trusts will be transitive and need to be created manually.

6. **Realm Trusts**

   A realm trust is used between an Active Directory forest and non-Windows Kerberos realms such as Unix and Linux. These trusts need to be created manually and can be one-way or two-way trusts. They can also be transitive or non-transitive trusts.

In the next section, we are going to learn about things we need to consider when we are establishing an Active Directory trust.

Creating an Active Directory trust

Before we establish an Active Directory trust, there are certain prerequisites we need to look into.

Firewall ports

When we consider connectivity between two domains or two forests, we need to make sure relevant traffic is allowed via corporate firewalls. The following list contains the minimum number of ports we need to open between two forests or two domains. The traffic flow is bi-directional, which means both networks need to allow incoming and outgoing connections via the given ports.

Service	Ports
LDAP	TCP 389
LDAPS (SSL)	TCP 636
DNS	TCP/UDP 53
RPC	TCP 135 TCP 1024-65535
SMB	TCP 445
Kerberos	TCP/UDP 88
Global Catalog	TCP 3268
Global Catalog (SSL)	TCP 3269

These ports need to be open between all the domain controllers. In a firewall, it is best to create a device group first and then apply policies/rules to it.

Conditional Forwarding

When we create an Active Directory trust, each forest or domain needs to know how to resolve the DNS name records on each other's forests or domains. As an example, I have two domains in two different forests. One is rebeladmin.com and the other one is contoso.com. To establish a two-way trust, each domain needs to resolve to the correct IP address. To make sure the DNS records resolve correctly, we can set up conditional forwarding in each domain's DNS servers. In a conditional forwarding setup, we need to define which DNS servers need to process DNS queries related to their domain. For example, by using a conditional forwarder, we can say if someone tries to resolve an FQDN related to contoso.com, the query should be processed by using the 10.10.10.1 and 10.10.10.2 DNS servers.

In my demo environment, I have two domains in two forests. Their IP address configuration is as follows:

Domain	Domain Controller	IP Address
rebeladmin.com	DC01.rebeladmin.com	10.1.0.4/24
contoso.com	CON-DC01.contoso.com	10.1.5.4/24

I have all the relevant firewall rules in place to allow traffic between these two domains to establish a trust. Before we go there, let's configure the relevant conditional forwarders.

I log in to DC01.rebeladmin.com and try to ping contoso.com. As expected, it gives me the wrong IP address.

```
PS C:\Users\dfrancis> ping contoso.com

Pinging contoso.com [40.113.200.201] with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 40.113.200.201:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
PS C:\Users\dfrancis>
```

Figure 12.2: Ping test to check DNS

This should resolve to 10.1.5.4. To fix this, let's go ahead and set up a conditional forwarder in DC01.rebeladmin.com for contoso.com and point it to DNS server 10.1.5.4.

```
Add-DnsServerConditionalForwarderZone -Name "contoso.com"
-ReplicationScope "Forest" -MasterServers 10.1.5.4
```

In the preceding command, I am creating an Active Directory integrated conditional forwarder for the contoso.com domain. This will replicate to all domain controllers in the forest.

Once the conditional forwarder is in place, I can resolve it to the correct IP address.

```
PS C:\Users\dfrancis> hostname
DC01
PS C:\Users\dfrancis> ping contoso.com

Pinging contoso.com [10.1.5.4] with 32 bytes of data:
Reply from 10.1.5.4: bytes=32 time=1ms TTL=128
Reply from 10.1.5.4: bytes=32 time=1ms TTL=128
Reply from 10.1.5.4: bytes=32 time=2ms TTL=128
Reply from 10.1.5.4: bytes=32 time=1ms TTL=128

Ping statistics for 10.1.5.4:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 2ms, Average = 1ms
PS C:\Users\dfrancis>
```

Figure 12.3: Ping result after conditional forwarder setup – contoso.com

Before we set up the Active Directory trust, we also need to set up a conditional forwarder in CON-DC01.contoso.com for the rebeladmin.com domain.

```
Add-DnsServerConditionalForwarderZone -Name "rebeladmin.com"
-ReplicationScope "Forest" -MasterServers 10.1.0.4
```

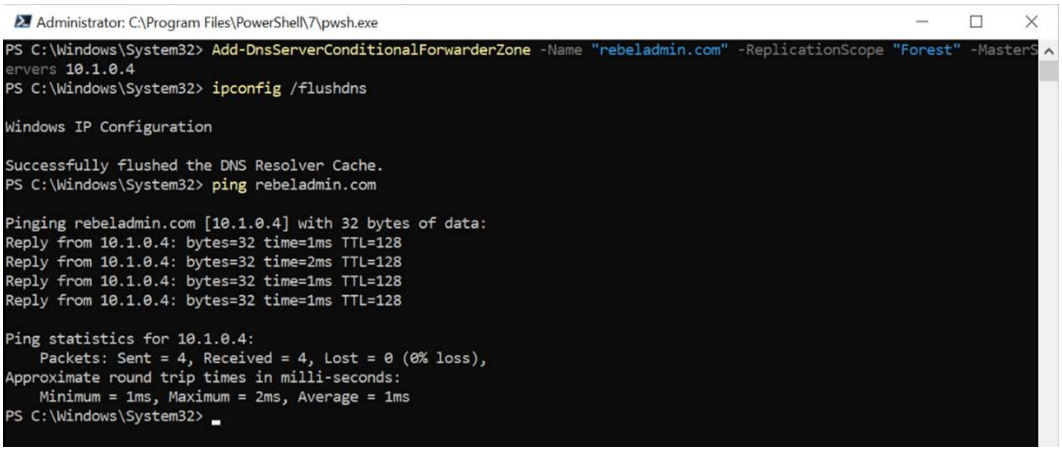


Figure 12.4: Ping result after conditional forwarder setup – rebeladmin.com

As per the above screenshot, the conditional forwarder also works in the contoso.com domain.

Note that instead of a conditional forwarder zone, we can also set up a secondary DNS zone in Contoso DNS servers for the rebeladmin.com domain. This will allow the Contoso domain to resolve DNS requests locally. However, in a trust, we only use certain DNS names – it's a waste to replicate the whole DNS zone. Also, this will expose more data about the rebeladmin infrastructure than required. More info about secondary zones is given in *Chapter 4*, *Active Directory Domain Name System*.

Setting Up an Active Directory Forest Trust

In the previous section, we created the relevant conditional forwarders on both sides. The next step is to establish an Active Directory forest trust:

1. To start the configuration, I logged in to DC01.rebeladmin.com as Enterprise Administrator.

2. Then I launched **Active Directory Domains and Trust MMC**, right-clicked on **rebeladmin.com**, and clicked **Properties**:

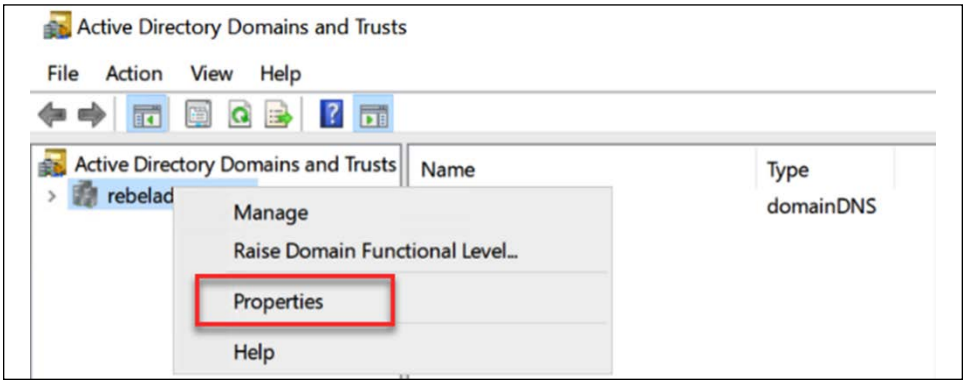


Figure 12.5: Domain Properties

3. Then, click on **Trusts | New Trusts**.
4. This will open up a new wizard. Click on **Next** to start the configuration process.

Figure 12.6: New Trust Wizard

5. In the next window, we need to type the remote domain name. In this example, it's `contoso.com`. Once the details are in place, click on **Next**.

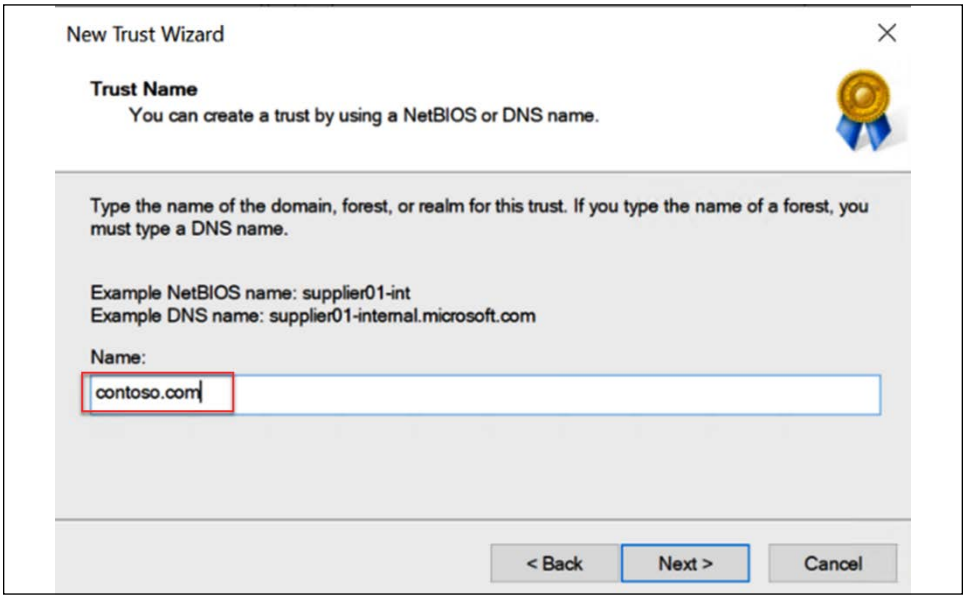


Figure 12.7: Remote Domain Name

6. In the next window, we need to select the trust type. We are going to create a forest trust, so select **Forest trust** and click **Next**.
7. Next, we need to define the direction of the trust. We are going to create a two-way trust so select the **Two-Way** option and click **Next**.

8. When we create a trust, we need to do it from both domains/forests. If we have appropriate permissions for the remote domain, we can create a trust on both sides in one go. In this example, I am going to create a trust on both sides. Once the correct option is selected, click on **Next** to proceed.

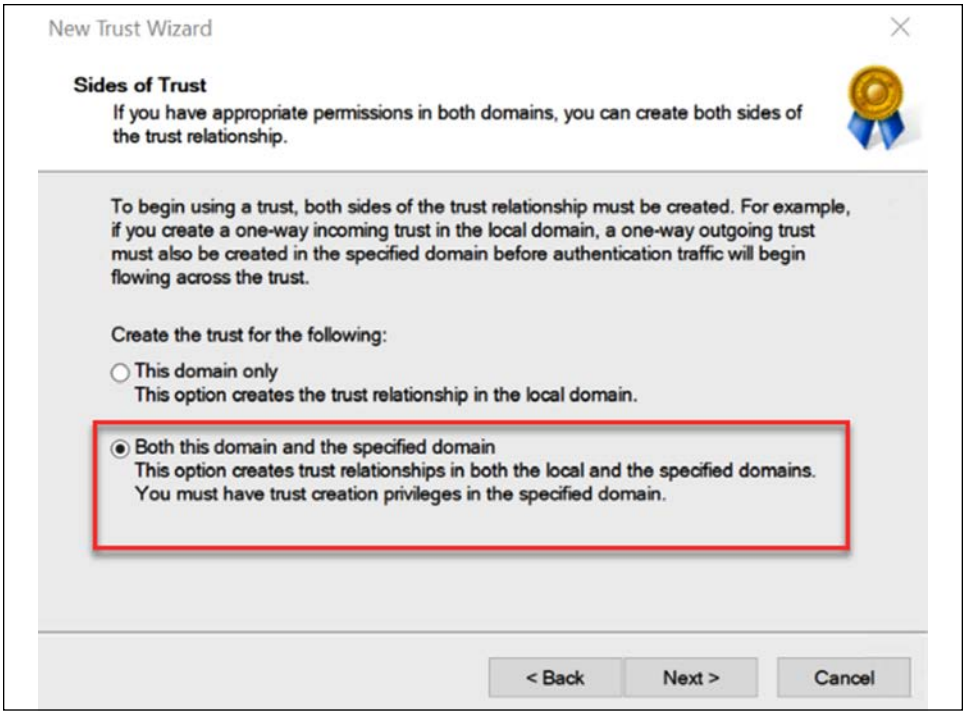


Figure 12.8: Create a trust on both sides

9. In the next window, the system asks for the user name and password for a privileged user in the remote domain to create the trust. Once the relevant information is in place, click on **Next**.
10. Then, we need to define the scope of the authentication for both domains. For both domains, I have selected the default option **Forest-wide authentication**.
11. This completes the configuration of the trust. In the next window, click **Next** to create the trust.

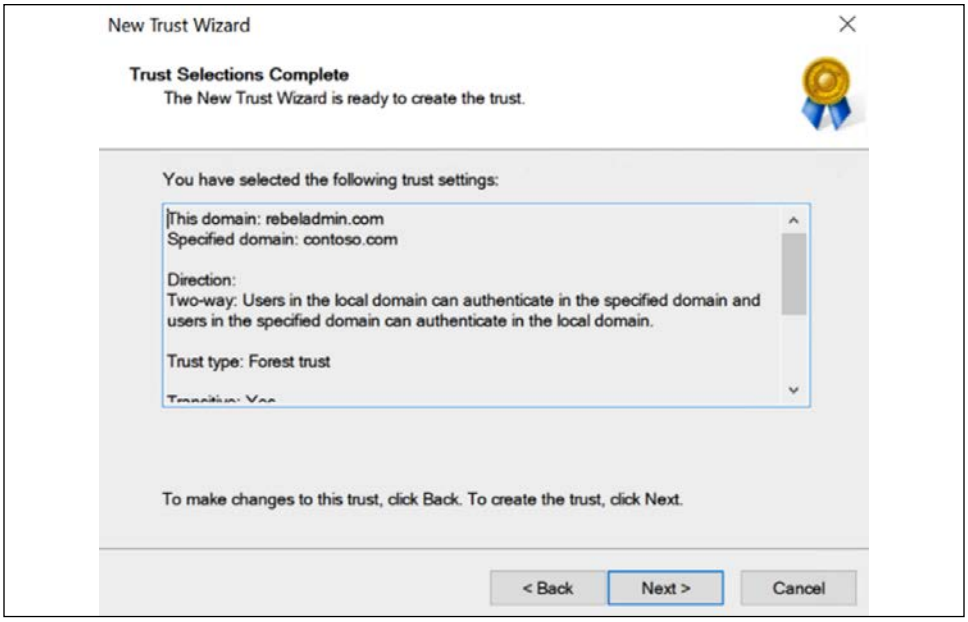


Figure 12.9: Verify Trust configuration

12. Once the trust is created successfully, the system asks if we need to confirm outgoing and incoming trusts. Select **Yes** for both to verify the trusts.

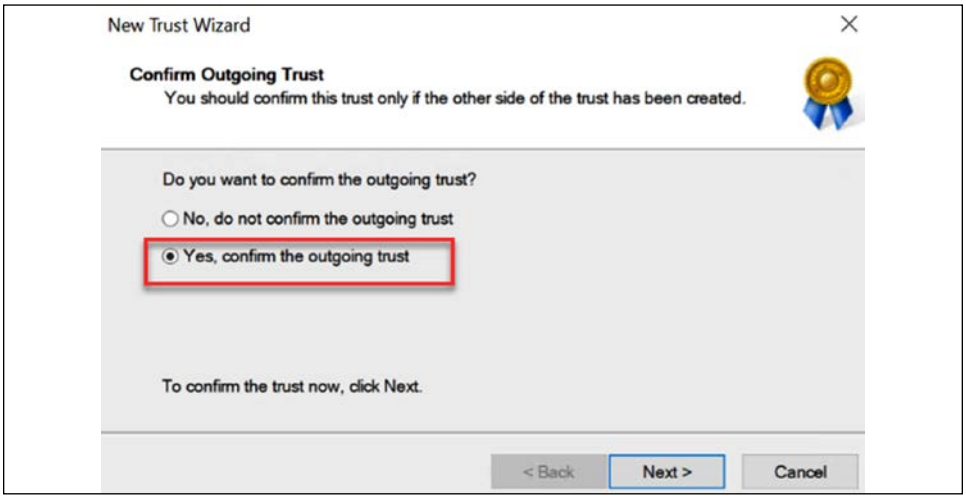


Figure 12.10: Confirm outgoing trust

13. After the system confirms the trusts, click on **Finish** to complete the process.
14. Now we can see the trust is set up in both forests as expected:

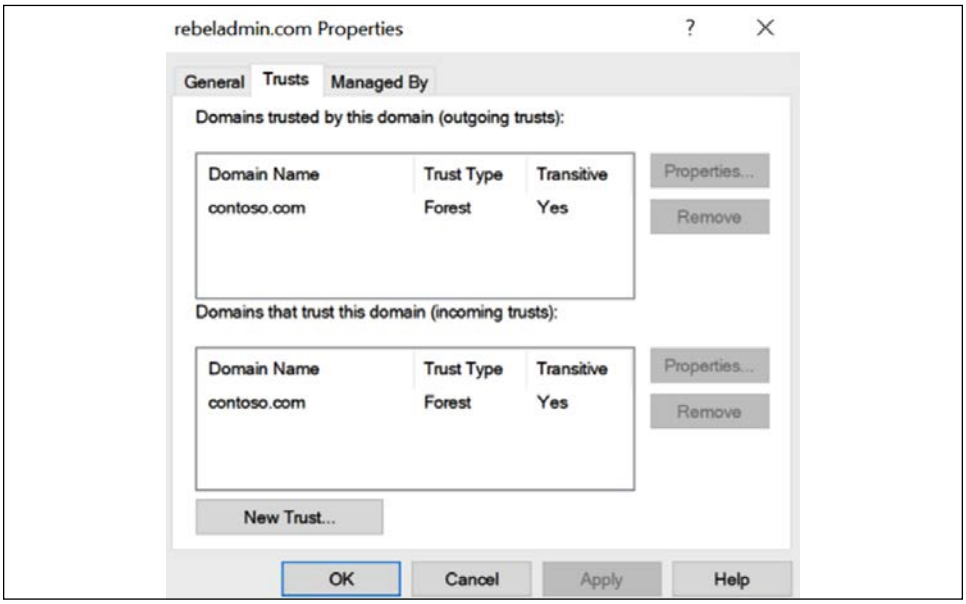


Figure 12.11: Trust details – rebeladmin.com

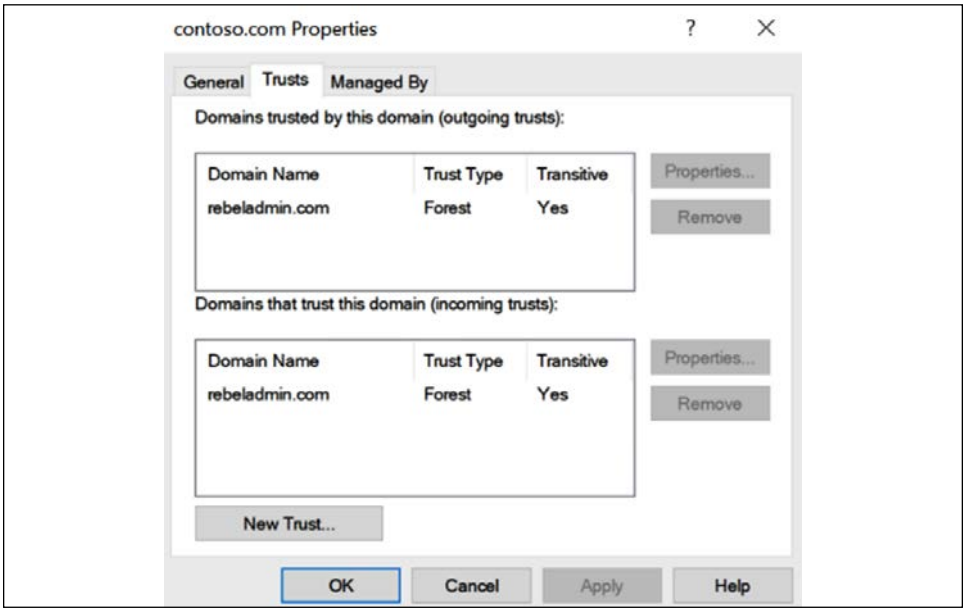


Figure 12.12: Trust details – contoso.com

Now we have the trust in place. The next step is to do some testing to verify the trust.

Testing

When we have the trust in place, we should be able to query users in the remote domain. To test that, I will log in to DC01.rebeladmin.com and am going to query users in one of the contoso.com domain **Organizational Units** (**OUs**).

```
Get-ADUser -Server CON-DC01.contoso.com -Filter * -SearchBase
"OU=Test,DC=CONTOSO,DC=COM"
```

In the preceding command, I am querying users under the **Test** OU in the contoso.com domain. As expected, I was able to query users from the contoso.com domain.

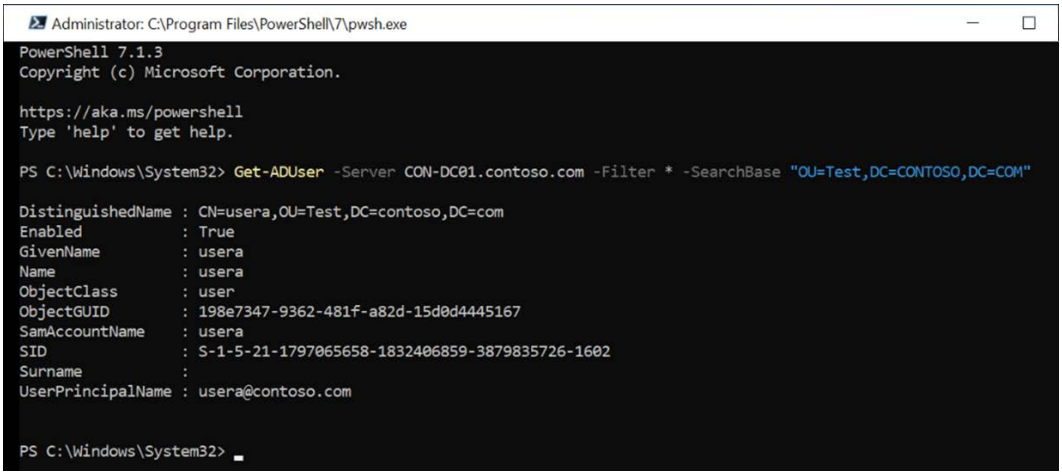


Figure 12.13: Active Directory User Query – contoso.com

Let's try the same from the contoso.com domain.

```
Get-ADUser -Server DC01.rebeladmin.com -Filter * -SearchBase
"OU=Sales,DC=rebeladmin,DC=com"
```

In the above command, I am querying users under the **Sales** OU in the rebeladmin.com domain. As expected, I was able to list the users.

Figure 12.14: Active Directory User Query – rebeladmin.com

I can also query contoso.com from DC01.rebeladmin.com.

```
Get-ADDomain contoso.com
```

```
PS C:\Windows\System32> hostname
DC01
PS C:\Windows\System32> Get-ADDomain contoso.com

AllowedDNSSuffixes                : {}
ChildDomains                      : {}
ComputersContainer                : CN=Computers,DC=contoso,DC=com
DeletedObjectsContainer           : CN=Deleted Objects,DC=contoso,DC=com
DistinguishedName                 : DC=contoso,DC=com
DNSRoot                           : contoso.com
DomainControllersContainer        : OU=Domain Controllers,DC=contoso,DC=com
DomainMode                        : Windows2016Domain
DomainSID                         : S-1-5-21-1797065658-1832406859-3879835726
ForeignSecurityPrincipalsContainer : CN=ForeignSecurityPrincipals,DC=contoso,DC=com
Forest                            : contoso.com
InfrastructureMaster              : CON-DC01.contoso.com
LastLogonReplicationInterval      :
LinkedGroupPolicyObjects          : {CN={31B2F340-016D-11D2-945F-00C04FB984F9},CN=Policies,CN=System,DC=contoso,DC=com}
LostAndFoundContainer             : CN=LostAndFound,DC=contoso,DC=com
ManagedBy                         :
Name                              : contoso
NetBIOSName                       : CONTOSO
ObjectClass                       : domainDNS
ObjectGUID                        : bcb8dfca-26c5-4a54-bf16-37ef2e282473
ParentDomain                      :
PDCEmulator                       : CON-DC01.contoso.com
PublicKeyRequiredPasswordRolling  : True
QuotasContainer                   : CN=NTDS Quotas,DC=contoso,DC=com
ReadOnlyReplicaDirectoryServers   : {}
ReplicaDirectoryServers           : {CON-DC01.contoso.com}
RIDMaster                         : CON-DC01.contoso.com
SubordinateReferences             : {DC=ForestDnsZones,DC=contoso,DC=com, DC=DomainDnsZones,DC=contoso,DC=com,
                                    CN=Configuration,DC=contoso,DC=com}
SystemsContainer                  : CN=System,DC=contoso,DC=com
UsersContainer                    : CN=Users,DC=contoso,DC=com
```

Figure 12.15: Active Directory Domain Query – contoso.com

As we can see, the two-way Active Directory forest trust is working as expected. In the next section, we are going to look into another form of domain controller.

RODCs

RODC is a great role introduced with Windows Server 2008. RODCs can be used in locations where we cannot guarantee physical security and regular maintenance. Throughout this chapter, we have discussed possible scenarios where we have required a domain controller in a remote site. When considering a domain controller in a remote site, the link between sites is not the only thing we need to consider. A domain controller, by default, will be aware of any changes in the Active Directory structure. Once an update triggers, it updates its own copy of the Active Directory database. This `ntds.dit` file contains everything about the Active Directory infrastructure, including the data about the user objects. If this file falls into the wrong hands, they could retrieve data related to identities and compromise the identity infrastructure.

When considering information security, physical security is also important. That's why data centers have all sorts of security standards. So, when deploying a domain controller in a remote site, we also need to take physical security into consideration. If you have a requirement for a domain controller in a remote site, but you cannot confirm its security, then an RODC is the answer. An RODC does not store any passwords in its database. All the authentication requests against an object will be processed by the closest writable domain controller. Therefore, even if someone manages to get a copy of the database, they will not be able to do much.

Another advantage of RODCs is that they only do one-way replications. When considering remote sites, you also need to consider how systems will be maintained. Not every organization can afford IT teams for remote offices. Most maintenance tasks can still be carried out remotely, but there are certain situations where you will need to delegate some permissions to people in remote sites to manage domain controllers in their location. Most of the time, these people are less experienced in IT, so any simple mistakes made by them can be replicated to other domain controllers and make a mess. RODCs' one-way replication will prevent this and no change will be replicated over to other domain controllers.

By default, RODCs do not save any passwords (except RODC objects) for Active Directory objects. Every time an authentication happens, the RODC needs to retrieve the data from the closest domain controller. Using a **Password Replication Policy** (**PRP**), we can allow certain passwords for objects to be cached. If the connection between a remote site and the closest domain controller is interrupted, the RODC will be able to process the request. One thing to remember is, in order to process a Kerberos request, RODC needs to cache the password for the user object as well as the computer object.

Azure AD Connect does not support RODCs. The domain controller used by Azure AD must be writable as Azure AD Connect does not know how to work with write redirects.

The RODC deployment process involves the following stages. In this process, we can use a preselected account and promote the RODC instead of using a Domain Admin or an Enterprise Admin account:

1. Set up a computer account for the RODC domain controller
2. Attach that account to the RODC during the promotion process

In order to create an RODC computer account, we can use the `Add-ADDSReadOnlyDomainControllerAccount` cmdlet:

```
Add-ADDSReadOnlyDomainControllerAccount -DomainControllerAccountName
REBEL-RODC-01 -DomainName rebeladmin.com
-DelegatedAdministratorAccountName "rebeladmindfrancis" -SiteName
LondonSite
```

The preceding command will create the RODC domain controller account for `REBEL-RODC-01`. The domain name is defined using `-DomainName`, and `-DelegatedAdministratorAccountName` defines which account to delegate the RODC installation to. The new RODC will be placed in `LondonSite`.

Now, we can see the newly added object under the Active Directory domain controllers:

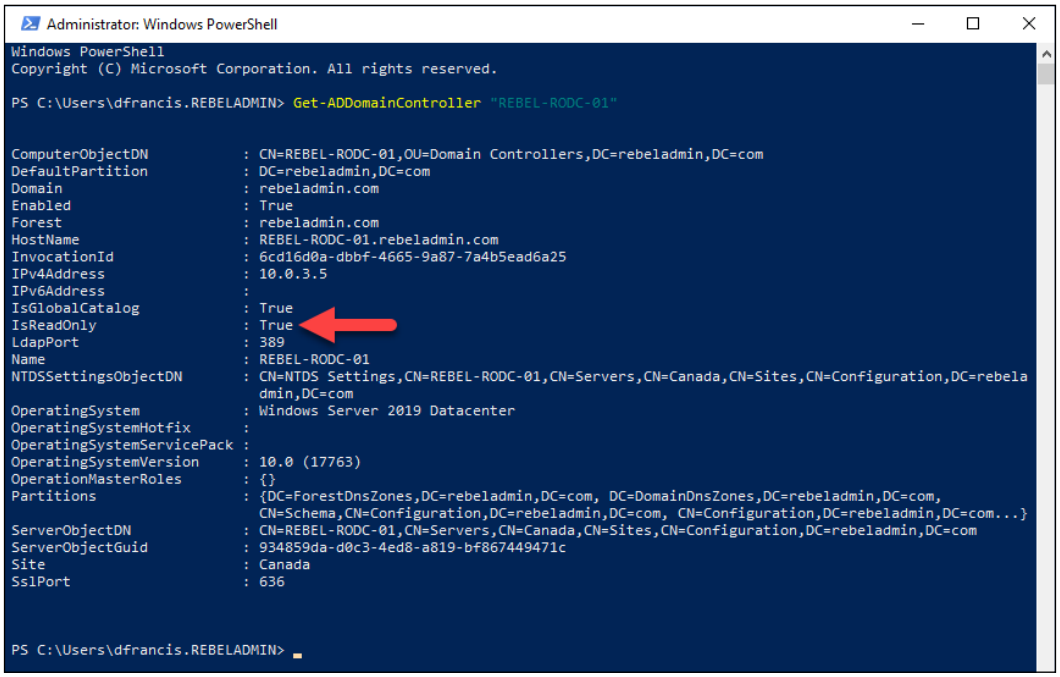


Figure 12.16: RODC Settings

Now, we have things ready for the new RODC and the next step is to install the relevant role service:

```
Install-WindowsFeature –Name AD-Domain-Services -IncludeManagementTools
```

The preceding command installs the AD DS role in the RODC. Once it's completed, we can promote it using the following command:

```
Import-Module ADDSDeployment
Install-ADDSDomainController `
-Credential (Get-Credential) `
-CriticalReplicationOnly:$false `
-DatabasePath "C:\Windows\NTDS" `
-DomainName "rebeladmin.com" `
```

```
-LogPath "C:\Windows\NTDS" `
-ReplicationSourceDC "REBEL-PDC-01.rebeladmin.com" `
-SYSVOLPath "C:\Windows\SYSVOL" `
-UseExistingAccount:$true `
-Norebootoncompletion:$false
-Force:$true
```

Once this is executed, the system will ask for the user account, and we need to input the user account information, which was delegated for the RODC deployment.

This will complete the promoting process and the next step is to look into PRP.

The default policy is already in place, and we can view the `Allowed` and `Denied` lists using the following command:

```
Get-ADDomainControllerPasswordReplicationPolicy -Identity REBEL-RODC-01
-Allowed
```

The preceding command lists the `Allowed` objects for password caching. By default, a security group called Allowed RODC Password Replication Group is allowed for the replication. This doesn't contain any members by default. If we need caching, we can add an object to the same group:

```
Get-ADDomainControllerPasswordReplicationPolicy -Identity REBEL-RODC-01
-Denied
```

The preceding command lists the `Denied` objects for password caching. By default, the following security groups are in the `Denied` list:

- Denied RODC Password Replication Group
- Account Operators
- Server Operators
- Backup Operators
- Administrators

These are high-privileged accounts in the Active Directory infrastructure; these should not be cached at all. By adding objects to Denied RODC Password Replication Group, we can simply block the replication.

Apart from the use of predefined security groups, we can add objects to the `Allowed` and `Denied` lists using the `Add-ADDomainControllerPasswordReplicationPolicy` cmdlet:

```
Add-ADDomainControllerPasswordReplicationPolicy -Identity REBEL-RODC-01
-AllowedList "user1"
```

The preceding command will add the user object named `user1` to the `Allowed` list.

The following command will add the user object named `user2` to the `Denied` list:

```
Add-ADDomainControllerPasswordReplicationPolicy -Identity REBEL-RODC-01
-DeniedList "user2"
```

To improve the security further, it is recommended to install an RODC in the Windows Core operating system.

Active Directory database maintenance

Active Directory maintains a multi-master database to store schema information, configuration information, and domain information. Normally, when we say *database*, the first thing that comes to our mind is software such as Microsoft SQL, MySQL, or Oracle. But here, it's quite different. Active Directory databases use the **Extensible Storage Engine** (**ESE**), which is an **Indexed and Sequential Access Method** (**ISAM**) technology.

Here, a single system works as the client and server. It uses record-oriented database architecture, which provides extremely fast access to records. The ESE indexes the data in the database file, which can grow up to 16 terabytes and hold over 2 billion records. Typically, the ESE is used for applications that require fast and structured data storage. The ESE is used for many other Microsoft applications, including Microsoft Exchange, DHCP, and FRS.

As the database creation process is part of the domain controller installation process, it creates the database under `C:\Windows\NTDS` unless we select a custom path. It is recommended to use a separate partition/disk, to increase the database performance as well as the data protection:

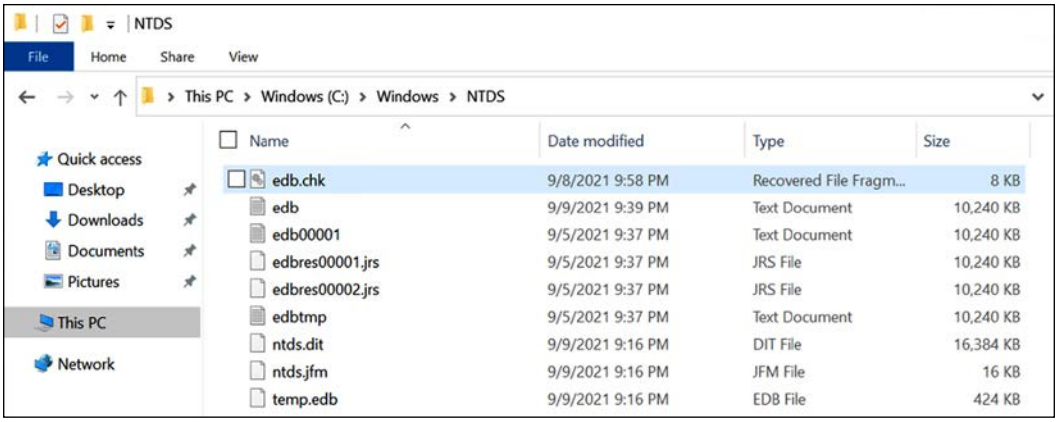


Figure 12.17: NTDS Folder

In this folder, we can see a few different files. Out of those, the following files are important:

- ntds.dit
- edb.log
- edb.chk
- temp.edb

Now let's go and see what these files do.

The ntds.dit file

This is the actual Active Directory database file. This database contains three main tables. The **schema** table includes data regarding the object classes, attributes, and the relationship between them. The **link** table includes data about values referring to another object. Group membership details are a good example of this. The **data** table includes all the data about users, groups, and any other data integrated with Active Directory. In the table, rows represent the objects and columns represent the attributes.

The edb.log file

Here, we can see that a few log files start with `edb*`, all of which are 10 MB or less in size. This is the transaction log maintained by the system to store the directory transaction before writing data into the database file.

The edb.chk file

This file is responsible for keeping track of the data transaction committed into the database from log files (`edb*.log`).

The temp.edb file

This file is used during the Active Directory database maintenance to hold data and, also, to store information about Active Directory data transactions that are in progress.

Every domain controller in the Active Directory environment needs to be aware of the changes made in each domain controller. When this happens, you may think that the database is being synced. But it is not the database; only the changes are being synced. Therefore, each domain controller in the domain will not be the same size.

Most database systems have their own automatic data grooming techniques to maintain the efficiency of the system. This also gives administrators a chance to perform custom maintenance tasks and granular maintenance. An Active Directory database is a self-maintained system. It does not require daily maintenance. However, there are some situations where it requires manual intervention:

- If the default database partition is running out of space or notices a potential hardware failure
- To free up unused space in an Active Directory database after mass object deletion

If required, we can move the Active Directory database from its default location. To do that, we can use a command-line tool called `ntdsutil`. When moving the database files, it is also recommended to move the log files. The minimum space requirement for the database file is 500 MB, or the database file size along with 20% of the database file size (whichever is greater). The log file space requirement is also the same.

The database and log files cannot be moved while AD DS is running. Therefore, the first step of the action is to stop the service:

```
net stop ntds
```

This will also stop the associated services, including KDC, DNS, and DFS.

The Active Directory database and log files cannot be moved to a non-existent folder. So before we move the files, the destination folder needs to be created.

In my demonstration, I will move it to a folder called `ADDB` in a different partition:

```
ntdsutil
activate instance ntds
files
move db to E:\ADDB
move logs to E:\ADDB
integrityquit
quit
```

In the preceding command, `ntdsutil` initiates the utility, `move db to E:ADDB` moves the database files to the new location, and `move logs to E:ADDB` moves the log files to the new directory. The `integrity` part verifies the integrity of the database and logs files in the new location.

Once it's completed, we need to start AD DS using the following command:

```
net start ntds
```

If you are regularly backing up Active Directory, it is recommended that you make a full backup of Active Directory as soon as the database migration process is completed. The previous backup that was taken will not be valid anymore.

Offline defragmentation

In any database system, the data will be added, modified, and deleted as it goes. When new data is added, it requires *new* space inside the database. When the data is removed, it *releases* space to the database. When the database is modified, it either needs new space or releases space. In the Active Directory database, once an object has been deleted, it releases the space it used to the database, and not to the filesystem. Then this released free space will be used for new objects. This process is called **online defragmentation** because it does not need to stop the Active Directory services. By default, it runs every 12 hours.

However, when a large number of objects or a global catalog server are removed, it is worth releasing this free space to the filesystem. In order to do that, we need to perform **offline defragmentation**. To do so, we need to stop the Active Directory services.

Once the service stops (`net stop ntds`), we can run the defragmentation using the following commands:

```
Ntdsutil
activate instance ntds
files
compact to E:\CompactDB
quit
quit
```

In the preceding process, we need a temporary folder location in which to save the compact `ntds.dit` file. In my demonstration, I created a folder named `E:\CompactDB` for this.

Once this is completed, the compact database should be copied to the original `ntds.dit` location. This can be done by using the following command:

```
copy "E:\CompactDB\ntds.dit" "E:\ADDB\ntds.dit"
```

After that, we also need to delete the old log file:

```
del E:\ADDB\*.log
```

After that, we can start AD DS using `net start ntds`.

This completes the two scenarios (database path change and defragmentation) where we will need to use manual intervention for Active Directory database maintenance.

Active Directory Backup and Recovery

Active Directory domain controllers are the main components responsible for the organization's identity infrastructure. Failure of the domain controllers or the services will impact the entire identity infrastructure. Therefore, as with any other critical system of a business, the Active Directory server's high availability is crucial. There are two types of disasters related to Active Directory domain controllers that can occur.

The first type of disaster is when there is a complete system crash due to faulty hardware. Apart from the Active Directory backup, maintaining multiple domain controllers helps organizations to recover from such situations. If it's not the **flexible single master operation** (**FSMO**) role holder, we can forcefully remove the crashed domain controller's related records and introduce a new domain controller. If it's the FSMO role holder, we can *seize* the FSMO roles and make them available from any other live domain controller. On the other hand, most workloads operate in a virtualized environment today, including domain controllers. These virtualized environments usually have solutions in place to recover from failures. As an example, virtualization solutions may use a clustered environment or advanced recovery solution such as Azure Site Recovery. Therefore, in modern infrastructures, I rarely see anyone who has had to restore a domain controller from a backup.

The second type of disaster is due to the deletion of, or configuration alterations in, Active Directory objects. Restoring a system from backup is not always a *no-impact* disaster recovery. It can take *time* to recover your system to a working condition. The recovery process can be followed by some data loss or operation impacts due to the time taken. If you want to recover an object you deleted in Active Directory, it doesn't make sense to restore the whole domain controller itself from a backup. Therefore, Microsoft uses different tools and methodologies to recover from both situations. In the following sections, we are going to look into these in detail.

Preventing the accidental deletion of objects

With AD DS 2008, Microsoft introduced a small but important feature to prevent accidental Active Directory object deletion. This is not a solution to recover from disasters, but it is a solution to prevent disasters. In every Active Directory object, under the **Object** tab, there is a small checkbox to enable this feature. This can be enabled when we create objects using PowerShell.

Even if we're not using PowerShell, this can still be enabled using the **Object** properties window at any time. When creating an OU using the GUI, this feature will be enabled by default.

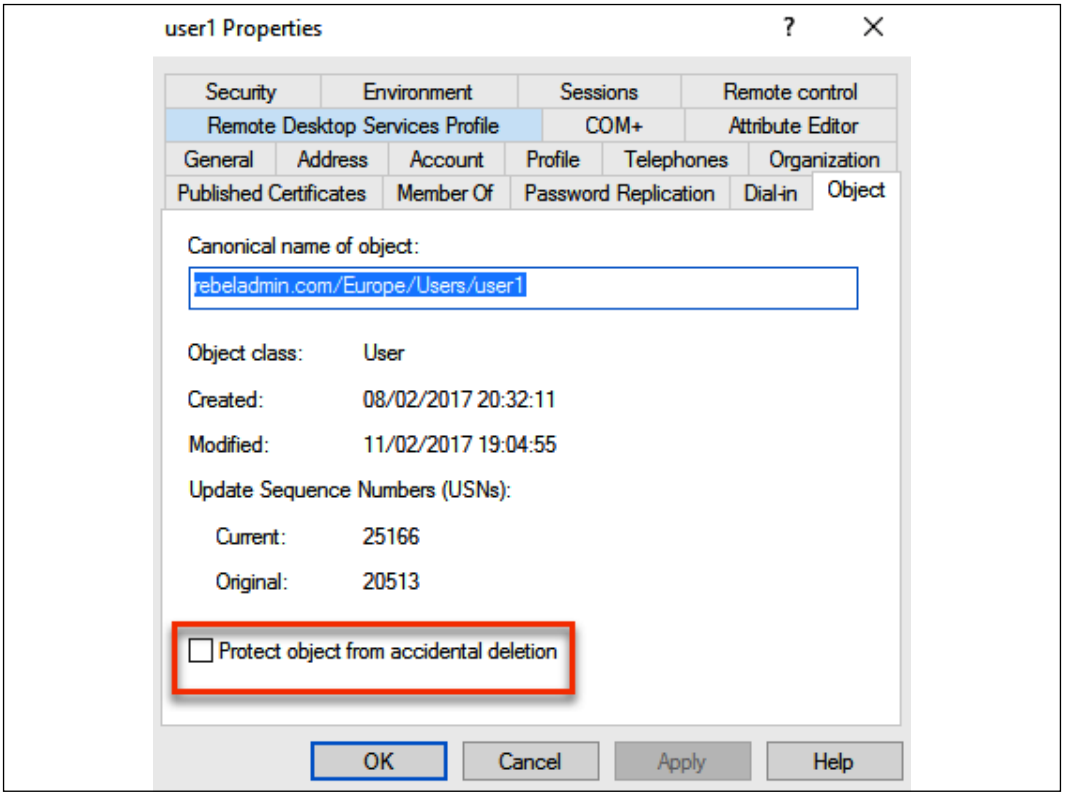


Figure 12.18: Protect objects from accidental deletion

When this option is enabled, it will not allow you to delete the object unless you disable this option:

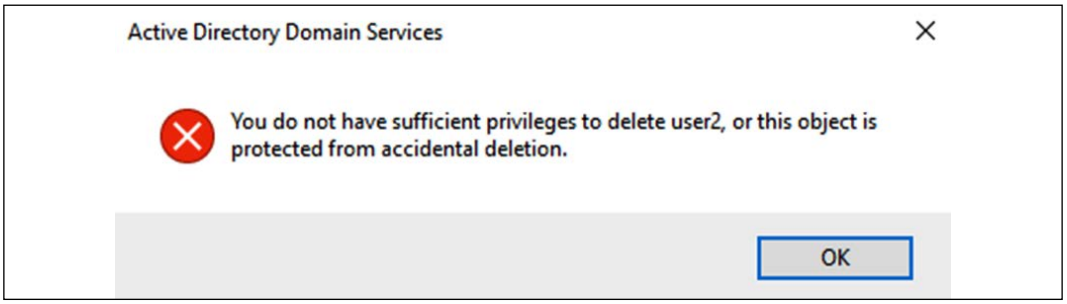


Figure 12.19: Error Message – Protected Object

In PowerShell, this can be done using the `-ProtectedFromAccidentalDeletion $true` parameter.

Active Directory Recycle Bin

The most common Active Directory-related disasters are due to accidentally deleting objects. When an object is deleted from Active Directory, it is not permanently deleted. As soon as an object is deleted, it will set the `isDeleted` attribute value to `True` and move the object under `CN=Deleted Objects`:

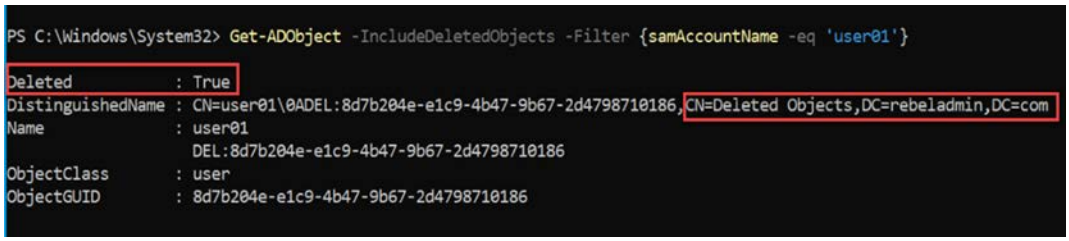


Figure 12.20: Deleted Active Directory Object

Then, it stays there until the system reaches the *tombstone lifetime* value. By default, this is 180 days, and it can be changed if required. As soon as the object passes the tombstone lifetime value, it is available for permanent deletion. When we discussed the Active Directory database in the *Active Directory database maintenance* section, we discussed online defragmentation. The process uses the garbage collector service to remove the deleted objects from the Active Directory database and release that space to the database. This service runs every 12 hours. Once the deleted object exceeds the tombstone lifetime value, it will be permanently removed in the next garbage collector service cycle. The problem with this is that during the tombstone process, most of the object values are stripped off. So, even if you were able to recover objects, values would need to be re-entered.

With Windows Server 2008 R2, Microsoft introduced the *Active Directory Recycle Bin* feature. When this feature is enabled, once the object is deleted, it still sets the `isDeleted` object value to `True` and moves the object under `CN=Deleted Object`. But, instead of the tombstone lifetime, it's now controlled by **Deleted Object Lifetime** (**DOL**) values. Object attributes will remain the same at this stage, and they are easily recoverable. By default, the DOL value is equal to the tombstone lifetime. This value can be changed by modifying the `msDS-deletedObjectLifetime` attribute value. Once it's exceeded the DOL, it is moved into the `Recycled` state and the `isRecycled` attribute value is set to `True`. In this state, it cannot be recovered, and it will be in this state until the tombstone lifetime value is exceeded. After it reaches the value, it will be permanently deleted from Active Directory.

The AD Recycle Bin feature requires a minimum of a Windows Server 2008 R2 domain and a forest functional level. Once this feature is enabled, it cannot be disabled.

This feature can be enabled using the following command:

```
Enable-ADOptionalFeature 'Recycle Bin Feature' -Scope
ForestOrConfigurationSet -Target rebeladmin.com
```

In the preceding command, `-Target` can be changed with your domain name.

Once `Recycle Bin Feature` is enabled, we can revive the objects that have been deleted using the following command:

```
Get-ADObject -filter 'isdeleted -eq $true' -includeDeletedObjects
```

The preceding command searches for the objects where the `isdeleted` attributes are set to `true`.

Now, we know the deleted object and it can be restored using the following command:

```
Get-ADObject -Filter 'samaccountname -eq "dfrancis"'
-IncludeDeletedObjects | Restore-ADObject
```

The preceding command restores the user object, `dfrancis`.

Active Directory snapshots

If you work with virtual servers, you know that *snapshots* are important for a fast recovery process. Snapshots allow us to revert the system to a previous working state with minimum impact.

It is not recommended to take snapshots and restore domain controllers using this method, as it will create integrity issues with other existing domain controllers and their data.

With Windows Server 2008, Microsoft introduced the Active Directory snapshot feature, which takes a snapshot of an Active Directory database at a given time. Later, it can be used to compare object value changes and export and import objects that have been deleted or modified. Do not mistake this for a typical snapshot.

This happens inside Active Directory, and we cannot use it to completely recover a domain controller.

It allows us to mount a snapshot while the existing AD DS configuration is running. However, it does not allow us to move or copy objects between snapshots and a working AD DS instance.

We can create the AD DS snapshot using `ntdsutil`. In order to run this, we need to have domain administrator privileges:

```
Ntdsutil
Snapshot
activate instance ntds
create
quit
quit
```

Now, we have a snapshot, and at a later time, it can be mounted. To mount it, we need to use the following command:

```
Ntdsutil
Snapshot
activate instance ntds
list all
mount 1
quit
quit
```

The preceding command mounts a snapshot called `1` from the list, which is listed under the given mount points.

The next step is to mount the snapshot, which can be done using the following command:

```
dsamain –dbpath C:$SNAP_201703152333_VOLUMEE$ADDBntds.dit –ldapport
10000
```

In the preceding command, `-dbpath` defines the AD DS database path, and `-ldapport` defines the port used for the snapshot. It can be any available TCP port.

Once the snapshot is mounted, we can connect to it using the server's name and the LDAP port, `10000`:

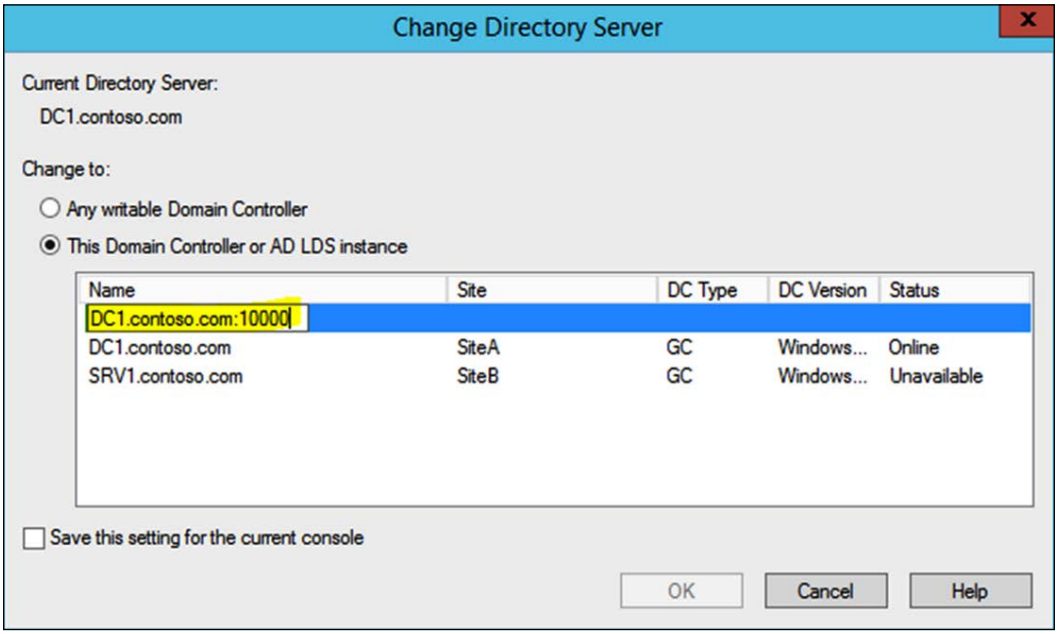


Figure 12.21: Connecting to Snapshot

Once the work is finished, it needs to be unmounted as well. To do that, we can use the following command:

```
Ntdsutil
Snapshot
activate instance ntds
list all
unmount 1
quit
quit
```

If you need to move an object from a snapshot, first you need to export the object, and then import it over.

Active Directory system state backup

An Active Directory system state backup is required in order to restore Active Directory in the event of a disaster where the database cannot be recovered using the previously explained object-level recovery options. Windows backup is supported for performing system state backups. Also, there are many third-party backup tools that use similar technology to do the Active Directory system state backup.

In a system state backup, the following files are included:

- AD DS database file (`ntds.dit`)
- The `SYSVOL` folder and its files
- The certificate store
- User profiles
- **Internet Information Services** (**IIS**) metabase
- Boot files
- **Dynamic-link library** (**DLL**) cache folder
- Registry info
- COM+ and WMI info
- Cluster service info
- Windows Resource Protection system files

After Windows Server 2008, the system state backup also included Windows system files, so the system state backup is larger than Windows Server 2003 system state backups. It is recommended that you take a system state backup for every domain controller.

The first step to proceed with configuration is to install the Windows backup feature in the Active Directory server:

```
Install-WindowsFeature -Name Windows-Server-Backup –
IncludeAllSubFeature
```

Next, let's create a backup policy using the following command:

```
$BKPolicy = New-WBPolicy
```

Then, let's go ahead and add a system state to the policy:

```
Add-WBSystemState -Policy $BKPolicy
```

It also needs the backup volume path:

```
$Bkpath = New-WBBackupTarget -VolumePath "F:"
```

Now, we need to map the policy with the path:

```
Add-WBBackupTarget -Policy $BKPolicy -Target $Bkpath
```

Finally, we can run the backup using the following command:

```
Start-WBBackup -Policy $BKPolicy
```

Active Directory recovery from system state backup

When the system needs to recover from the system state backup, it needs to be done via **Directory Services Restore Mode** (**DSRM**).

The first system needs to be rebooted; press the *F8* key and select **Directory Services Restore Mode**.

Once it's loaded in safe mode, we can use the following commands:

```
$ADBackup = Get-WBBackupSet | select -Last 1
Start-WBSystemStateRecovery -BackupSet $ADBackup
```

This will restore the most recent backup the system has taken.

If you are using Active Directory backup software other than Windows, the recovery options will be different from the aforementioned options and you should refer to the vendor guidelines.

Summary

We started this chapter by looking into Active Directory trusts, which enable collaboration between organizations. Then, we moved on to RODCs and looked into their features and deployment scenarios. Later, we looked into Active Directory database maintenance, which included different tools and techniques used to optimize Active Directory database performance. Last but not least, we looked at Active Directory recovery options.

In the next chapter, we are going to look into another important Active Directory role service: AD CS.

13
Active Directory Certificate Services

The two-man rule in security is used to secure high-valued assets and operations. As an example, many banks provide safe deposit box facilities. People can rent safe deposit boxes to store valuable assets. Most of these safe deposit boxes are designed to support a two-man rule. This means that each safe deposit box has two locks. One key to the lock is held by the bank, and another key for a second lock is issued to the customer. To open it, customers and bank agents need to use their keys at the same time. When a customer shows up at the bank, there is a process to follow to get access to safe deposit boxes. Banks will verify the customer's *identity* first. They will ask for a passport or driving license to verify the customer's identity. Following successful verification, the bank will assign a member of staff to go with the customer and open the box using the bank's and the customer's keys. The end goal of these layers of security is to verify that the customer is the person they *claim* to be in order to allow access to the high-valued assets in the safe deposit box.

The **public key infrastructure** (**PKI**) works in a similar way. The PKI is responsible for verifying objects and services by using digital certificates. When we apply for visas or jobs, sometimes we are asked to verify our identity using police certificates. We may have already provided a copy of our passport and identity card with the application forms. However, the police are a well-known authority that anyone can trust. Therefore, a police certificate that verifies our identity will further confirm that we are the person we claim to be. The police department is responsible for the certificate they issued for us. Before providing certificates, it's their responsibility to verify our identity using their own methods.

Modern businesses are increasingly using PKI to counter modern infrastructure threats. As an example, people use digital certificates to secure online transactions, to get access to corporate Wi-Fi, to authenticate to different applications/services, to **secure network logins and network traffic** (**SSL VPN**), and so on. **Active Directory Certificate Services** (**AD CS**) allows organizations to set up and maintain their own PKI in their infrastructure to create, manage, store, renew, and revoke digital certificates. In this chapter, we are going to look at the following topics:

- How does PKI work?
- How to design your PKI
- Different PKI deployment models
- How to set up a two-tier PKI?
- Certificate Authority Migration from Windows Server 2008 R2
- Certificate Authority Disaster Recovery

Before we look at AD CS role configuration, it is important to know what PKI is and how it works. Without this knowledge, it is really hard to troubleshoot CA issues. So, let's get on and set out the foundation.

PKI in action

In general, we know "encryption" is *more secure* and that it requires **secure sockets layer** (**SSL**) certificates. But the question is, do we really know what the role of a certificate is and how this encryption and decryption works? It is very important to know how these all work, as it makes deployment and management of PKI easy.

Symmetric keys versus asymmetric keys

There are two types of cryptographic methods used to encrypt data:

- **Symmetric keys**: Symmetric methods work in exactly the same way as your door lock works. You have one key to lock or open. This method is also known as the shared **secret method** or **private key method**. **Virtual private network** (**VPN**) connections and backup software are good examples of systems that use symmetric keys to encrypt data.
- **Asymmetric keys**: This method, on the other hand, uses a **key pair** to perform the encryption and decryption. It includes two keys: one is a **public key**, and the other one is a **private key**. Public keys are always distributed to the public and anyone can have them.

Private keys are unique to the object in question and are not distributed to others. Any message encrypted using a public key can be decrypted only using its private key. Any message encrypted using a private key can be decrypted only using a public key. PKI uses the asymmetric key method for digital encryption and digital signatures.

Let's move on and explore the asymmetric key method in more detail and how it works with encryption and decryption.

Digital encryption

Digital encryption means that a data transfer between two parties will be encrypted, and the sender will ensure that the transferred data can only be opened by the intended recipient. Even if an unauthorized party gains access to that encrypted data, they will not be able to decrypt the data. The best way to explain it is through an example:

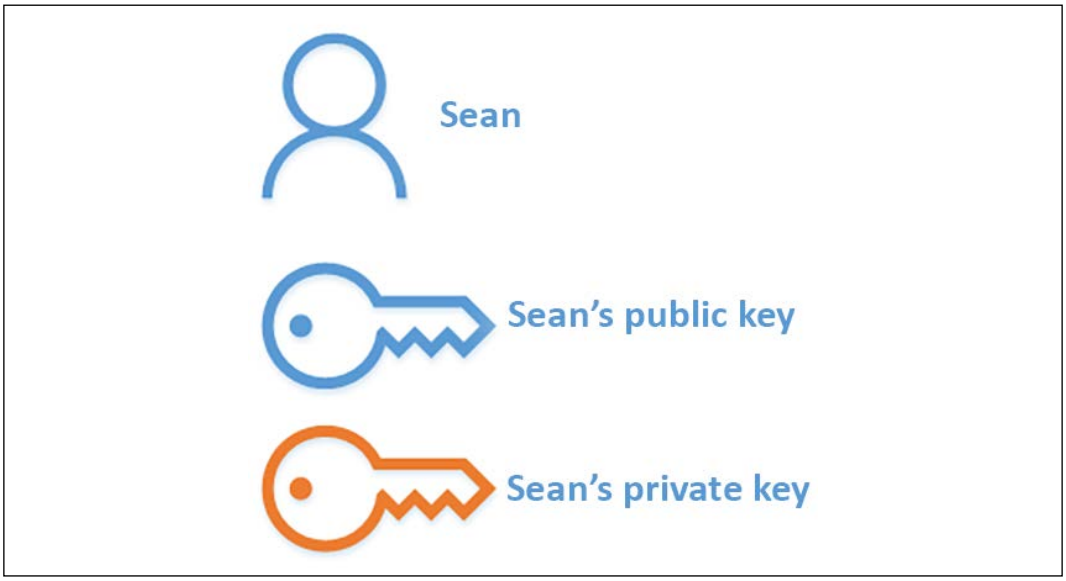


Figure 13.1: A user's key pair

We have an employee in the organization called **Sean**. The organization already has PKI in place. In the PKI environment, the user owns two keys: a public key and a private key. **Sean** can use these keys for encryption and the signature process. Sean is expecting some confidential data from the company account manager, **Chris**. Sean doesn't want anyone else to access this confidential data.

The best way to do this is to encrypt the data that is going to be sent from Chris to Sean:

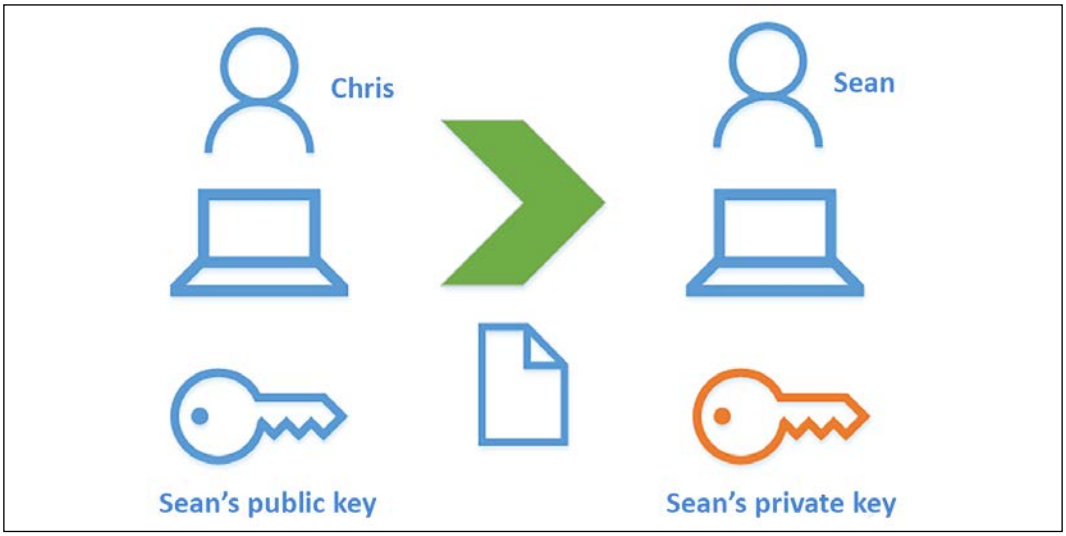


Figure 13.2: Digital encryption example

To encrypt the data by using the asymmetric key method, **Sean** sends his public key to **Chris**. There is no security issue with providing the public key as we need both keys to encrypt or decrypt. Then, **Chris** uses this public key to encrypt the data that will be sent over to **Sean**. This encrypted data can only be opened using **Sean**'s private key. He is the only one who has this private key. Once Sean receives the encrypted data, he decrypts it using his private key and processes the data.

Digital signatures

A digital signature verifies the authenticity of a service or data. It is similar to signing a document to prove its authenticity. As an example, before buying anything from a website, we can check its digital certificate and verify the authenticity of the website and confirm that it's not a phishing website. Let's look into this further with a use case. In the previous scenario, **Sean** successfully decrypted the data he received from **Chris**. Now, **Sean** wants to send some confidential data back to **Chris**. It can be encrypted using the same method as was used with Chris's public key. But the issue is that **Chris** is not part of the PKI setup, and he does not have a key pair. The only thing **Chris** needs to verify is the fact that the sender is legitimate and that he's the same user that he claims to be. If **Sean** can certify this using a digital signature, and if **Chris** can verify it, the problem is solved:

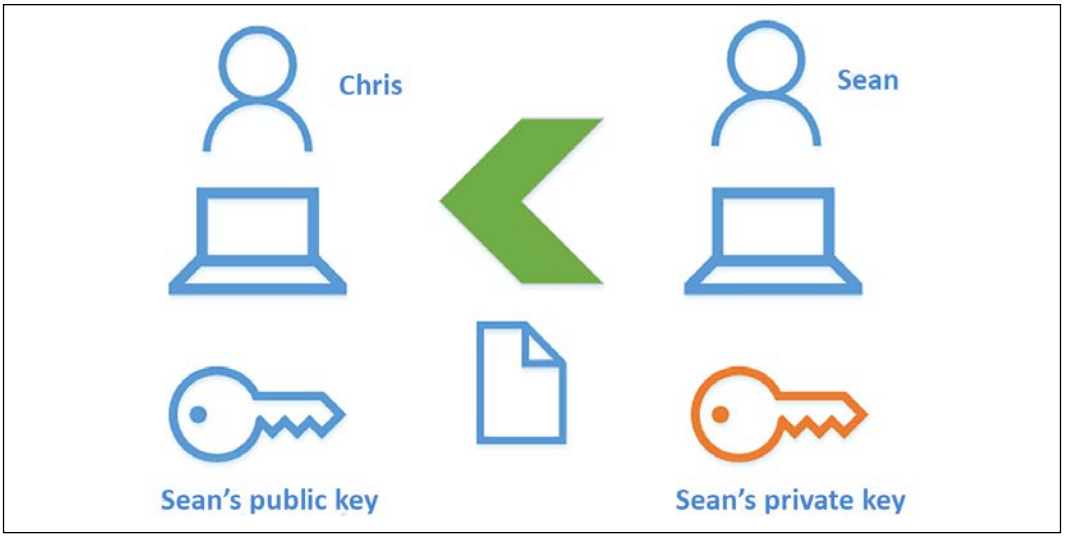


Figure 13.3: Digital signature example

In this example, **Sean** encrypts the data using his private key instead of **Chris's** public key. The only key with which the data can be decrypted is **Sean's** public key. **Chris** already has the public key. When **Chris** receives the data, he decrypts it using **Sean's public key** and this confirms that the sender is indeed **Sean**.

Signing, encryption, and decryption

In the previous two scenarios, I have explained how digital encryption and digital signatures work with PKI. But these scenarios can be combined to provide encryption and signing at the same time. To do that, we use two additional techniques:

- **Symmetric keys**: A one-time symmetric key will be used for the message encryption process, as it is faster than asymmetric key encryption algorithms. This key needs to be available for the receiver, but to improve security, it will still be encrypted using the receiver's public key.
- **Hashing**: During the signing process, the system will generate a one-way hash value to represent the original data. Even if someone manages to get that hash value, it will not be possible to reverse engineer the hash to get the original data. If any modification is done to the data, the hash value will be changed, and the receiver will know this straight away. These hashing algorithms are faster than encryption algorithms, and also, the hashed data will be smaller than the actual data values.

Let's look into this with the aid of a scenario. Let's assume that we have two employees, **Simran** and **Brian**, and both are using a PKI setup. Both have their private and public keys assigned:

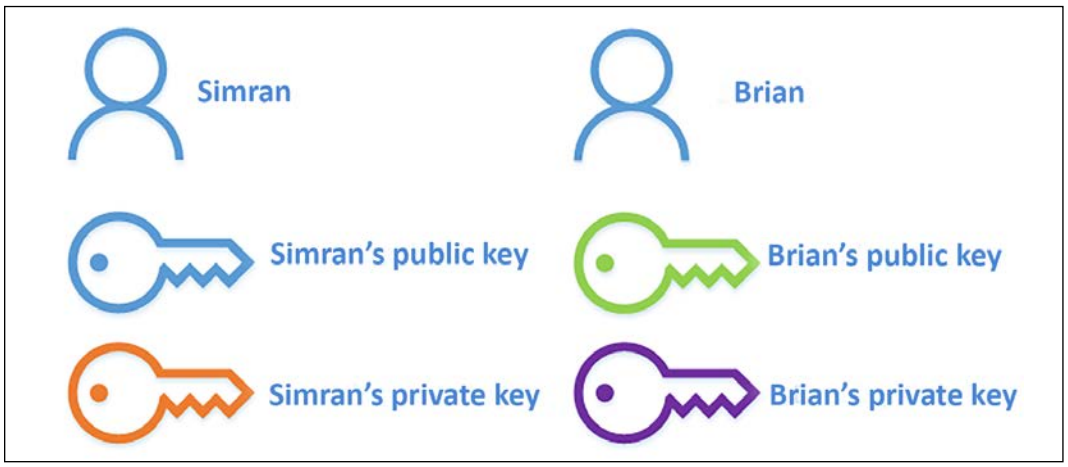


Figure 13.4: An example of key usage

Simran wants to send an encrypted and signed data segment to **Brian**. This process can be divided into two stages: **data signing** and **data encryption**. The data will go through both stages before being sent to **Brian**:

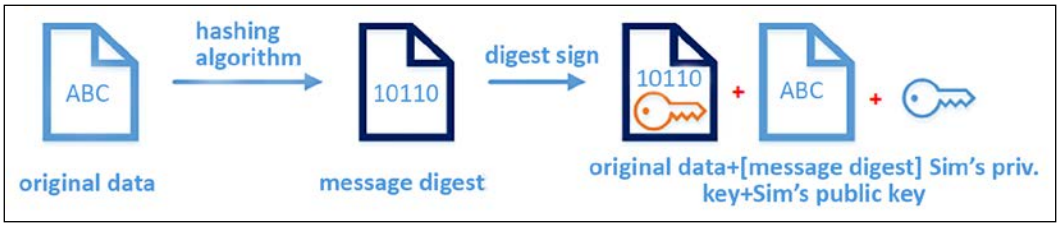


Figure 13.5: Data signing

The first stage of the process is the signing of the data segment. The system receives the data from **Simran**, and the first task is to generate a **message digest** using a **hashing algorithm**. This will ensure data integrity; if data is altered, the receiver can easily identify this during the decryption process. This is a one-way process. Once a **message digest** is generated, the **message digest** will be encrypted using **Simran's** private key to digitally sign it. It will also include **Simran's** public key, so **Brian** will be able to decrypt and verify the authenticity of the message. Once the encryption process is concluded, this additional data will be attached to the **original data**.

This process will ensure that data was not altered and was sent from the expected sender:

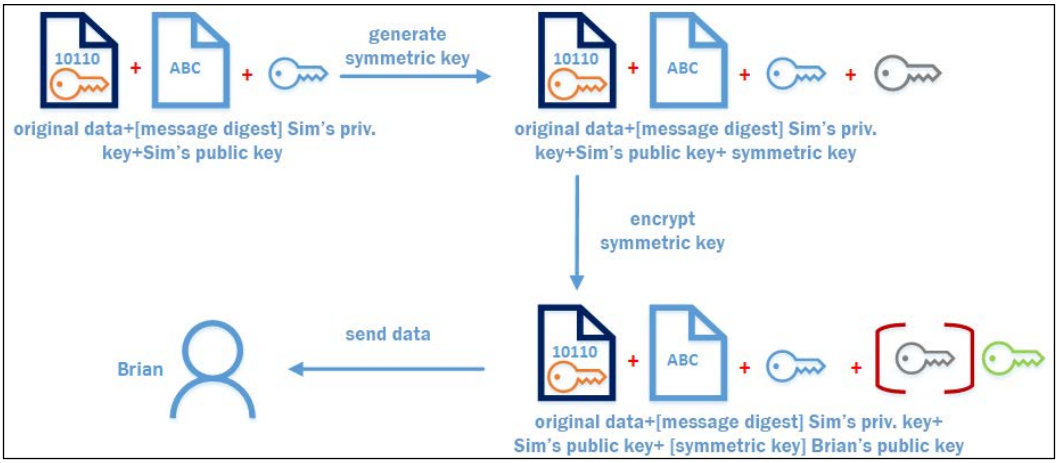


Figure 13.6: Data encryption

The next stage of the process is to encrypt the data. The first task in the process is to generate a one-time symmetric key to encrypt the data. An asymmetric algorithm is less efficient than symmetric algorithms for use with long data segments. Once a symmetric key is generated, the data will be encrypted using that key (including the **message digest** and signature). This symmetric key will be used by **Brian** to decrypt the message. Therefore, we need to ensure that it is only available to **Brian**. The best way to do this is to encrypt the symmetric key using **Brian's** public key. So, once he receives it, he will be able to decrypt it using his private key. This process only encrypts the symmetric key in itself, and the rest of the message will stay the same. Once this is complete, the data can be sent to **Brian**.

The next step of the process is to see how the decryption process will happen on Brian's side:

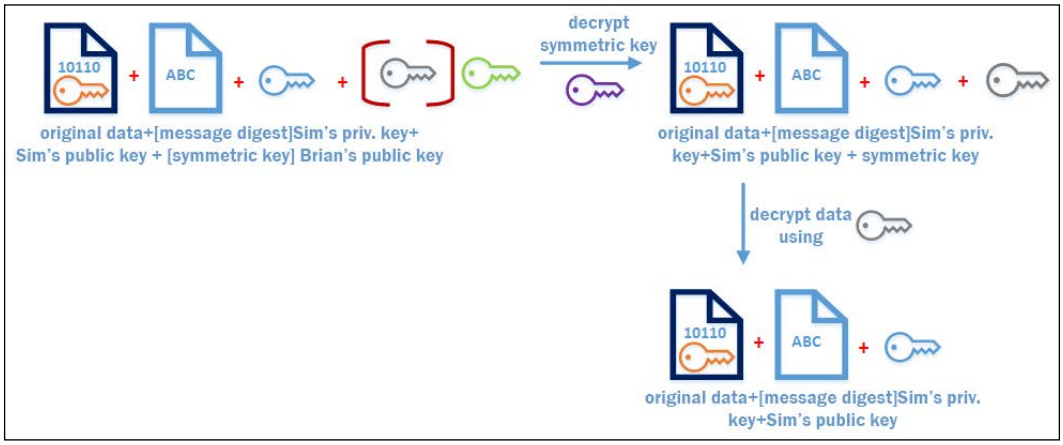


Figure 13.7: Data decryption

The message decryption process starts with decrypting the symmetric key. **Brian** needs the symmetric key to go further with the decryption process. It can only be decrypted using **Brian's** private key. Once it's decrypted, the symmetric key can be used to decrypt the **message digest** along with the signature. However, once the decryption is done, the same key information cannot be used to decrypt similar messages since it's a one-time key:

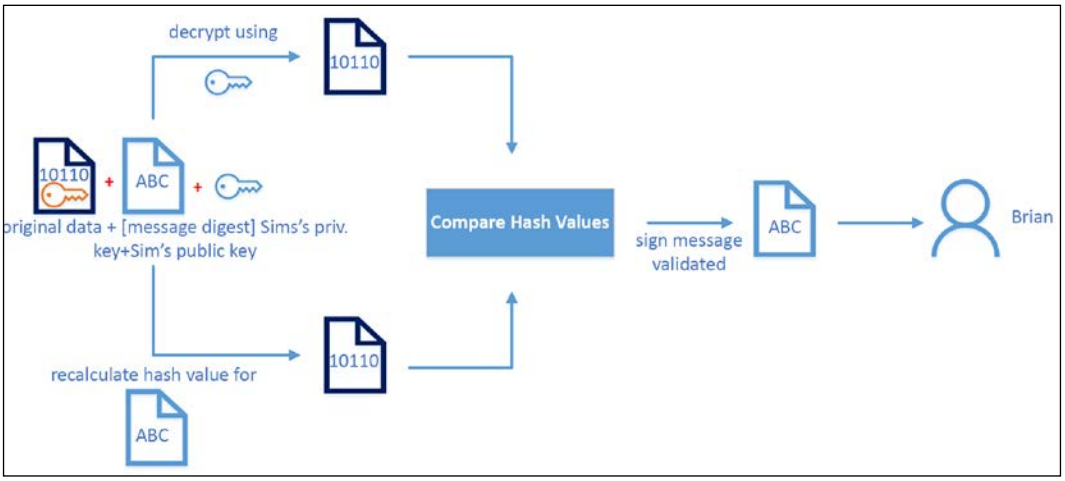


Figure 13.8: Verifying a signature

Now that we have the decrypted data, the next step is to verify the signature. At this point, we have the **message digest**, which is encrypted using **Simran's** private key. It can be decrypted using **Simran's** public key, which is attached to the encrypted message. Once it's decrypted, we can retrieve the **message digest**. This digest value is one-way. We cannot reverse engineer it. Therefore, the retrieved original data digest value will be recalculated using the exact same algorithm used by the sender. After that, this newly generated digest value will be compared with the digest value attached to the message. If the values are the same, it will confirm that the data wasn't modified during the communication process. When the values are equal, the signature will be verified and the original data will be issued to Brian. If the digest values are different, the message will be discarded, as it's been altered or not signed by **Simran**.

Now you have a good understanding of how the PKI environment works with the encryption/decryption process, as well as the digital signing/verification process.

SSL certificates

So far, we have talked about how asymmetric key pairs and symmetric keys work in PKI. But when we talk about PKI, it is the SSL certificates that come to mind. So, what is the role of the certificates?

I travel regularly between London and Seattle. When I reach Seattle-Tacoma International Airport, border security officers ask for my passport to verify my identity. They do not know me personally, but the passport I hold is issued by an authority that operates under international migration laws, and they certify that the person who owns the passport is "Dishan Francis." If they want to check its authenticity, they can confirm with the authority who issued my passport. Once they have confirmed my identity, they can also check the visa status to decide on my entry to the country.

Similarly, when looking at public-key cryptography, we know that a public key can be used by many applications and services. But how exactly can it be published, and how can the receiver confirm its authenticity? This is done using *digital certificates* issued by a **certification authority** (**CA**).

These digital certificates follow a similar structure, so everyone can understand it. It is similar to the way the passport works; folks with the border agency know where to look, even if it's a passport they have never seen before. These certificates are also time-bound, only being valid for a certain period. Once it has exceeded the validity period, it needs to be reissued, similar to a passport renewal.

The validity period is defined by the certificate template used by the object or service. In the event of exceeding the validity period, you need to contact the CA and request a renewal. It can either be an automatic renewal process or a manual renewal process.

The certificate includes the following data:

- **Version**: X.509 standards define the format of the certificate. It was first introduced in 1988, and currently, it uses version 3.
- **Serial number**: A unique identifier used by the CA to identify the certificate.
- **Signature algorithm**: The type of algorithm used by the CA to sign the digital certificate.
- **Signature hash algorithm**: The type of hash algorithm used by the CA.
- **Issuer**: The name of the CA who issued the certificate.
- **Valid from**: The day the certificate was issued by the CA.
- **Valid to**: The day the certificate will expire.
- **Subject**: The individual to whom the certificate was issued.
- **Public key**: The public key of the certificate owner. This will be the object or the service it was issued to.

The following screenshot shows a sample certificate:

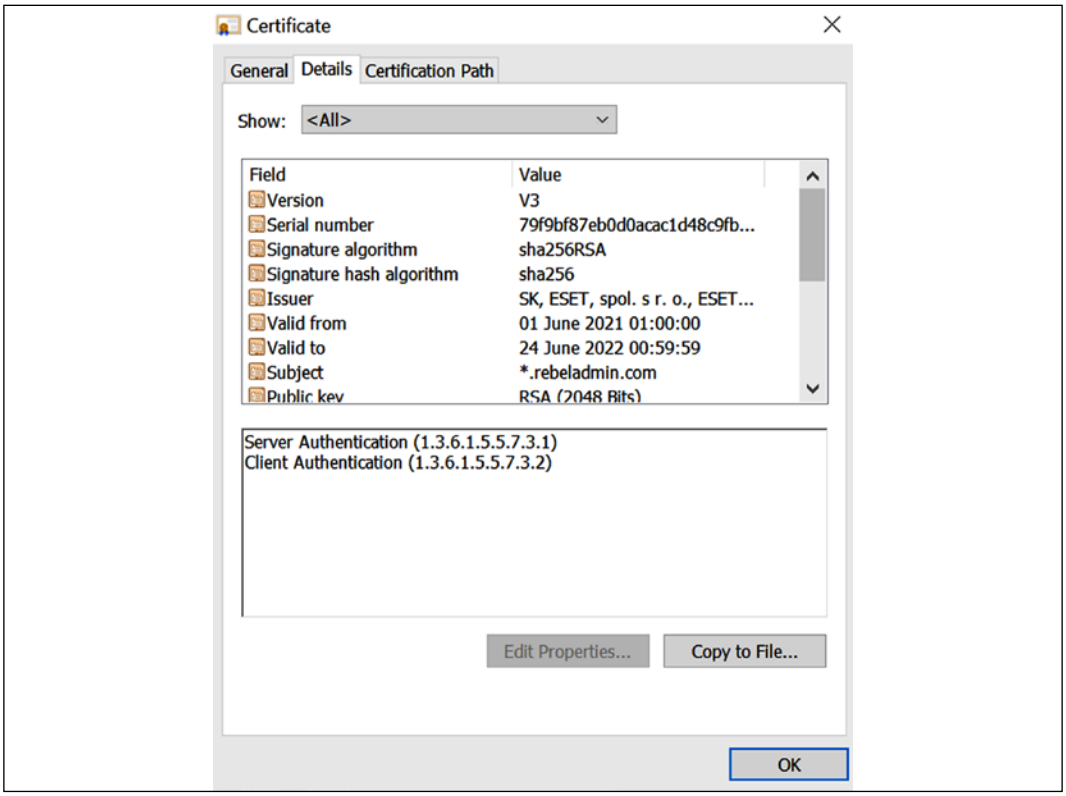


Figure 13.9: Sample certificate

Certificates are usually issued by CAs unless it is a self-signed certificate. Let's go ahead and look at different types of certification authorities.

Types of certification authorities

There are two types of certification authorities:

- **Private CAs**: This type of CA is mainly for internal use. By using private CAs, we can issue, manage, renew, and revoke certificates for internal users, devices, and services. It doesn't cost anything to issue certificates. **AD CS** is usually installed in an Active Directory environment. However, if necessary, AD CS components can also be installed in a workgroup environment (a standalone CA). Private CAs can also issue certificates to external resources, but this requires a "root certificate" to be shared with a third party.

- **Public CAs**: Public CAs are available for anyone, and users can pay and generate certificates. Most certificate vendors provide insurance along with certificates to ensure a certain level of protection. Internal CAs can trust internal resources as they are internal and known to the CAs. But for internet-facing services, it doesn't make sense to use internal CA-issued certificates as not everyone will trust the issuer. Instead of that, we can use a certificate issued by a well-known CA that everyone can trust. Most organizations use both types of CAs. All internet-facing services can use certificates issued by public CAs, and resources within corporate networks can use private CAs. A public CA can also issue certificates to internal resources, but the cost is prohibitive.

How do certificates work with digital signatures and encryption?

In the previous section, we looked at a scenario where Simran was sending encrypted and digitally signed data to Brian. During the process, we saw how Simran and Brian used each other's private and public keys. A public key has to be shared between two parties. Now, the problem we have is to work out how exactly the system knows that Brian's public key is his, and not from someone that is pretending to be Brian. To overcome this challenge, we can use certificates to verify whether shared public keys are from their purported source.

The digital signature process works as follows:

- Simran's private key will be used to encrypt the message digest. This private key will be retrieved from Simran's digital certificate. The private key verifies that the certificate is issued from a valid authority and that it's authentic.
- Simran's public key is also attached to the message as it can be used by Brian to verify the signature. This will be available to Brian via Simran's digital signature.

The data encryption process works as follows:

- A one-time symmetric key is used to encrypt the whole message, and after that, the key itself will be encrypted using Brian's public key. This public key will be retrieved using Brian's digital certificate, as it confirms that it is from Brian. This certificate is certified by a CA that Simran also trusts.

During the certificate validation process, the system will verify the certificates using the CA's public key as this will confirm the authenticity of the CA. It also checks the validity period of the certificates using the `Valid to` value in the certificate.

The data decryption process works as follows:

- The first step is to decrypt the one-time symmetric key using Brian's private key. This symmetric key will be retrieved using Brian's digital certificate. Once the key is retrieved, the key will be decrypted, and it will be used to decrypt the entire message.

The signature verification process works as follows:

- The message digest (hash) is encrypted using Simran's private key. It can be decrypted using Simran's public key. This public key can be retrieved from Simran's digital certificate. This certificate is issued by a CA that is trusted by Brian.

The rest of the steps are exactly the same as I explained in the *Signing, encryption, and decryption* section.

What can we do with certificates?

The scenarios explained previously are not the only ways in which we can use a certificate. Let's look at some of the scenarios where we can use certificates:

- When we allow communication between two systems in two different networks, it is important to protect the data transfer between two networks. Intercepted network traffic can cause serious infrastructure security issues. **IPSEC** is a network protocol used to encrypt network traffic using cryptographic keys. Using this, we can use certificates to encrypt traffic between two peers.
- The physical security of the data also matters when considering data security. More and more people are moving into mobile computing, and we need a way to protect the data inside these laptops and mobiles if they are stolen. **Encrypted File System** (**EFS**), which is based on certificates, can be used to encrypt and decrypt files. It will prevent any unauthorized access to data.
- Wireless networks have less control over connections compared to physical cable connections. Anyone who knows a wireless password can connect to the network. Instead of using passwords, we can use certificates to authenticate into a wireless network, and communication will only be allowed from trusted devices.
- In an Active Directory environment, the main authentication method is the username and password. In addition to that, certificates can be used to verify the authenticity of the users' or computers' authentication requests.

- Some services and applications have multiple roles and subservices integrated to provide one solution. The communication between these role services or subservices is unique and crucial for system operations. Therefore, those services can use certificates to verify the connectivity to each component and encrypt the communication between them to ensure the protection of application-related traffic.
- The **Secure/Multipurpose Internet Mail Extensions** (**S/MIME**) protocol can be used to encrypt and digitally sign email messages. When it digitally signs emails, it ensures the authenticity of the message and checks that no alteration has been made after the email is sent. This is done based on certificates. If you have Exchange Server 2013 SP1 or later, you can use S/MIME. Office 365 also supports this protocol.
- When we search for applications or drivers on the internet, sometimes we can observe fake websites that look similar to the original vendor. These websites usually have malware or viruses that can harm an infrastructure. Therefore, application vendors and manufacturers use certificates to digitally sign their applications, drivers, and code to confirm their authenticity.
- The most common use of certificates is with websites. The certificate of a website proves a couple of things. One thing it proves is the website's authenticity and that it is not a fake site. The other thing is that the user of the website knows that the communication between the user and the web server is secure and that any information passed between them is encrypted. This is important when we do online shopping.
- Non-repudiation is another benefit of certificates. If an object or service has signed a set of data, they cannot deny that they are not the private key holder. The data is signed using a private key, and the public key is attached to the data segment. These keys were retrieved from the certificate, which was issued by a trusted CA. This is why a public CA provides assurance, and they are bound to pay compensation to customers if the key is compromised.

Now we know what certificates can do. In the next section, we are going to go through AD CS components and their roles.

AD CS components

AD CS is a collection of role services, and they can be used to design the PKI for your organization. Let's look into each of these role services and their capabilities.

The CA

CA role service is responsible for issuing, storing, managing, and revoking certificates. The PKI setup can have multiple CAs. There are two main types of CAs:

- **The root CA**: The root CA is the most trusted CA in the PKI setup. A compromised root CA will compromise an entire PKI. Therefore, the security of the root CA is crucial. Best practice is to bring the root CA online only when required. By considering the security and hierarchy of the PKI, it is recommended to use the root CA only to issue certificates to subordinate CAs.
- **Subordinate CAs**: In PKI, subordinate CAs are responsible for issuing, storing, managing, and revoking certificates for users, devices, or services. Once a CA receives a certificate request, it will process it and issue the certificate. A PKI can have multiple subordinate CAs. Each subordinate server should have its own certificate from the root CA. The validity period of these certificates is normally longer than that for ordinary certificates. A subordinate CA also needs to renew its certificate from the root CA when it reaches the end of the validity period:

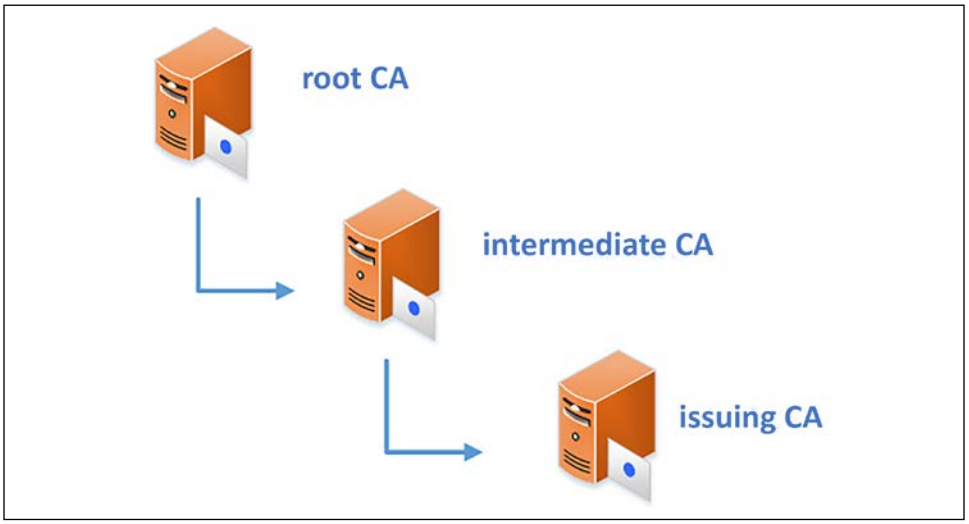


Figure 13.10: CA hierarchy

Subordinate CAs can have more subordinate CAs under them. In such situations, subordinate CAs are also responsible for issuing certificates for their own subordinate CAs. The parent subordinate CAs are called **intermediate CAs**. These will not be responsible for issuing certificates to users, devices, or services. Subordinate CAs that issue certificates to users, devices, or services are called **issuing CAs**.

Certificate Enrollment Web Service

Certificate Enrollment Web Service allows users, computers, or services to request a certificate or renew a certificate via a web browser, even if it is not domain-joined or is temporarily out of the corporate network. If it is domain-joined and in the corporate network, then auto-enrollment or a template-based request process can be used to request a certificate.

Certificate Enrollment Policy Web Service

This role service works with Certificate Enrollment Web Service and allows users, computers, and services to perform policy-based certificate enrollment. Similar to Certificate Enrollment Web Service, client computers can be non-domain-joined devices or domain-joined devices that are outside the company's network boundaries. When a client requests policy information, the Enrollment Policy Web Service queries the AD DS using **Lightweight Directory Access Protocol** (**LDAP**) for the policy information and then delivers it to the client via HTTPS. This information will be cached and used for similar requests. Once the user has the policy information, then they can request the certificate using Certificate Enrollment Web Service.

Certification Authority Web Enrollment

This is similar to a web interface for a CA. Users, computers, or services can request certificates using a web interface. Using the interface, users can also download the root certificates and intermediate certificates to validate a certificate. This can be used to request the **certificate revocation list** (**CRL**). The CRL includes all the certificates that have expired or been revoked within the PKI. If any presented certificate matches an entry in the CRL, it will be automatically refused.

Network Device Enrollment Service

Network devices such as routers, switches, and firewalls can have device certificates to verify the authenticity of the traffic that passes through them. The majority of these devices are not domain-joined, and their operating systems are also very unique and do not support typical Windows computer functions. To request or retrieve certificates, network devices use **Simple Certificate Enrollment Protocol** (**SCEP**). It allows network devices to have X.509 version 3 certificates similar to other domain-joined devices. This is important since, if a device is going to use IPSEC, it must have an X.509 version 3 certificate.

Online Responder

Online Responder is responsible for producing information about certificate status. When I talked about Certification Authority Web Enrollment, I explained how the CRL includes the entire list of certificates that are expired or have been revoked within the PKI. The list will keep growing based on the number of certificates it manages.

Instead of using bulk data, Online Responder will respond to individual requests from users to verify the status of a particular certificate. This is more efficient than the CRL method as the request is focused on finding out the status of a certificate at a given time.

The types of CA

Based on the installation mode, CAs can be divided into two types: the **Standalone CA** and the **Enterprise CA**. The best way to explain the capabilities of both types is to compare them:

Feature	Standalone CA	Enterprise CA
AD DS dependency	Does not depend on AD DS; can be installed on a member server or standalone server in a workgroup	Can only be installed on a member server
Operate offline	Can stay offline	Cannot be offline
Customized certificate templates	Only supports standard templates	Supported
Supported enrollment methods	Manual or web enrollment	Auto, manual, or web enrollment
Certificate approval process	Manual	Manual or automatic based on the policy
User input for certificate fields	Manual	Retrieved from AD DS
Certificate issuing and managing using AD DS	N/A	Supported

Standalone CAs are mostly used as the root CA. In the previous section, I explained how important root CA security is. The standalone CA supports keeping the server offline and can bring it online when it needs to issue a certificate or renew a certificate. Root CAs are only used to issue certificates to a subordinate CA.

So, manual processing and approval are manageable, as this may only have to be done every few years. This type is also valid for public CAs. Issuing CAs, on the other hand, is responsible for day-to-day CA-related tasks such as issuing, managing, storing, renewing, and revoking certificates for users, devices, or services. Depending on the infrastructure size, there can be hundreds or thousands of users and devices using these issuing CAs. If the request and approval process is manual, it may take a lot of manpower to maintain it. Therefore, in corporate networks, it is always recommended to use an enterprise-type CA.

Enterprise CAs allow engineers to create certificate templates with specific requirements and publish those via AD DS. End users can request certificates based on these templates. Enterprise CAs can only be installed on the Windows Server Enterprise Edition or the Datacenter Edition.

Planning PKI

By now, we understand what PKI is and how it works. You have also learned about AD CS components and their capabilities. The next thing is to plan the deployment of the PKI. In this section, we will look into the things we need to consider during the PKI planning process.

Internal or public CAs

AD CS is not just a role that we can install on a server and leave to run. It requires knowledge to set up and operate. It needs to be maintained like any other IT system. We also need to consider high availability. All this comes at a cost. Public CA certificates need to be purchased through a service provider. Each provider has many different types of certificates with different price ranges. It is important to evaluate these associated costs against your company's requirements. If the company is looking for a few web service certificates, there is no point in maintaining a few servers internally just for that. If a public CA can offer the same thing for $15, it makes sense to invest in that rather than wasting resources and money by maintaining an internal CA. However, it's not only the cost that we need to consider. An internal CA provides greater flexibility when it comes to administration. It allows the creation of templates and policies according to organizational requirements. Public CAs give only limited control. All you can do is pay for the certificate, submit the signing request, and then download the certificate once it's issued. Public CAs do have a reputation. If a user outside the corporate network needs to trust a certificate issued by the internal CA, the user needs to trust the issuing CA and the rest of the CAs in the chain, but not everyone would like to do that. However, if the certificate is from a reputable CA, it gives users confidence in the certificate and the content protected by it.

When you use a public CA, customers can get professional support via the vendor whenever required – there is no need to have advanced knowledge to request and retrieve a digital certificate – whereas an internal CA requires skill to deploy, manage, and maintain. Considering all these pros and cons, we need to decide which CA model is best suited for the organization's requirements.

Identifying the correct object types

Certificates can be issued to users, computers, services, or network devices. User certificates are mainly used for the authentication process. Certificates can also be used with an application or service. User certificates will be installed in a user certificate store. Computer certificates allow you to uniquely identify a device. Computer certificates will be stored in a computer certificate store. Network devices are allowed to use X.509 certificates, and they can be used to certify a device and encrypt the traffic passing through it. Services such as web and email can use certificates to authenticate or encrypt data. A service itself will not have a certificate, but it will use a computer certificate or user certificate that is associated with the service. It is important to understand what types of objects will have certificates, as the configuration of the CA will be based on it. As an example, if network devices need certificates, we need to install Network Device Enrollment Service and configure it. Certificate templates should be modified to support the object type.

With a CA, we can use Microsoft's default cryptographic provider, which is the **RSA Microsoft Software Key Storage Provider**, or other advanced providers, such as ECDSA_P256, ECDSA_P521, or ECDSA_P384. Based on the provider used, the length of the cryptography key and hash algorithm will also change. Unless there are specific reasons to do otherwise, it is always recommended to use Microsoft RSA.

The cryptographic key length

When the size of the cryptographic key is increased, it gets harder and harder for attackers to compromise the kyes. The minimum recommended key size is 2048 bits, and this size can change based on the cryptographic provider. When the key size is increased, the encryption/decryption process takes more system resources.

Hash algorithms

During CA deployment, we can choose the hash algorithm standard. By default, it is SHA256, and it can change into SHA384, SHA512, or MD5. SHA1 is no longer recommended for use as it has been proven to be a weaker hash algorithm than the others.

The certificate validity period

Certificates are time-bound. Using certificate templates, we can specify the validity period of a certificate. It can be months or years. Before a certificate expires, it will need to be renewed (the certificate template will define how many days or weeks in advance it can be renewed). The expiry date of an issued certificate cannot be changed unless there is a renewal or reprocess.

The CA hierarchy

The root CA in a PKI can have more subordinate CAs. The number of subordinate CAs depends depend on the organization's requirements. There are two main types of hierarchical design: two-tier and three-tier. These will be explained in detail in the next section, but here, what I want to emphasize is that selecting the correct hierarchy model will reduce operational costs and resource waste.

High availability

Based on the organization's requirements, we have to decide what the best solution is in terms of maintaining high availability. If it's a heavily used PKI, the availability of the CA role services is important. Service uptime can be guaranteed by running the service in a clustered environment or using advanced site recovery solutions such as Azure Site Recovery. Based on the maximum downtime an organization can afford, the investment in high availability products, the technologies used, and the general approach taken will also change.

Deciding certificate templates

Not every user, computer, or service needs the same type of certificate. If you purchase certificates from a public CA, there are lots of different types of certificates available, all with different prices. Each of these certificates has different value-added services. Certificate templates allow you to create custom templates that can match different certificate requirements. As an example, user certificates may need to be renewed every year due to staff changes, while computer certificates may be renewed every 5 years. As part of the planning process, we need to evaluate certificate requirements so that we can create new templates to match them (only if Enterprise CA).

The CA boundary

Before starting a deployment, it is important to decide on the operation boundaries of the PKI. We need to decide under which domain, forest, or network segment the PKI will operate. Once the boundaries are defined, it can be hard to extend them later without any physical or logical network layer changes. As an example, if you have a CA in the corporate network and you need to extend the perimeter network, that will require some network layer changes. In another example, if a partner company or third party wants to use the corporate CA, that will require AD CS role changes, firewall changes, network routing changes, DNS changes, and more. Therefore, it's important to evaluate these types of operational requirements in the planning process.

PKI deployment models

At several points in this chapter, I have mentioned the PKI hierarchy and components such as root CAs, intermediate CAs, and issuing CAs. Based on the business and operational requirements, the PKI topology will also change. There are three deployment models that we can use to address the PKI requirements. In this section, we will look into these models and their characteristics.

The single-tier model

The single-tier model is also referred to as the **one-tier model**, and it is the simplest deployment model for a PKI. This is not recommended for use in any production network as it's a single point of failure for the entire PKI:

Figure 13.11: Single-tier model

In this model, a single CA will act as a **root CA and issuing CA**. As explained previously, the root CA is the most trusted CA in the PKI hierarchy. A compromised root CA will compromise the entire PKI. In this model, it's a single server, so any breach of the server will easily compromise the entire PKI, as it doesn't need to spread through different hierarchical levels. However, this model is easy to implement and easy to manage.

Some CA-aware applications require certificates in order to function. **System Center Operations Manager** (**SCOM**) is one good example. It uses certificates to secure web interfaces, authenticate management servers, and more. If the organization doesn't have an internal CA, you can either purchase certificates from the vendor or deploy a new CA. Engineers usually use this single-tier model as it's only used for a specific application or task:

Advantages	Disadvantages
Fewer resources are needed to manage it, as it's all running from a single server.	There is a high possibility of being compromised as the root CA is online and running all the PKI-related roles from one single server. If someone gets access to a private key of the root CA, they have complete ownership over the PKI.
Deployment is faster and it is possible to get the CA running in a short timeframe.	There is a lack of redundancy, as certificate issuing and management all depend on a single server; the server's availability will decide the availability of the PKI.
N/A.	It is not scalable, and the hierarchy will need to be restructured if more role servers need to be added.

The two-tier model

This is the most commonly used PKI deployment model in corporate networks. In this design, the root CA is kept offline. It will help to protect the private key of the root certificate from being compromised.

Root CAs will issue certificates for subordinate CAs, and subordinate CAs are responsible for issuing certificates for objects and services:

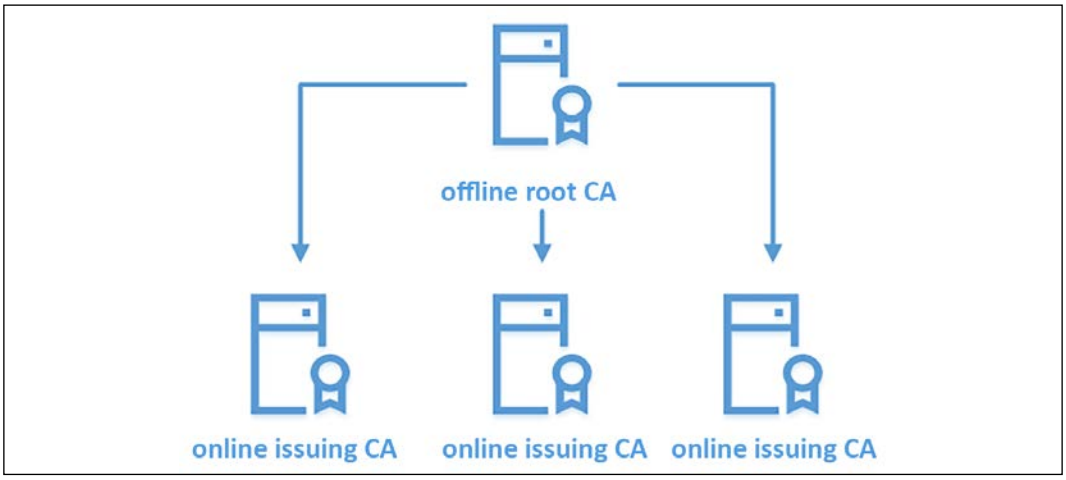


Figure 13.12: Two-tier model

If a subordinate CA's certificate expires, the **offline root CA** will need to be brought online to renew the certificate. The root CA doesn't need to be a domain member, and it should be operating at the workgroup level (a standalone CA). Therefore, certificate enrollment, approval, and renewal will be a manual process. This is a scalable solution and the number of issuing CAs can be increased based on the organization's requirements. This allows us to extend the CA boundaries to multiple sites, too. In a single-tier model, if the PKI was compromised, in order to recover all the issued certificates, you would need to manually remove them from the respective devices. In a two-tier model, though, we just need to revoke the certificates issued by the CA, publish the CRL, and then reissue the certificates:

Advantages	**Disadvantages**
It provides improved PKI security, as in this model, the root CA should stay offline. It will protect the private key from being compromised.	High maintenance – it requires the maintenance of multiple systems and skills to process the manual certificate request/approval/renewal process between the root and subordinate CAs.
Flexible scalability – can start small and expand by adding additional subordinate CAs when required.	Cost – the cost of resources and licenses is high compared to those for a single-tier model.
You can restrict the issuing CA's impact on the CA hierarchy by controlling the scope of the certificate. It will prevent the issuing of rogue certificates.	The manual certificate renewal process between the root CA and subordinate CAs adds more risks; if administrators forget to renew a certificate on time, it can bring the whole PKI down.
Improved performance, as workloads can be shared among multiple subordinate CAs.	N/A.
Flexible maintenance capabilities, as there are fewer dependencies.	N/A.

Three-tier models

The three-tier model is the most advanced model on the list, and it operates with greater security, scalability, and control. Similar to a two-tier model, it also has an offline root CA and online issuing CAs. In addition to that, there are offline intermediate CAs, which operate between the root and subordinate CAs. The role of the intermediate CAs is to operate as policy CAs. In larger organizations, different departments, different sites, and different operation units can have different certificate requirements. As an example, a certificate issued to a perimeter network will require a manual approval process, while users in the corporate network will prefer auto-approval. IT teams prefer to have large keys and advanced cryptographic providers for their certificates, while other users operate with the default RSA algorithms.

All these different requirements are defined by the policy CA, and it publishes relevant templates and procedures to the other issuing CAs:

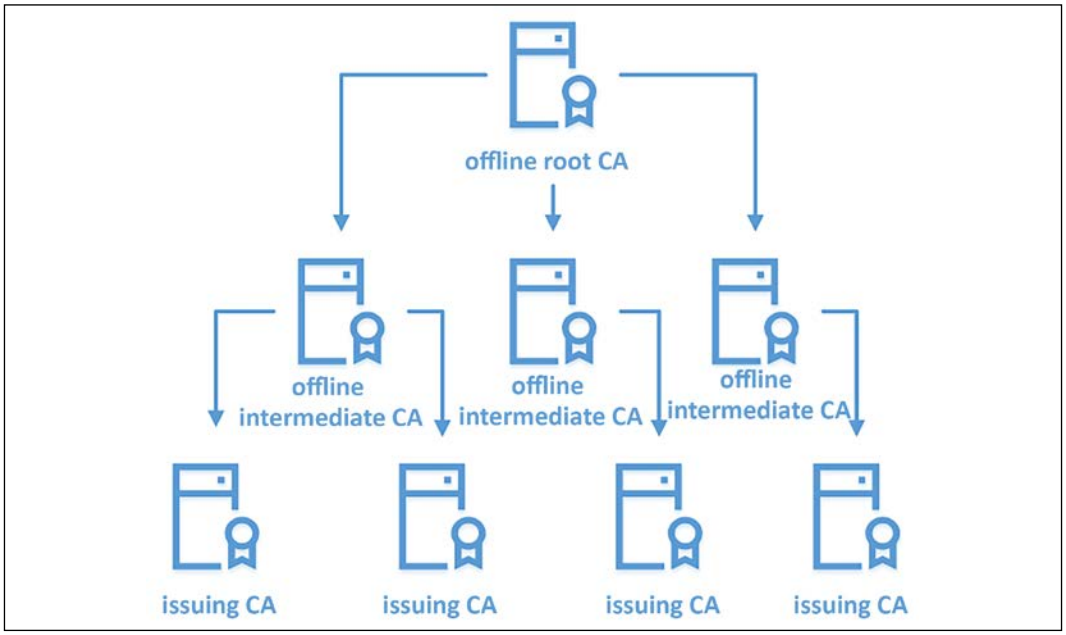


Figure 13.13: Three-tier model

This model adds another layer of security to the hierarchy. However, if you are not using CA policies, the intermediate tier will not be useful. It can be a waste of money and resources. Therefore, most organizations prefer a two-tier model to start with, and then expand as required.

In this model, both the root CA and intermediate CAs operate as standalone CAs. The root CA will only issue certificates to intermediate CAs, and those will only issue certificates to issuing CAs:

Advantages	**Disadvantage**
Improved security, as it adds another layer of CAs to the certificate verification.	Cost – the cost of resources and licenses is high, as three layers need to be maintained. This also increases the operating costs.
Greater scalability, as each tier can extend horizontally.	High maintenance – when the number of servers increases, the efforts needed to maintain them also increase. Both the tiers that operate standalone CAs require additional maintenance, as automatic certificate request/approval/renewal is not supported.

In the event of the compromise of the issuing CA, the intermediate CA can revoke the compromised CA with minimum impact to the existing setup.	The implementation complexity is high compared to other models.
High-performance setup, as workloads are distributed and administrative boundaries are well defined by intermediate CAs.	N/A.
Improved control over certificate policies, allowing enterprises to have tailored certificates.	N/A.
High availability, as dependencies are further reduced.	N/A.

Now that we know about different PKI deployment models, in the next section, I am going to demonstrate how to set up a PKI.

Setting up a PKI

Now we have finished the theory part of this chapter and are moving on to the deployment part. In this section, I am going to demonstrate how we can set up a PKI using the two-tier model. I have used this model as it is the most commonly used model for medium and large organizations:

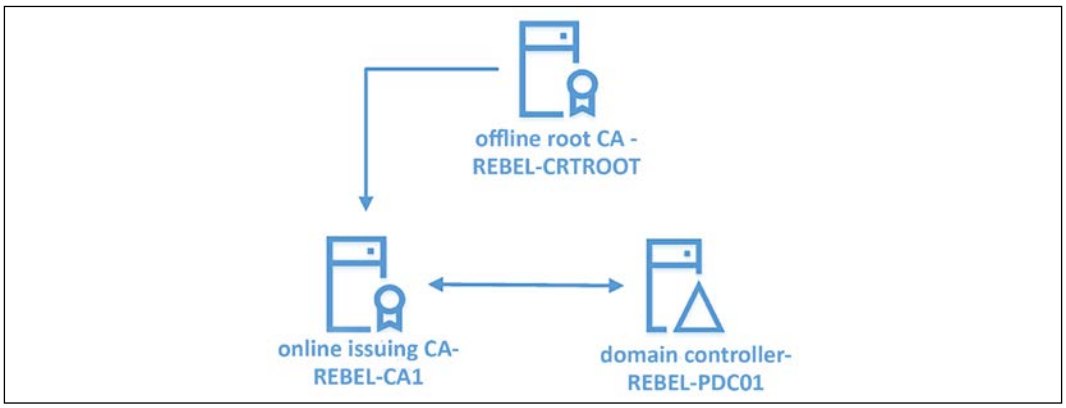


Figure 13.14: Planned PKI setup

The preceding diagram explains the setup I am going to configure. Here, I have one domain controller, one standalone root CA, and one issuing CA. All are running with Windows Server 2022 with the latest updates.

Setting up a standalone root CA

The first step is to set up the standalone root CA. This is not a domain member server and is operating on the workgroup level. Configuring it on a separate VLAN will add additional security to the root CA.

Once the server is ready, log in to the server as a member of the local administrator group. The first task is to install the AD CS role service. This can be done using the following command:

```
Add-WindowsFeature ADCS-Cert-Authority -IncludeManagementTools
```

Once the role service is installed, the next step is to configure the role and get the CA up and running:

```
Install-ADcsCertificationAuthority -CACommonName "REBELAdmin Root CA"
-CAType StandaloneRootCA -CryptoProviderName "RSA#Microsoft Software
Key Storage Provider" -HashAlgorithmName SHA256 -KeyLength 2048
-ValidityPeriod Years -ValidityPeriodUnits 20
```

In the preceding code, `-CACommonName` defines the common name for the CA. Then, `-CAType` defines the CA operation type. In our case, it is `-StandaloneRootCA`. The other available types for CA type are `EnterpriseRootCA`, `EnterpriseSubordinateCA`, and `StandaloneSubordinateCA`. Now, `-CryptoProviderName` specifies the cryptographic service provider. In this demo, I am using the Microsoft default service provider. `-HashAlgorithmName` defines the hashing algorithm used by the CA. The option for it will be changed, based on the **Cryptographic Service Provider** (**CSP**) we choose. SHA1 is no longer counted as a secure algorithm; it is recommended to use SHA256 or above. `-KeyLength` specifies the key size for the algorithm. In this demo, I am using the `2048` key. `-ValidityPeriod` defines the validity period of CA certificates. It can be hours, days, weeks, months, or years. `-ValidityPeriodUnits` is followed by `-ValidityPeriod` and specifies how many hours, days, weeks, months, or years it will be valid for. In my demo, I am using 20 years:

```
PS C:\Users\dfrancis> Install-ADcsCertificationAuthority -CACommonName "REBELAdmin Root CA" -CAType StandaloneRootCA -Cr
yptoProviderName "RSA#Microsoft Software Key Storage Provider" -HashAlgorithmName SHA256 -KeyLength 2048 -ValidityPeriod
 Years -ValidityPeriodUnits 20

Confirm
Are you sure you want to perform this action?
Performing the operation "Install-AdcsCertificationAuthority" on target "REBEL-CRTROOT".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [?] Help (default is "Y"): A

RunspaceId  : cc73471e-e2d8-4756-a194-d50383e01d84
ErrorId     : 0
ErrorString :
```

Figure 13.15: Configuring an AD CS role

After we have the root CA up and running, we need to do certain configuration changes before we use it.

DSConfigDN

As I mentioned earlier, in a two-tier PKI setup, we use a standalone root CA and it is not part of the domain. However, **CRL Distribution Points** (**CDP**) and **authority information access** (**AIA**) locations, which are required by the CA, will be stored in Domain Controller. Since these records use **Distinguished Names** (**DN**) with a domain, the root CA needs to be aware of the domain information to publish it properly. It will retrieve this information via a registry key:

```
certutil.exe –setreg ca\DSConfigDN CN=Configuration,DC=rebeladmin,DC=c
om
```

The preceding command needs to run using Command Prompt.

CDP locations

CDPs define the location from where the CRL can be retrieved. This is a web-based location and should be accessible via HTTP. This list will be used by the certificate validator to verify the given certificate against the revocation list.

Before we do this, we need to prepare the web server. It should be a domain member, similar to the issuing CA.

In my demonstration, I am going to use the same issuing CA as the CDP location.

The web server can be installed using the following command:

```
Install-WindowsFeature Web-WebServer -IncludeManagementTools
```

Next, create a folder and create a share so that it can be used as the virtual directory:

```
mkdir C:\CertEnroll
New-smbshare -name CertEnroll C:\CertEnroll -FullAccess
SYSTEM,"rebeladmin\Domain Admins" -ChangeAccess "rebeladmin\Cert
Publishers"
```

As part of the exercise, I am setting the share permissions to `rebeladmin\Domain Admins` (full access) and `rebeladmin\Cert Publishers` (change access).

After the folder has been created with the relevant permissions, we also need to copy the root CA certificate and the CRL file to it:

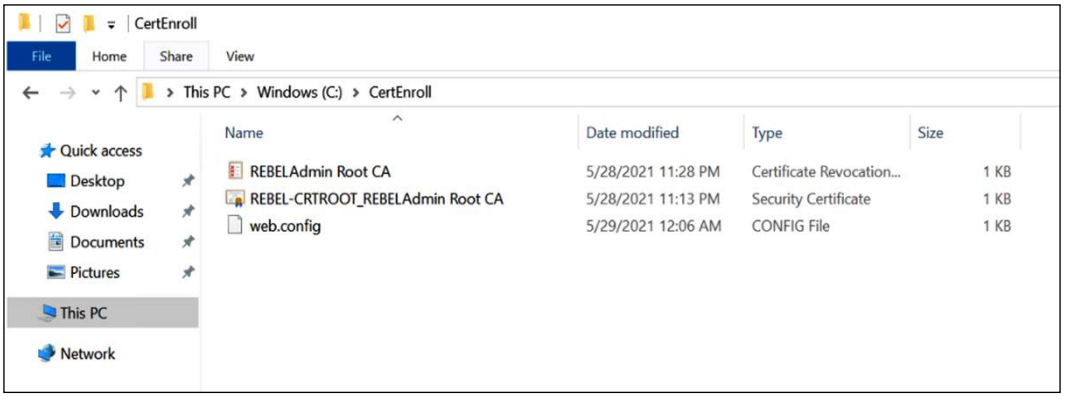


Figure 13.16: Configuring an AD CS role

After that, load the **Internet Information Services** (**IIS**) manager and add a virtual directory, `CertEnroll`, with the aforementioned path:

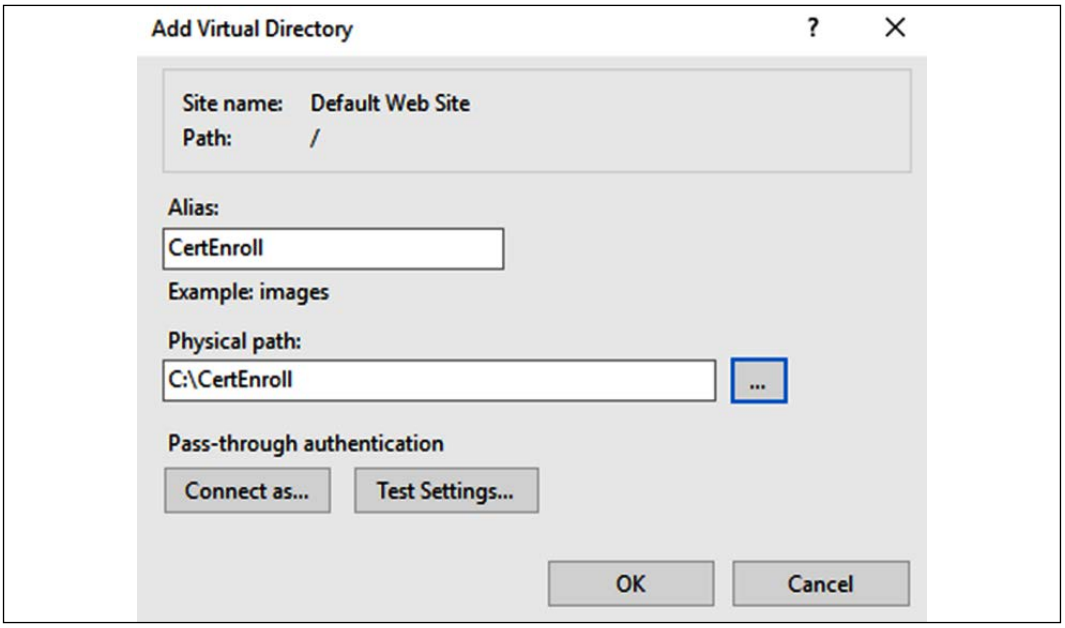


Figure 13.17: Setting up the CertEnroll virtual directory

Last but not least, we need to create a DNS record for the service URL. In this demo, I am using `crt.rebeladmin.com` as the domain name. This will allow us to access the new distribution point using `http://crt.rebeladmin.com/CertEnroll`.

Now everything is ready, and we can publish the CDP settings using the following command:

```
certutil -setreg CA\CRLPublicationURLs "1:C:\Windows\system32\CertSrv\
CertEnroll\%3%8%9.crl \n10:ldap:///CN=%7%8,CN=%2,CN=CDP,CN=Public
Key Services,CN=Services,%6%10\n2:http://crt.rebeladmin.com/
CertEnroll/%3%8%9.crl"
```

The preceding command needs to run on the root CA server.

The single numbers in the command refer to the options, and numbers with `%` refer to the variables:

Option	Details
0	No changes.
1	Publish the CRL to the given location.
2	Attach the CDP extensions of issued certificates.
4	Include in the CRL to find the delta CRL locations.
8	Specify whether there is a need to publish all CRL information to AD when publishing manually.
64	Delta CRL location.
128	Include the **Issuing Distribution Point** (**IDP**) extension of the issued CRL.

All these settings can be specified using the GUI. To access it, go to **Server Manager | Tools | Certification Authority**, right-click and select **Properties** of the server, and then go to the **Extension** tab.

There, you can add the following variables using the GUI:

Variable	GUI reference	Details
%1	<ServerDNSName>	The DNS name of the CA server
%2	<ServerShortName>	The NetBIOS name of the CA server
%3	<CAName>	The given name for the CA
%4	<CertificateName>	The renewal extension of the CA
%6	<ConfigurationContainer>	DN of the configuration container in AD
%7	<CATruncatedName>	Truncated name of the CA (32 characters)
%8	<CRLNameSuffix>	Inserts a name suffix at the end of the filename before publishing a CRL

%9	<DeltaCRLAllowed>	Replaces CRLNameSuffix with a separate suffix to use the delta CRL
%10	<CDPObjectClass>	The object class identifier for the CDP
%11	<CAObjectClass>	The object class identifier for a CA

Once the CDP settings are in place, the next step is to go ahead and set up the AIA locations.

AIA locations

AIA is an extension that is in the certificate and defines the location where the application or the service can retrieve the issuing CA's certificate. This is also a web-based path, and we can use the same location we used for the CDP.

The AIA location can be set using the following command:

```
certutil -setreg CA\CACertPublicationURLs "1:C:\Windows\system32\
CertSrv\CertEnroll\%1_%3%4.crt\n2:ldap:///CN=%7,CN=AIA,CN=Public
Key Services,CN=Services,%6%11\n2:http://crt.rebeladmin.com/
CertEnroll/%1_%3%4.crt"
```

The preceding command needs to run on the root CA server.

As shown in the following table, the options are very much similar to the CDP options, but there are a few small differences as well:

Option	Details
0	No changes.
1	Publish a CA certificate to a given location.
2	Attach the AIA extensions of issued certificates.
32	Attach the **Online Certificate Status Protocol** (**OCSP**) extensions.

CA time limits

When we set up the CA, we defined the CA validity period as 20 years. But that doesn't mean every certificate it issues will have a 20-year validity period. Root CAs will only issue certificates to issuing CAs. For this demo, I will set the certificate validity period to 10 years:

```
certutil -setreg ca\ValidityPeriod "Years"
certutil -setreg ca\ValidityPeriodUnits 10
```

CRL time limits

The CRL also has some associated time limits:

```
Certutil -setreg CA\CRLPeriodUnits 13
Certutil -setreg CA\CRLPeriod "Weeks"
Certutil -setreg CA\CRLDeltaPeriodUnits 0
Certutil -setreg CA\CRLOverlapPeriodUnits 6
Certutil -setreg CA\CRLOverlapPeriod "Hours"
```

In the preceding commands, the following is true:

- **CRLPeriodUnits**: This specifies the number of days, weeks, months, or years for which the CRL will be valid.
- **CRLPeriod**: This specifies whether the CRL validity period is measured by days, weeks, months, or years.
- **CRLDeltaPeriodUnits**: This specifies the number of days, weeks, months, or years that the delta CRL is valid for. Offline CAs should disable this.
- **CRLOverlapPeriodUnits**: This specifies the number of days, weeks, months, or years that the CRL can overlap.
- **CRLOverlapPeriod**: This specifies whether the CRL overlapping validity period is measured by days, weeks, months, or years.

Now we have all the settings submitted. To apply the changes, run the following command:

```
restart-service certsvc
```

This will restart the certificate service and apply all the pending changes.

The new CRL

The next step is to create a new CRL, which can be generated using the following command:

```
certutil -crl
```

Once that's done, there will be two files under `C:\Windows\System32\CertSrv\CertEnroll`:

```
PS C:\Windows\System32\CertSrv\CertEnroll> dir

    Directory: C:\Windows\System32\CertSrv\CertEnroll

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a---          5/28/2021 11:28 PM            694 REBELAdmin Root CA.crl
-a---          5/28/2021 11:13 PM            793 REBEL-CRTROOT_REBELAdmin Root CA.crt

PS C:\Windows\System32\CertSrv\CertEnroll>
```

Figure 13.18: Root certificate and CRL

This completes the initial configuration of a standalone root CA. As the next step, we need to publish the root CA certificate to Active Directory. By publishing this, AD-joined computers will have the root CA certificates under the computer's Trusted Root Certification Authorities certificates.

Publishing the root CA data to Active Directory

In the preceding screenshot, we see that we have two files. One ends with `.crt`. This is the root CA certificate. In order to distribute it to other clients in the domain, it first needs to be published to Active Directory. To do that, go ahead and copy this file from the root CA to the Active Directory server. Then, log in to the domain controller as Domain Admin or Enterprise Admin and run the following command:

```
certutil -f -dspublish "REBEL-CRTROOT_REBELAdmin Root CA.crt" RootCA
```

The preceding command needs to run from Command Prompt. In the demo, I have copied the file to the `C:\` drive and am running the command from the same path:

```
Administrator: C:\Program Files\PowerShell\7\pwsh.exe
PowerShell 7.1.4
Copyright (c) Microsoft Corporation.

https://aka.ms/powershell
Type 'help' to get help.

PS C:\Windows\System32> cd c:/
PS C:\> certutil -f -dspublish "REBEL-CRTROOT_REBELAdmin Root CA.crt" RootCA
ldap:///CN=REBELAdmin Root CA,CN=Certification Authorities,CN=Public Key Services,CN=Services,CN=Configuration,DC=rebela
dmin,DC=com?cACertificate

Certificate added to DS store.

ldap:///CN=REBELAdmin Root CA,CN=AIA,CN=Public Key Services,CN=Services,CN=Configuration,DC=rebeladmin,DC=com?cACertific
ate

Certificate added to DS store.

CertUtil: -dsPublish command completed successfully.
PS C:\>
```

Figure 13.19: Publishing the root CA data

The next file ends with `.crl`. This is the root CA's CRL. This also needs to be published to Active Directory so that everyone in the domain is aware of it. To do that, copy the file from the root CA to the domain controller and run the following command:

```
certutil –f –dspublish "REBELAdmin Root CA.crl"
```

Setting up the issuing CA

Now that we're done with the root CA setup, the next step is to set up the issuing CA. Issuing CAs will be run from a domain member server and will be AD-integrated. To begin the installation, log in to the server as the Domain Admin or Enterprise Admin.

1. The first task will be to install the AD CS role:

```
Add-WindowsFeature ADCS-Cert-Authority -IncludeManagementTools
```

2. I will use the same server for the Web Enrollment Role Service. This can be added using the following command:

```
Add-WindowsFeature ADCS-web-enrollment
```

3. After that, we can configure the role service using the following command:

```
Install-ADcsCertificationAuthority -CACommonName "REBELAdmin
IssuingCA" -CAType EnterpriseSubordinateCA -CryptoProviderName
"RSA#Microsoft Software Key Storage Provider" -HashAlgorithmName
SHA256 -KeyLength 2048
```

4. To configure the Web Enrollment Role Service, use the following command:

```
Install-ADCSwebenrollment
```

5. This completes the initial role configuration process of the issuing CA. The next step is to create a certificate for the issuing CA.

Issuing a certificate for the issuing CA

The issuing CA needs a certificate issued from the root CA we just deployed. During the role configuration process, it automatically creates the certificate request under `C:\`, and the exact filename will be listed in the command output from the previous command:

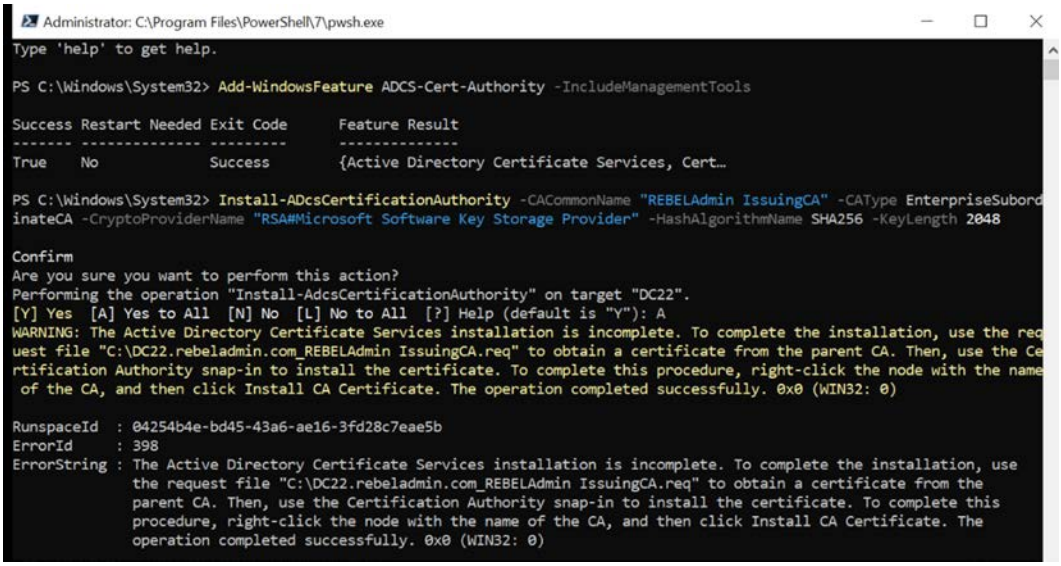


Figure 13.20: Issuing CA role configuration

1. The file needs to be copied from the issuing CA to the root CA, and then you need to execute the following command:

```
certreq -submit "REBEL-CA1.rebeladmin.com_REBELAdmin IssuingCA.
req"
```

2. As I explained before, any request to the root CA will be processed manually, so this request will be waiting for manual approval. To approve the certificate, go to **Server Manager | Tools | Certification Authority | Pending Requests**; right-click on the certificate and select **All Tasks | Issue**.
3. Once it has been issued, it needs to be exported and imported into the issuing CA:

```
certreq -retrieve 2 "C:\REBEL-CA1.rebeladmin.com_REBELAdmin_
IssuingCA.crt"
```

4. The preceding command will export the certificate. The number `2` is the **request ID** in the CA **Microsoft Management Console** (**MMC**).
5. Once the export is complete, move the file to the issuing CA, and from there, run the following command:

```
Certutil –installcert "C:\REBEL-CA1.rebeladmin.com_REBELAdmin_
IssuingCA.crtstart-service certsvc
```

This will import the certificate and start the CA service.

Post-configuration tasks

In the same way as the root CA, following the initial service setup, we need to do certain configuration changes.

CDP locations

The CDP location configuration is similar to that of the root CA, and I am going to use the already-created web location for it:

```
certutil -setreg CA\CRLPublicationURLs "1: C:\Windows\system32\
CertSrv\CertEnroll\%3%8%9.crl\n2:http://crt.rebeladmin.com/
CertEnroll/%3%8%9.crl\n3:ldap:///CN=%7%8,CN=%2,CN=CDP,CN=Public Key
Services,CN=Services,%6%10"
```

After defining the CDP locations, the next step is to update the AIA location settings.

AIA locations

The AIA locations can be specified using the following command:

```
certutil -setreg CA\CACertPublicationURLs "1: C:\Windows\system32\
CertSrv\CertEnroll\%1_%3%4.crt\n2:http://crt.rebeladmin.com/
CertEnroll/%1_%3%4.crt\n3:ldap:///CN=%7,CN=AIA,CN=Public Key
Services,CN=Services,%6%11"
```

After this, the next step of the configuration is to update the CA and CRL time limits.

CA and CRL time limits

The CA and CRL time limits also need to be adjusted. This can be done using the following commands:

```
certutil -setreg CA\CRLPeriodUnits 7
certutil -setreg CA\CRLPeriod "Days"
certutil -setreg CA\CRLOverlapPeriodUnits 3
certutil -setreg CA\CRLOverlapPeriod "Days"
certutil -setreg CA\CRLDeltaPeriodUnits 0
certutil -setreg ca\ValidityPeriodUnits 3
certutil -setreg ca\ValidityPeriod "Years"
```

Once all this is done, in order to complete the configuration, restart the certificate service using the following command:

```
restart-service certsvc
```

Last but not least, run the following command to generate the CRLs:

```
certutil -crl
```

Once all of this is done, we can run `PKIView.msc` to verify the configuration:

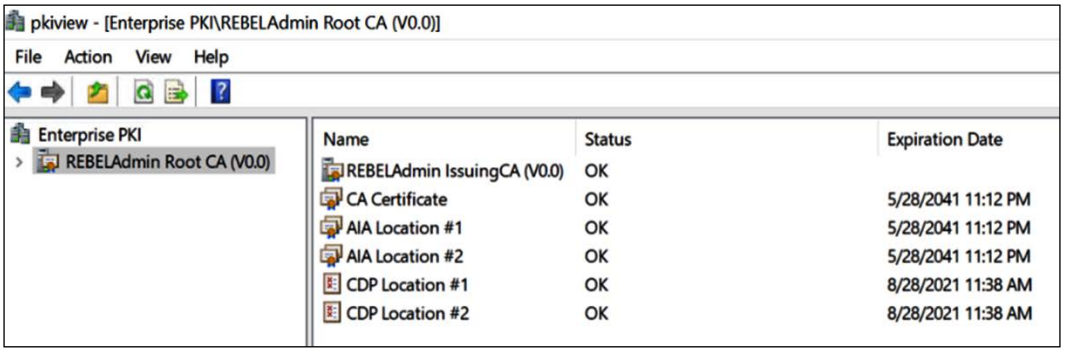


Figure 13.21: PKIView

`PKIView.msc` was first introduced with Windows Server 2003 and provides visibility regarding the enterprise's PKI configuration. It also verifies the certificates and CRL for each CA.

Certificate templates

Now we have a working PKI, and we can turn off the standalone root CA. It should only be brought online if the issuing CA certificates are expired or the PKI is compromised.

The CA comes with predefined certificate templates. These can be used to build custom certificate templates according to the organization's requirements and can be published to AD.

CA certificate templates are available under the Certificate Templates MMC. They can be accessed using **Run | MMC | File | Add/Remove Snap-in... | Certificate Templates**.

1. To create a custom template, right-click on a template and click on **Duplicate Template**:

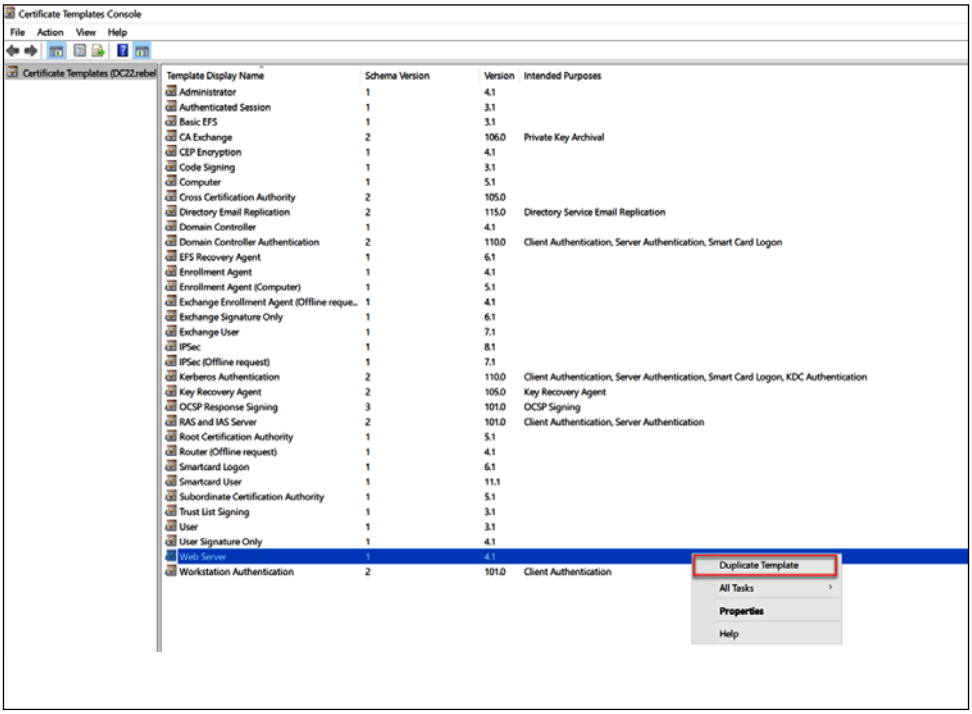


Figure 13.22: Duplicate certificate template

2. This will open up the **Properties** window, where you can change the settings of the certificate template to match the requirements. Some common settings to change in templates are listed here:
    - **Template display name** (the **General** tab): The display name of the template.
    - **Template name** (the **General** tab): The common name of the template.
    - **Validity period** (the **General** tab): The certificate validity period.
    - **Security**: Authenticated users or groups must have Enroll permission to request certificates:

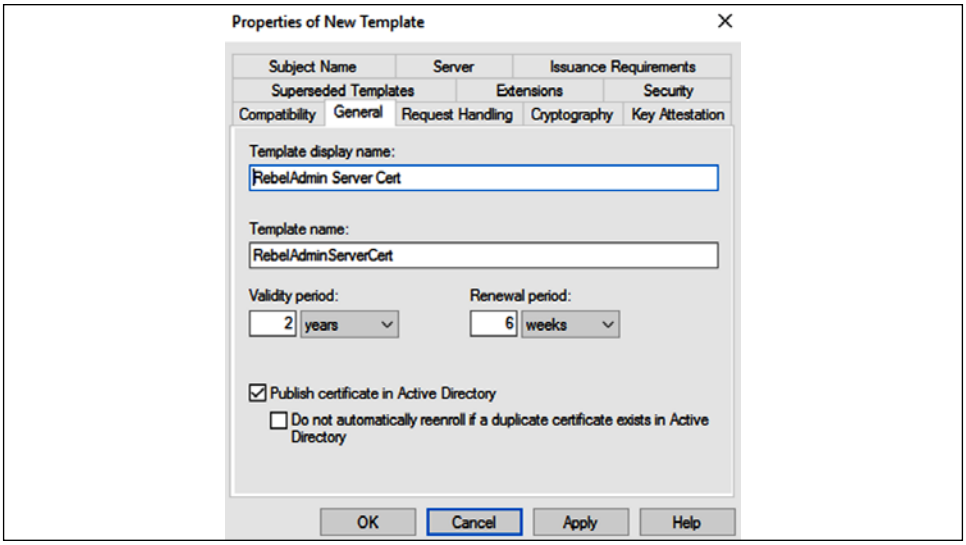


Figure 13.23: Properties of the new template

3. Once the template is ready, we need to issue it. Then, the members of the domain can request certificates based on the new template.
4. To do so, go to **Certification Authority MMC | Certificate Templates** and then right-click and select **New | Certificate Template to Issue**.
5. Then, from the list, select the template to issue and click on **OK**:

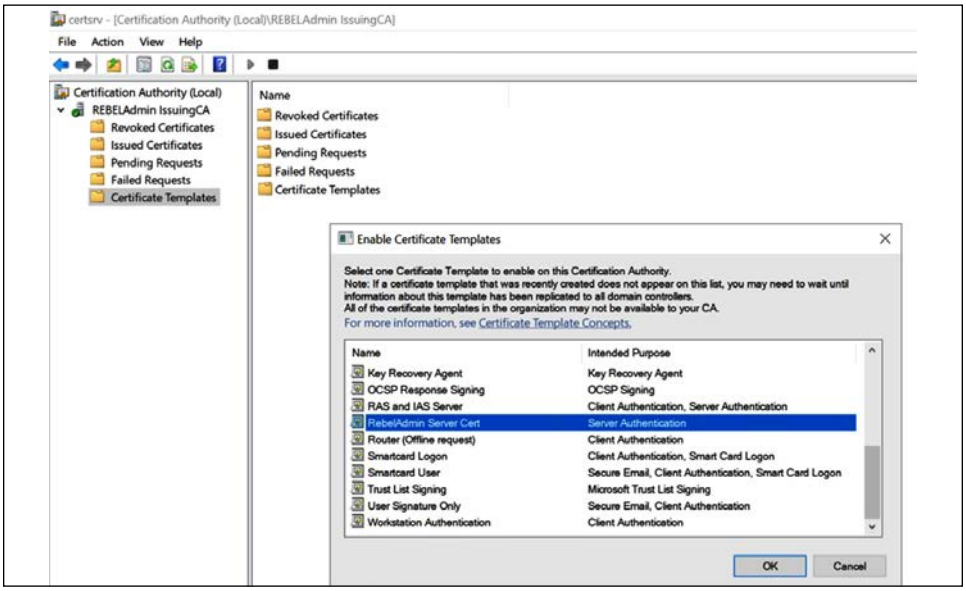


Figure 13.24: Publishing a new template

Now we have the new certificate template. The next step is to request a certificate based on the new template we created.

Requesting certificates

Based on the certificate templates published, users can request certificates from the issuing CA. I have logged in to an end user's PC and I am going to request a certificate based on the template we created in the previous step.

1. To do that, go to **Run**, type `MMC` | **Add/Remove Snap-in... | Certificates**, and click on the **Add** button.
2. From the list, select the computer account. Once selected, in the next window, select **Local computer** as the target.

If the user is not an administrator and only has default permissions, the user will only be allowed to open the `Current User` snap-in. To open the computer account, MMC needs to be `Run as administrator`.

3. Once MMC is loaded, go to the `Personal` container, right-click, and then follow **All Tasks | Request New Certificate**.
4. This will open a new window. Click **Next** until you reach the **Request Certificates** window. In there, we can see the new template. Click on the checkbox to select the certificate template and then click on the link with a warning sign to provide the additional details that are required for the certificate:

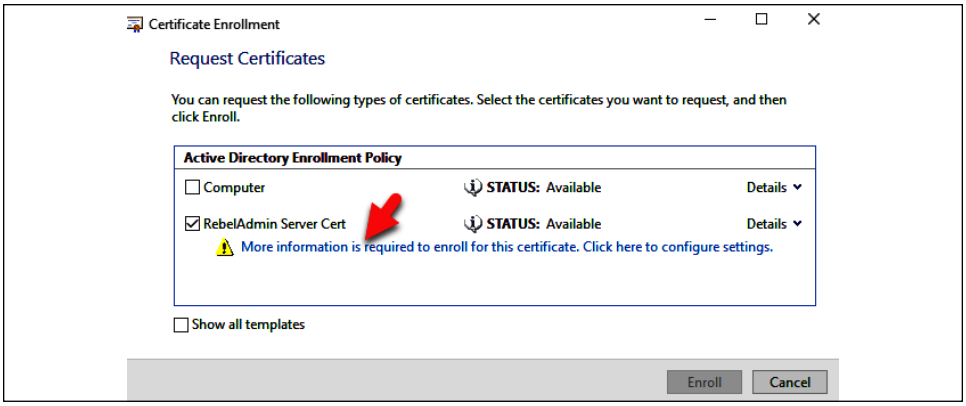


Figure 13.25: Requesting a new template

5. Then, provide values for the required fields and click on **OK** to proceed. Most of the time, **Common name** is what's required:

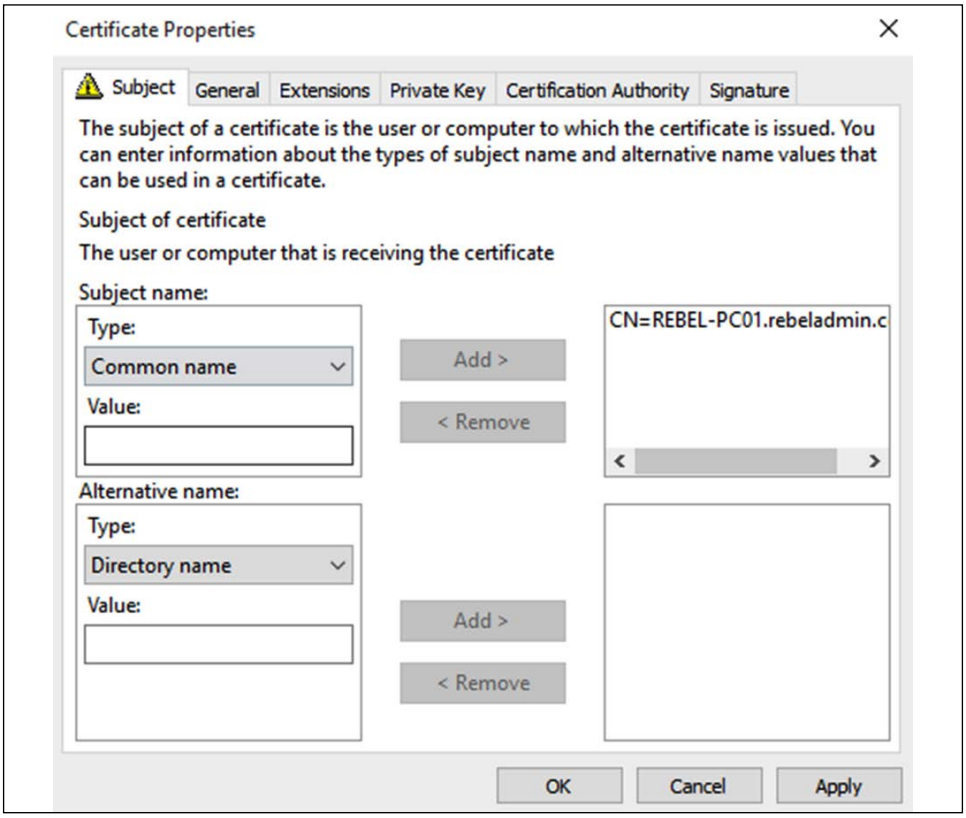


Figure 13.26: Additional information for the certificate request

6. Once that's done, click on **Enroll** to request the certificate. Then, it will automatically process the certificate request and issue the certificate. Once it's issued, it can be found under the **Personal | Certificate** container:

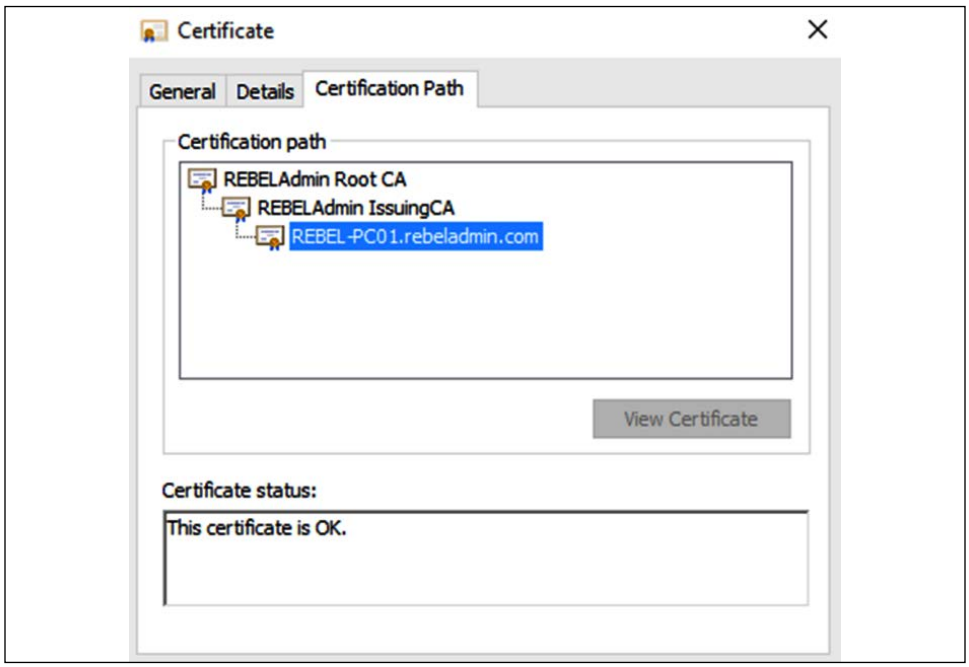


Figure 13.27: Certification path

As expected, we can see that a valid certificate has been issued. Information regarding this issued certificate can be found under **Certification Authority MMC | Issued Certificate** of the issuing CA.

In this exercise, you learned how to set up a two-tier PKI from the ground up. Following the setup, as with any other system, regular maintenance is required to maintain the health of the system. Also, it is important to have proper documentation regarding the setup, along with certificate templates and procedures in order to issue, renew, and revoke certificates successfully.

Migrating AD CS from Windows Server 2008 R2 to Windows Server 2022

Windows Server 2008 R2 extended support ended on January 14, 2020.

This raised interest in migrating various Windows Server roles from Windows Server 2008 R2 to the latest version.

I thought it would be useful if I include steps for migrating AD CS roles from Windows Server 2008 R2 to Windows Server 2022. We also can use the same steps to migrate an AD CS role from Windows Server 2012/2012R2/2016/2019.

Demo setup

The following diagram shows the demo environment that I will be using for this particular task:

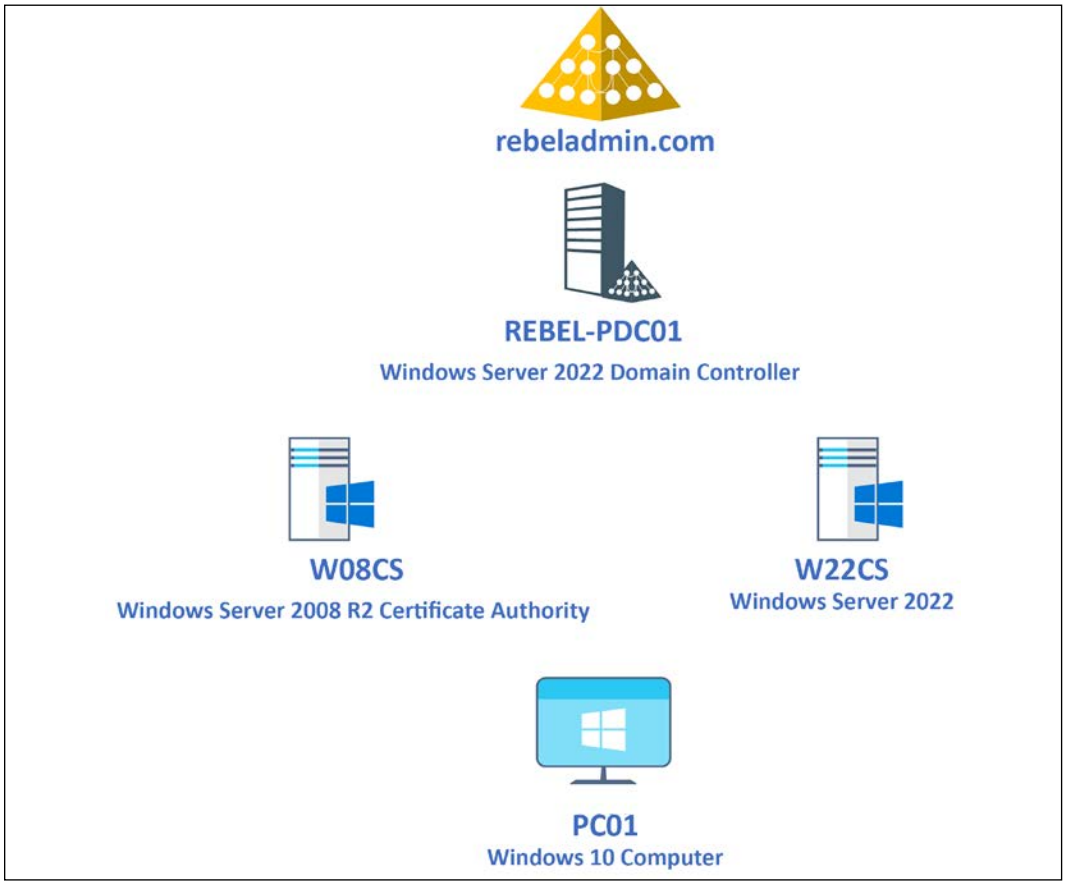


Figure 13.28: Demo environment

As illustrated in the preceding diagram, in the demo environment, I have 4 servers/PCs. The role of each server/PC is as follows:

Host Name	Operating System	Role
REBEL-PDC01	Windows Server 2022	Primary Domain Controller in rebeladmin.com Active Directory Domain.
W08CS	Windows Server 2008 R2	Existing CA.
W22CS	Windows Server 2022	After AD CS configuration is migrated, this server will become the CA in the rebeladmin.com domain.
PC01	Windows 10	Test PC.

Here, the plan will be to migrate the AD CS configuration from the existing Windows Server 2008 R2-based CA to the newly built server with Windows Server 2022. For the purposes of the demo, the current CA is deployed using a single-tier model, which means a single server will work as the root CA as well as the issuing CA. The AD CS migration process primarily comprises the following steps:

1. Backing up the configuration of the existing CA
2. Removing the AD CS role from Windows 2008 R2 Server
3. Installing an AD CS role in the new Windows 2022 Server
4. Restoring the configuration from the previous CA
5. Testing

Let's go ahead and start the process by exporting the current CA configuration from the W08CS server.

Backing up the configuration of the existing CA (Windows Server 2008 R2)

There are two ways of exporting the CA database and private key from the Windows Server 2008 R2 CA:

1. By using the `certutil` command utility
2. By using the CA mmc

From Windows Server 2012, we also can use the **Backup-CARoleService** PowerShell cmdlet to back up the existing CA configuration.

In this demo, I am going to use the **certutil** command utility to back up up the CA. To do this, perform the following steps:

1. Log in to the Windows 2008 R2 CA Server as Domain Administrator.
2. Launch PowerShell as Administrator.
3. Run `certutil -backup C:\CABackup`:

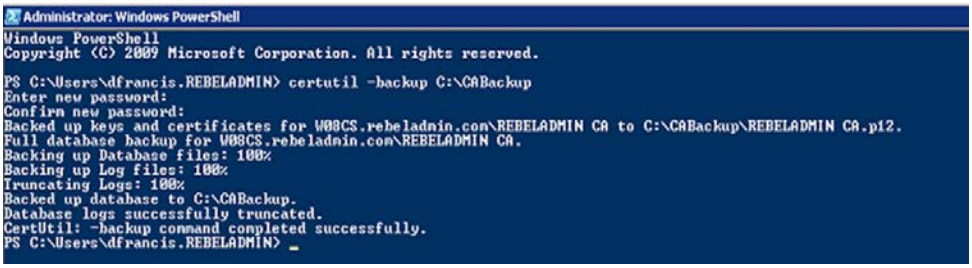


Figure 13.29: Backup CA configuration

The preceding command will back up the following items to the `C:\CABackup` folder:

- Certificate database
- Certificate database log files
- CA certificate and private key:

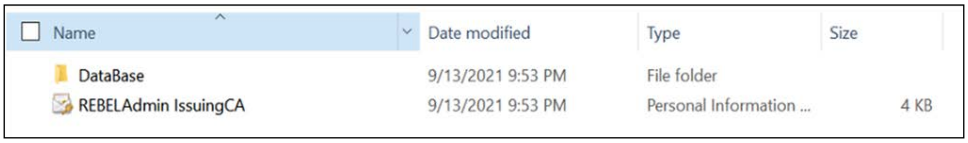


Figure 13.30: CA backup content

4. We also need to export the CA configuration settings saved under the **HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CertSvc\Configuration** registry key. To export the key, run the following PowerShell command:

```
reg.exe export HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\
Services\CertSvc\Configuration C:\CABackup\careg.reg
```

5. The preceding command will back up the registry key to the **C:\CABackup** folder and save it as `careg.reg`.

Now that we have the relevant backup in place, before we import it to the new server, we need to remove the AD CS role from the existing CA server. Remove the AD CS role from Windows 2008 R2 Server.

To remove the AD CS role, perform the following steps:

1. Log in to the existing CA as Domain Admin.
2. Launch PowerShell console as Administrator.
3. Run the `Remove-WindowsFeature -Name AD-Certificate` command. This will remove the AD CS role from the server. Once the role is removed, restart the server to complete the process.
4. Note: You may have to run `Import-Module Servermanager` first before you use WindowsFeature PowerShell commands.
5. Copy the `C:\CABackup` folder to the new Windows 2022 Server.

After copying the backup folder, we can also shut down and remove the old CA server from the domain.

Installing an AD CS role in the new Windows 2022 Server

The next step of the configuration is to install an AD CS role in the new Windows 2022 Server (W22CS). To do that, perform the following steps:

1. Log in to W22CS Server as Domain Administrator.
2. Launch PowerShell 7.1 console as Administrator (I have PowerShell 7.1 configured already).
3. Run the `Add-WindowsFeature ADCS-Cert-Authority -IncludeManagementTools` command to install the AD CS role.

This will install the AD CS role on the server. However, we need to restore the configuration before using it.

Restoring the configuration from the previous CA

Now we have the AD CS role installed on the new Windows 2022 Server and can restore the AD CS backup by using one of the following methods:

- By using the `Restore-CARoleService` PowerShell cmdlet
- By using the `Certutil` command utility
- By using the `CA mmc`

In this demo, I am going to restore the backup by using the `Restore-CARoleService PowerShell` cmdlet. But before that, we first need to configure the AD CS role with an existing certificate. To do that, perform the following steps:

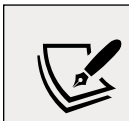
1. Log in to W22CS Server as Domain Administrator.
2. Launch PowerShell 7.1 console as Administrator (I have PowerShell 7.1 configured already).
3. Run `Install-AdcsCertificationAuthority -CAType EnterpriseRootCa -CertFile "C:\CABackup\REBELADMIN CA.p12" -CertFilePassword (read-host "Cert Password" -assecurestring)`:

Figure 13.31: Configuring an AD CS role with the existing CA certificate

The preceding command configures the AD CS role with the existing CA certificate, which is saved as **C:\CABackup\REBELADMIN CA.p12** from the previous CA backup.

4. As the next step, we can restore the CA configuration by using the `Restore-CARoleService PowerShell` cmdlet. Before we use it, first we need to stop the AD CS using `Stop-Service CertSvc`.
5. After that, we can restore the CA database by using `Restore-CARoleService -Path C:\CABackup -DatabaseOnly -force`.
6. The preceding command will restore just the CA database. We do not need to restore the key as we have already done that.
7. Once the preceding step is complete, we can start the AD CS service using `Start-Service CertSvc`.
8. The next step of the configuration is to restore the registry settings related to the AD CS service. To do that, double-click on the registry file, which is saved as `C:\CABackup\careg.reg`.
9. Once the key is imported, restart the server.

This completes the CA migration process. The next step will be to do some testing to verify the configuration.



It is not required to rename the new CA server name with the old one. As long as the relevant DNS records (for URLs) are in place, the CA will work.

Testing

In the demo environment, we are using a Windows 10 computer (PC01). This computer already has a certificate issued by the previous Windows Server 2008 R2 CA. I went ahead and logged in to PC01 and opened the mmc certificate. In there, I can see the existing certificate, and it shows as a valid certificate:

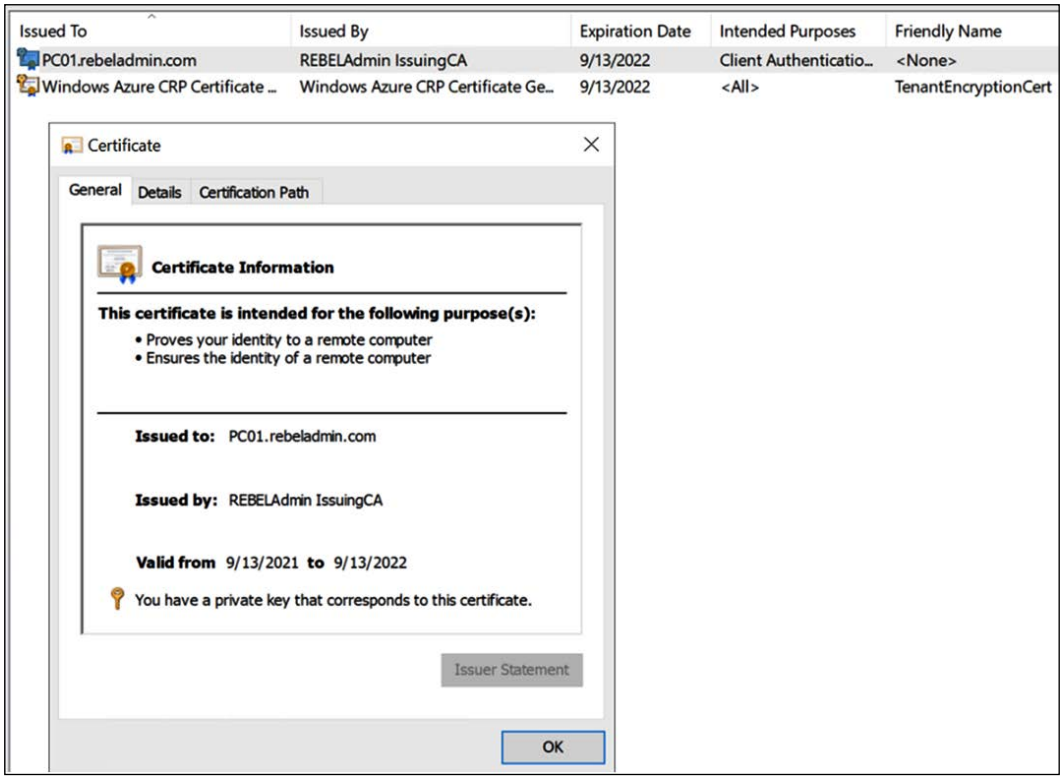


Figure 13.32: Existing certificate

I also checked the certificates under **Certificate Authority | Issued Certificates** and I can see the same certificate there. This means that the previous CA configuration has been migrated to the new CA.

In Windows Server 2008 R2 CA, I had a custom certificate template created for computers. From PC01, I went ahead and requested a new computer certificate. As expected, I can still see the certificate template I created previously:

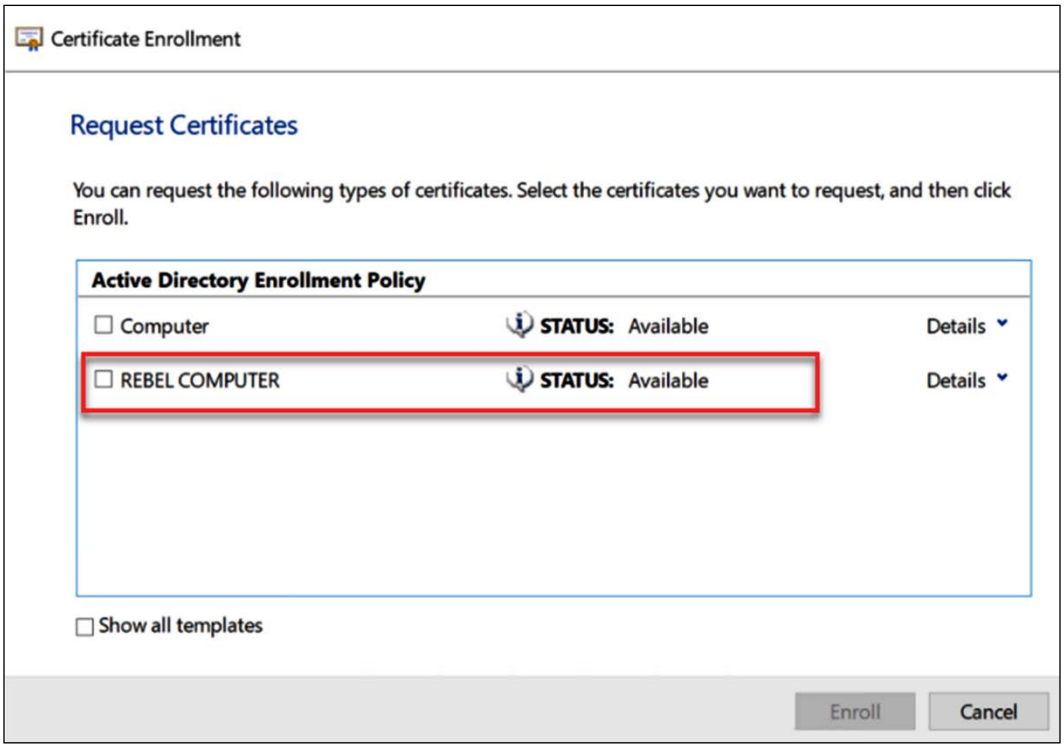


Figure 13.33: Requesting a new certificate

I went ahead and requested a certificate with the custom certificate template and I was able to get a new certificate. This means that the old certificate templates have also been migrated.

After completing the enrollment process, I went to the **Certificate Authority | Issued certificate** mmc in the W19CS server. There, I can see the newly issued certificate:

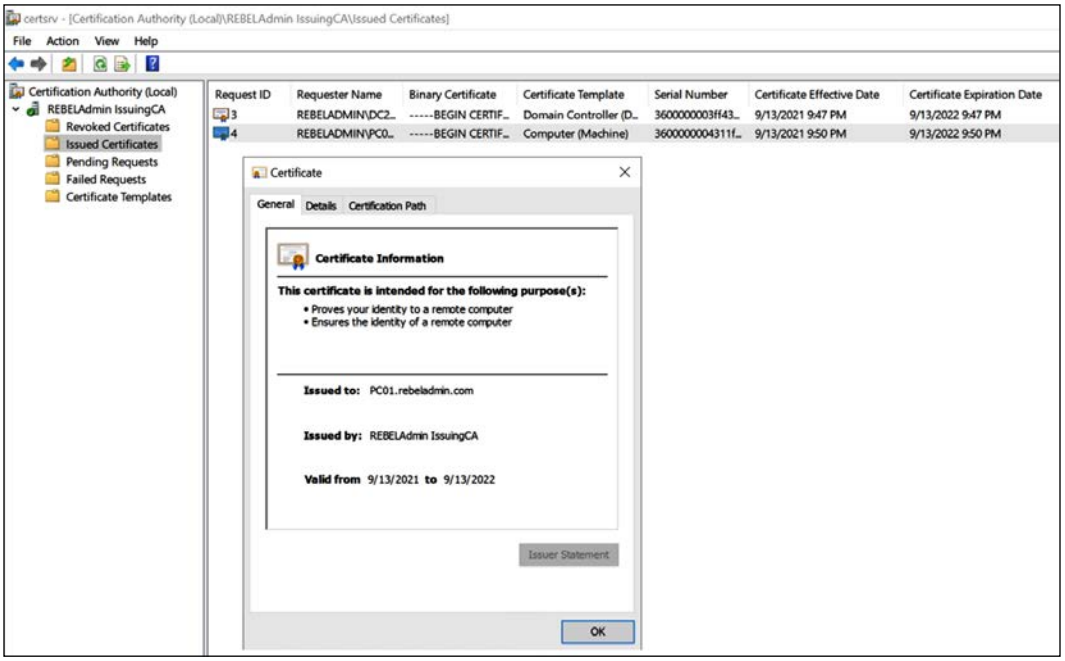


Figure 13.34: New certificate from the Windows Server 2022 CA

As we can see here, the new CA is working as expected and already has the previous CA configuration. This completes the AD CS migration process and, in the next section, we are going to look at CA backup and **disaster recovery** (**DR**).

AD CS disaster recovery

Like any other computer system, CA also may face disasters due to operating system issues, hardware failures, and so on. Depending on the CA's role in the infrastructure, we can plan how to recover from such failure. There are certain things we need to consider when developing a disaster recovery plan for the CA.

- **Role of the CA**: The importance of the CA in infrastructure has a major impact on the disaster recovery plan. If an organization is using certificates for day-to-day activities such as computer authentication and Wi-Fi connection, this means the CA's availability is crucial for carrying on operations.

  On the other hand, if the CA is only used to issue certificates for a few internal applications or systems that are renewed every few years, CA availability is not as crucial as in the first scenario.

In the event of a disaster, even a redeployment of the CA may not have a huge impact. As we can see, the role of the CA will decide how much investment we need to make in relation to disaster recovery and what optimal **Recovery Point Objective** (**RPO**) and **Recovery Time Objective** (**RTO**) values are.

- **Investment**: In the event of a disaster, some DR solutions such as Azure Site Recovery can bring critical servers up and running in completely different geographical locations within minutes. However, the investment and the selection of technology for the DR solution depends on the RTO and RPO values. If the CA is mission-critical, we need to maintain lower RTO and RPO values. In an ideal world, both values should be closer to zero as far as possible, but that's going to be expensive. To maintain lower RPO and RTO values most of the time, we have to use replication services. This is generally more expensive than typical backup solutions. If the CA can afford a longer downtime, such as 24 hours, a daily backup or snapshot will be sufficient and the cost of such a solution is lower compared to real-time replication.

The RPO value explains the frequency of the backup. In the event of failure, how much data can we afford to lose? Is it 5 minutes or a few hours?

The RTO value explains how long it will take to recover from a disaster.

- **Documentation**: In the worst-case scenario, if the CA cannot be recovered, we will have to rebuild the CA. To do that, we need to know the current configuration of the CA. It is important to document the current configuration of the CA. It should include information such as the following:
    1. The purpose of the CA
    2. Topology
    3. Certificate template details
    4. CA permissions

Apart from the CA documentation, we also need to have documentation that covers the recovery process of the CA. This should be as detailed as possible as it will help engineers to avoid unnecessary problems and delays during the recovery process.

Disaster recovery methods

Many different products and services can be used to replicate or back up the CA.

I am not going to talk about these different products here, but to recover the CA using a backup, we require the following:

1. A certificate database
2. Certificate database log files
3. A CA certificate and private key
4. CA registry configuration

There are three methods we can use to back up the preceding data:

1. System state backup
2. The certutil command utility + Registry Export
3. The Backup-CARoleService PowerShell cmdlet + Registry Export

Let's look at each of these methods in turn.

System state backup

System state backup includes operating system files and registry data. If a system loses its system files or registry data in the event of a disaster, by using system state backup, we can recover the system. If the system is a CA, system state backup will also include a certificate database and other role configuration data. We can take a system state backup by using a native Windows backup or any other backup solution, including Azure Backup, Commvault, and Veeam. However, to use system state backup, the computer should still start, but on occasion, it may not start at all. To recover from such a situation, we can use the "bare-metal" backup option instead of system state backup. This method will allow you to restore the complete system in a like-for-like server. The new server should use the same make, model, and configuration. The bare-metal backup method is also available on many other solutions, such as Azure Backup, System Centre **Data Protection Manager** (**DPM**), and Veeam.

The certutil command utility + Registry Export

Certutil is a command-line utility that is installed as part of certificate services. This utility can be used to configure the CA, view the existing configuration of the CA, and back up/restore CA components. To back up the CA configuration by using certutil, we have to use `certutil -backup C:\folderpath`.

This backup will include the following:

- A certificate database
- Certificate database log files
- A CA certificate and private key

To restore from an existing backup, we can use the `certutil -restore C:\folderpath` command. However, this will not include the CA configuration settings saved in `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CertSvc\Configuration`. We need to export it as separate file. To do that, we can use the `reg.exe export HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CertSvc\Configuration C:\folderpath\cabackup.reg` command.

The Backup-CARoleService PowerShell cmdlet + Registry Export

The ADCSAdministration Windows PowerShell module was first introduced with Windows Server 2012, but with Windows Server 2012 R2, this module has two new cmdlets that can be used to back up and restore the CA:

- Backup-CARoleService
- Restore-CARoleService

To backup the CA configuration, we can run the `Backup-CARoleService C:\CABackup` PowerShell command. Here, `C:\CABackup` is the backup folder path:

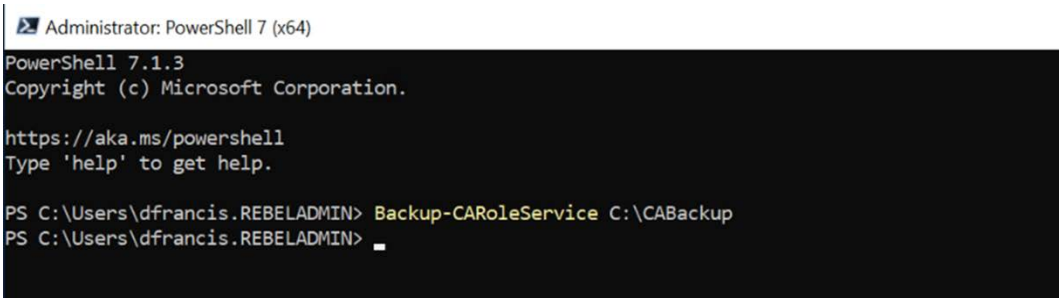


Figure 13.35: Backup CA configuration

This backup includes the following:

- A certificate database
- Certificate database log files

- A CA certificate and private key:

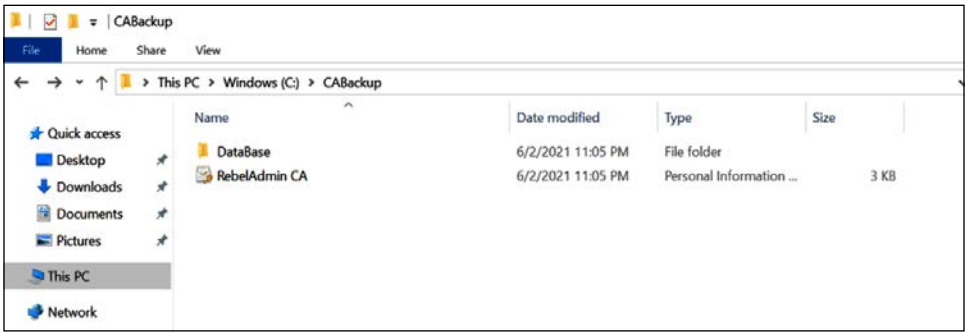


Figure 13.36: File in the CABackup location

Even with this method, we still need to export the registry configuration as a separate file.

To restore the CA configuration, we can use the `Restore-CARoleService C:\CABackup -force` PowerShell command.

Note: We need to stop the AD CS role service prior to the restore process. Also, if you do not use the `-force` option, you will receive a `Restore-CARoleService: The directory is not empty. (Exception from HRESULT: 0x80070091)` error.

Both the command-line and PowerShell backup methods can be automated using scripts. After selecting the backup method, perform the following steps:

1. Run a test backup and restore before scheduling it. That way, you know precisely whether it's working and how long it takes for the restore process.
2. Create a DR plan and include all the recovery steps in detail to avoid unnecessary delays in the restore process.
3. Run a DR test at least once a year to validate the solution in place. We do not need surprises in the actual DR situation.

For DR testing, please make sure to use a completely isolated environment that doesn't communicate with production servers.

Summary

Digital certificates are increasingly used in modern infrastructure as additional layers of security to prove that objects and services are genuine. In this chapter, you learned what a PKI is and how it works exactly. Then, we looked into AD CS components and their responsibilities.

After that, we moved into the planning of a PKI and discussed what needs to be considered when building it. Then, we looked into PKI deployment models and evaluated their pros and cons. Later, we went through a step-by-step guide to setting up a two-tier PKI. Windows Server 2008 is out of support now and it is important to know how we can migrate the CA configuration from Windows Server 2008 to Windows Server 2022. This scenario has also been covered in this chapter. Last but not least, we learned how to recover the CA from a disaster.

In the next chapter, you are going to learn about another AD role service—AD Federation Service—and see how identities are handled in a federated environment.

14

Active Directory Federation Services

The COVID-19 pandemic has accelerated the digital transformation of businesses. Most businesses no longer operate in a closed or isolated mode. With digital transformation, they are collaborating more with other companies, partners, and consumers to provide better products or services. This also creates new challenges for IT to accommodate new collaboration requirements. As an example, a business might need to share one of its applications with another external company. Or, a business might want to share resources (such as access to certain servers or data shares) with a partner company. In such situations, the question is how to manage user accounts and access permissions in a secure, reliable, and scalable way.

In an **Active Directory** (**AD**) environment, most applications or services can be Active Directory-integrated. This means we can use Active Directory accounts to authenticate into applications or services. But what if we need to access applications or services hosted in a different environment where we do not have control over access management? In such situations, we usually end up having different user accounts and passwords to log in to different systems. This does not just affect end users. Let's look at the problem from the service provider's perspective.

Imagine that we have an application developed in-house, and we want to sell it as a service. External users need to access it in order to authenticate, and we have to create a user name and password for every one of them. Setting up an account is not the only thing we need to consider; when we create an account, it becomes a part of our identity infrastructure. We need to make sure it's secured and only has access to that particular application.

All of a sudden, new challenges arise for identity management, and if this isn't handled appropriately, it can make the whole system vulnerable. **Active Directory Federation Services** (**AD FS**) allows businesses to manage their own identity infrastructures and use claims-based authentication to access applications, services, or resources. With this method, users do not need to use a separate login to access systems, and the resource owners do not need to manage identities for external users. In this chapter, we are going to learn about the following:

- How does AD FS work?
- AD FS components and how to use them in the AD FS setup
- AD FS deployment and management
- **Multi-Factor Authentication** (**MFA**) in action
- How to enable federation with Azure AD
- Azure AD federation with AD FS

Before we look into AD FS role deployment and configuration, it is important to understand how AD FS works. This will help us to understand the full potential of AD FS role deployment.

How does AD FS work?

Rebeladmin Inc. is an IT service provider. There are many customers who use different IT and cloud-based services from the company. Recently, the company introduced a new web-based control panel where customers can log in and manage their resources. The same application is also used by internal staff to manage infrastructure services. Rebeladmin Inc. uses **Active Directory Domain Services** (**AD DS**) to manage identities. When a member of internal IT staff logs in to the portal, it doesn't ask for any login details. This is because the web application uses **Integrated Windows Authentication** (**IWA**) to allow access.

This process is also called **NTLM authentication** or **domain authentication**. It doesn't prompt for the login information initially, or transfer hashed data about the currently logged-in user to the web server to check whether it's allowed. This web server is domain joined and the application itself is Active Directory-integrated. Now, users of the Depache solution also like to get access to the same portal to manage their workloads.

There are two ways to facilitate this:

- **Using a user account in Rebeladmin Inc.'s Active Directory**: When external users try to access the web portal, the initial IWF will fail as the application doesn't understand the external users' accounts.

Then, it will prompt them for their login details. If a user has an account in Rebeladmin Inc.'s Active Directory instance, it can be used to authenticate into the portal. However, this method opens up a few security concerns. When users have an account in Active Directory, by default, Active Directory allows users to access any resources that have *everyone* or *authenticated users* permissions assigned. In the internal network, it is possible to force users to follow policies and best practices to protect their identities. However, it is not possible to apply the same standards to an external party. So, these accounts have a high likelihood of getting compromised. As an example, if an internal user resigns, then normally, their Active Directory login will be reset and disabled. But if it's an account that is shared with external users, then even if the relevant user resigns, they may still have access to the portal (until it is informed). Some vendors use **Active Directory Lightweight Directory Services** (**AD LDS**) for each customer to minimize the security impact. But this still adds management overhead to keep different instances running.

- **Active Directory trust between two infrastructures**: When there is Active Directory trust, resource access can be allowed from remote infrastructures. In order to have successful trust, there should be a connection between two infrastructures, which is based on TCP/UDP ports such as `389` (**Lightweight Directory Access Protocol** (**LDAP**)) and `53` (**Domain Name System** (**DNS**)). These ports need to be allowed via firewalls in both infrastructures. This method adds additional security risks for both infrastructures.

As we can see, even though both options can allow *access* from the external infrastructures, both struggle with security and management-related challenges.

Federation trust is the answer to all these concerns. In simple English, a federation service is a web service that authenticates users from the **Identity Provider** (**IdP**) and provides access to claim-based applications from the **Service Provider** (**SP**). There are many federation service providers and the Microsoft federation service is called **Active Directory Federation Services** (**AD FS**):

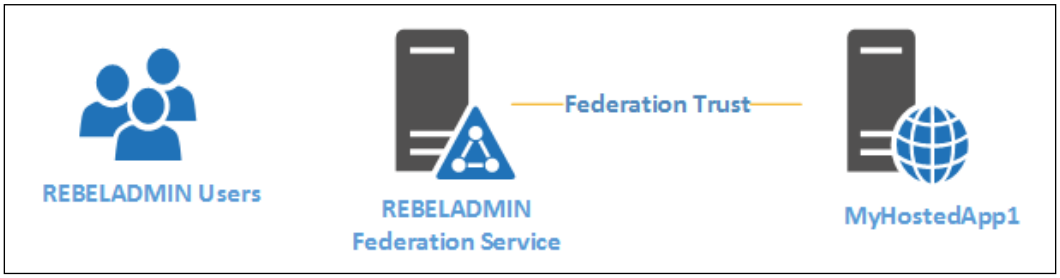


Figure 14.1: Federation trust with a hosted app

In the preceding example, Rebeladmin Inc. is using AD DS to manage the identities. Its users are using a hosted web application (**MyHostedApp1**) from a cloud service provider. Rebeladmin Inc. uses AD FS to create federation trust between Rebeladmin Inc. and the SP. This allows the Rebeladmin Inc. users to launch **MyHostedApp1** using their own Windows credentials. In this setup, the federation service is hosted in the Rebeladmin Inc. infrastructure and it becomes the **IdP**. From AD FS' point of view, it is also called the **Claims Provider** (**CP**). The users are authenticated by the **CP**. The vendors who host the application also have their own identity infrastructure. The application vendor becomes the SP, also known as the **Relying Party** (**RP**), in federation trust, and this depends on the claims provided by the federation service to allow/deny access to the application. Also, this setup only needs TCP port `443` opened.

Even if the connection is called trust, it cannot be used to replace Active Directory domains or forest trusts. If Active Directory trust is in place, then administrators can manage access to resources from remote users in the exact same way it was done for internal users. Users can be allowed to access folders and files based on **New Technology File System** (**NTFS**) permissions. Users can be given permission to log in to devices. Any application that works with internal users can be set to allow access for remote users too. But in a federated environment, access can only be allowed to *claims-aware* applications. A claim is simply an attribute and a relative value. As an example, a claim can have a `username` attribute and its value, `dfrancis`. The federation service will request access from the SP based on the claims. If the SP's application doesn't understand *claims*, it cannot decide whether to allow or deny access:

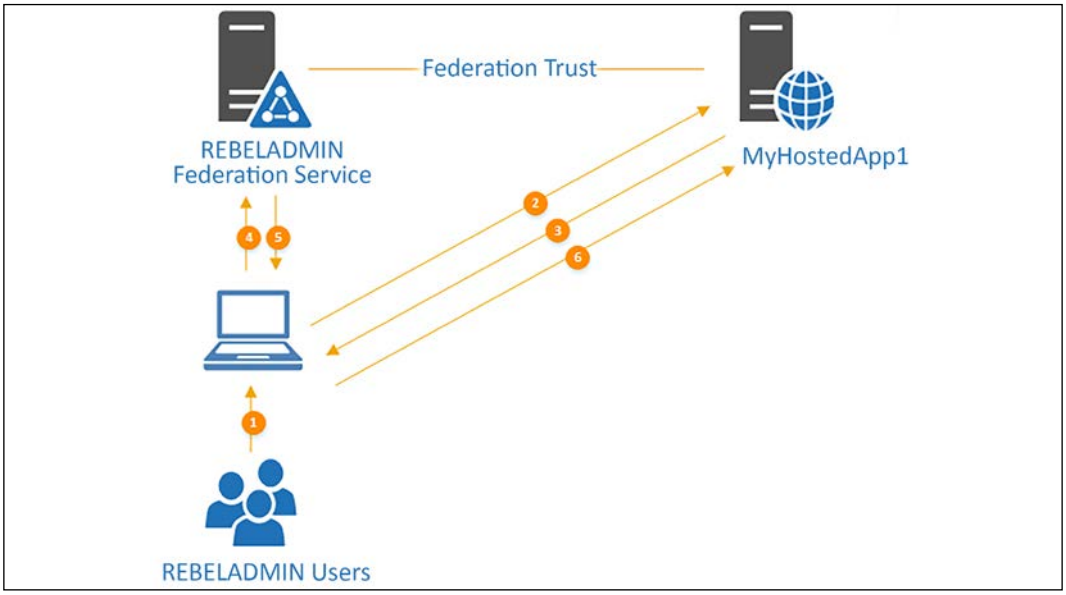


Figure 14.2: Federation trust in action

Let's revisit the previous example with a detailed explanation,

1. In the first step, the Rebeladmin Inc. user logs in to a computer in the Rebeladmin Inc. infrastructure. This is a domain-joined device. The user uses their domain credentials to log in to the device.
2. Once the user is logged into the device successfully, they launch their web browser and type the URL for **MyHostedApp1**. This application is hosted in the remote infrastructure.
3. As soon as the application receives the access request from the Rebeladmin Inc. user, it first checks the data passing through the browser (IWF). Then, it realizes it is not from the internal infrastructure, but it's an account from the identity infrastructure, which the SP has a federated relationship with. If it's from a federated environment, the user access rights need to be decided based on claims. Therefore, the SP sends a redirect response to redirect users to the Rebeladmin Inc. federation service web interface.
4. Then, the user is automatically redirected to the Rebeladmin Inc. federation service web interface. In the web interface, the user has to type in their login details.
5. Once the user provides their credentials, they will be validated with the Rebeladmin Inc. AD DS and it will retrieve their access rights. Then, the federation service will create a security token that includes user claims, such as name, group memberships, **User Principal Name** (**UPN**), and email address. This security token will be signed by the issuer's digital certificate. At the end of the process, federation services send responses back to the user with a generated security token, plus a `redirect` command to redirect the user's browser session back to **MyHostedApp1**.
6. The user browser session is redirected to the hosted application, and this time, it is the session presented with the security token it received from the federation service. Then, the web server's claims-aware agent decrypts the security token and looks into the claims it provided. Based on the claims, it will decide whether the user is allowed to access the application or not. If the claims are accepted, the user will be able to see the initial page of the application successfully.

In the preceding scenario, the federation trust is made directly with the application. But in some scenarios, the SP will also have the federation service environment on its end. This is mainly when the service provider has multiple clients with federation trusts. In such a situation, the following will happen:

1. When the application finds out that the user needs to log in via the federation, the user will initially be redirected to the SP's federation service.

2. Then, the SP's federation service will redirect the user to Rebeladmin's federation service.
3. Once the user receives the security token, the security token will be presented to the SP's federation service. It will decrypt the token and retrieve the claims inside it. Then, it will map the claims to the SP's claims and filter them to match with application claim requirements.
4. Based on the filtered claims, the SP's federation service will create a new security token, which will be signed using the SP's federation service digital certificate. This new security token is the one that will be forwarded to the application server.

A few times in the preceding example, I have mentioned the term "claims." But what exactly is a claim, and how is one generated?

What is a claim?

A claim is simply a statement about a user that is used for the authorization purposes of claim-aware applications. Each claim contains a value about a user such as their **UPN**, **email address**, and **Common Name** (**CN**).

AD FS supports many different claim types. Claim types are used to show what sort of value will be included in the claim. The following table contains the most commonly used claim types:

Claim type	**Description**
UPN	UPN of the user
Email	RFC 5322-type email address
Given name	Given name of the user
CN	CN value of the user account
Name	Name of the user
Surname	Surname of the user
Windows account name	Domain account in domain/user format
Group	Group the user belongs to
Role	Role of the user
AD FS 1.x UPN	UPN of the user when interacting with AD FS 1.x
AD FS 1.x email address	RFC 5322-type email address of the user when interacting with AD FS 1.x

Claims retrieve values from the attribute store. The attribute store is a directory or database that contains user accounts and associated attributes.

Active Directory can also play the role of an attribute store. As an example, in an Active Directory environment, if the claim type is UPN, the claim will retrieve its value through users' attributes.

AD FS also supports many industry standards, which are used to build third-party claim-based solutions. It guarantees the interoperability of many cloud-based or hosted applications in the market today.

Security Assertion Markup Language (SAML)

In the federated environment, the **IdP** and the **SP** need to exchange authentication and authorization data. SAML is an XML-based open standard that is used to pass authorization credentials from IdP to SP. This standard was first introduced in 2001 by the **OASIS Security Services Technical Committee** and the latest version available is 2.0. This is a commonly used standard by many federation service providers and application developers to provide an SSO experience. The claim requesting and claim processing process is exactly the same as the example used in the previous section, with the only difference being the format of the token request and response. SAML uses a signed XML file as the token. In SAML terminology, the security tokens generated at the IdP end are called **asserts**, and the decryption and processing of asserts at the SP end is called **assertion**.

WS-Trust

This is part of many WS* standards, including WS-Security, WS-Federation, and the WS-Security policy. **WS** stands for **Web Services**. WS-Trust defines the protocols used in requesting and issuing security tokens by WS-Security. The **Security Token Service** (**STS**) is a big feature of WS-Trust, and it can be used to convert locally issued security tokens into other security token formats that can be processed by the application. It can also convert incoming security tokens into supported token formats.

WS-Federation

WS-Federation is also a part of WS* standards. While SAML only works with SAML tokens, WS-Federation supports the use of many token types, including SAML. Basically, WS-Federation provides a mechanism to simplify the communication between an IdP and an SP. The fundamental goal of WS-Federation is to simplify the development of federated services through the cross-realm communication and management of federation services by reusing the WS-Trust STS model and protocol. More information about WS-Federation can be found at `https://ibm.co/3CGyRbO`.

AD FS components

Before we install the AD FS role, there are a few related components that we need to be aware of. Before Windows Server 2012 R2, there were four AD FS role services: the federation service, the federation service proxy, the claim-aware agent, and the Windows token-based agent (which supported AD FS 1.x interoperability). These are no longer available as role services, and when we go to install AD FS, it will only have the federation service role.

Federation service

This is the main role service for AD FS, and it can work at the IdP end as well as the SP end. In order to install the AD FS role service, the system needs to be a member server of an Active Directory domain. Depending on the workload, multiple federation servers can be installed under the same domain, and this is called an **AD FS farm**. The federation server is responsible for generating security tokens and signing them with its signing certificate. Let's look into the AD FS versions that have been released so far.

AD FS 1.0

AD FS was first introduced with Windows Server 2003 R2. This version of AD FS is no longer available as Windows Server 2003 is end of life. This provided basic SSO capabilities. It had less compatibility with other federation service providers in the market.

AD FS 1.1

This was introduced with Windows 2008, and it continued with Windows Server 2008 R2. It wasn't changed much from version 1.0. It also suffered from providing limited support to other federation services. It only supported the WS-Federation passive requester profile and SAML 1.0.

AD FS 2.0

This version was released after Windows Server 2008 R2 but as a separate installation. All other versions came as part of the OS. Before version 2.0, it was supported to use AD LDS as the authentication store. This meant users could authenticate with AD LDS, similar to Active Directory. With version 2.0, it no longer supports LDS as the account store. It can work as the attribute store, which can store AD FS data but cannot be used for authentication. AD FS 2.0 also supports a parent-child domain environment, so users in a child domain can use AD FS in another domain for the federation.

It reduces the management overhead. It also improved support for federation trusts with the use of industry-standard metadata formats. It allows organizations to create trust between federation partners quickly. Systems that run with version 1.x can have an in-place upgrade to 2.0.

AD FS 2.1

This version comes with Windows Server 2012 and no major changes from version 2.0 were made.

AD FS 3.0

This version was introduced with Windows Server 2012 R2. This removed the AD FS Proxy service from AD FS 2.0 and replaced it with **Web Application Proxy** (**WAP**). It operates from the **Demilitarized Zone** (**DMZ**) and doesn't need to be domain joined. The idea of it is to protect the identity infrastructure with a bogus token. It also removed the AD FS 1.x web agents, which provided connections with other systems.

Workspace Join is one of the greatest features that came with AD FS 3.0. It allows us to register mobile devices (even non-Windows) with a company or organization to access applications and data with SSO. AD FS 3.0 does not require **Internet Information Services** (**IIS**) anymore and is installed as a separate role. It also supports the **group Managed Service Account** (**gMSA**). This is a new type of service account that supports automatic password changes. The creation and management of this account are explained in *Chapter 8*, *Managing Users, Groups, and Devices*. This version also supports OAuth 2.0 standard access tokens. They are JSON format tokens and are easy to use with modern applications.

AD FS 4.0

AD FS 4.0 is the latest version available with Windows Server 2016/2019/2022. This is what we will use throughout this chapter. This version is supported by modern hybrid cloud requirements. If you are already using Azure AD, this version allows you to use Microsoft Azure MFA without installing and configuring additional components. With previous AD FS versions, it needed an additional server to configure. With the new version, AD FS has a built-in Azure MFA adapter. Similar to AD FS 3.0, the new version also supports mobile device registration to maintain an organization's compliance requirements. If it's in an Azure AD environment, then by using AD FS 4.0, you can apply conditional access policies to on-prem components.

This version also supports modern authentication standards, such as OpenID Connect and OAuth 2.0.

It provides an enhanced user experience with Windows 10 and the latest Android and iOS apps. AD FS 4.0 also supports authentication with LDAP v3.0-compliant directories. It allows people to use AD FS more and more, even when they are not running AD DS. Windows 10 introduced the new password-less login methods Windows Hello and Microsoft Passport. These are based on PIN and biometric input, such as fingerprint or facial recognition. AD FS 4.0 supports these new sign-in methods.

Migration from AD FS 2012 R2 has been simplified as well. Before, if we needed to migrate from one version to another, then we needed to build a farm parallel to a production AD FS farm, and then migrate the configuration over. But, with the new version, we can introduce the AD FS 2022 server to the existing Windows Server 2012 R2 farm, and it will start to work at the Windows Server 2012 R2 operation level. Once all the Windows Server 2012 R2 servers are removed from the farm, the operation level can upgrade to Windows Server 2022.

What is new in AD FS 2022?

As far as we know, AD DS 2022 doesn't have any significant changes from AD DS 2019. There is no new forest or domain functional level either. The same applies to AD FS 2022 as well. Below I have listed some enhancement AD FS 2019 had. This also applies to AD FS 2022:

- **Primary authentication via third-party authentication providers**: When authenticating via AD FS, so far, we have only been able to use built-in authentication methods such as form authentication, certificate authentication, or Azure MFA as primary authentication. Then, as secondary authentication, we were able to use any other external authentication methods. With AD FS 2022, we can now use any third-party authentication provider's login method as primary authentication (the provider needs to register with the AD FS farm first).
- **Password-less authentication**: AD FS 2022 is fully supported to use password-less authentication as a primary authentication method. With AD FS 2016, it required additional components and additional configurations in order to do so.
- **Risk Assessment Model**: Engineers can now build their own plugins to block or assign risk scores to authentication requests during the pre-authentication stage by using the AD FS 2022 Risk Assessment Model. More information about this module is available at `https://bit.ly/3l2Qe0m`.

- **Extranet Smart Lockout (ESL) is built into AD FS 2022**: ESL protects users' accounts from extranet account lockout when they log in from familiar locations. From a familiar location, if we detect multiple login failures for a particular user account, that means it could be an error rather than malicious activity. By using ESL, we can define different lockout thresholds for familiar and unfamiliar locations. AD FS 2022 also allows us to use audit mode to learn about familiar locations while the environment is still protected by classic extranet lockout functionality. With AD FS 2016, you do not have protection if you are using audit mode.

More information about other features and bug fixes is available at `https://bit.ly/3FLLy6T`.

The Web Application Proxy

We use proxy servers to access the internet because they perform the required communication with the internet on behalf of the internal users and protect them from external threats. The Web Application Proxy allows us to publish web applications (including AD FS) to the public without exposing the backend of them. This role is no longer a part of AD FS, and it comes as a part of the remote access role. AD FS does not require the Web Application Proxy to work, but it is recommended to use it if users log in from external networks. It also provides basic **Denial-of-Service** (**DoS**) protection by throttling and quieting connections.

The communications between proxy servers and web clients are encrypted (based on SSL). The Web Application Proxy is not supported for installing on the same AD FS server. It doesn't have built-in load balancing capabilities. If load balancing is required, it can be done using any supported software-/hardware-based load balancer.

AD FS configuration database

AD FS configuration settings need to be saved in a database. AD FS supports two types of databases. The simplest method is to use the **Windows Internal Database** (**WID**), which comes with the AD FS installation. This is not a standalone database installation, and it is capable of providing high availability by copying databases to other servers in the AD FS farm. When we go for the AD FS configuration, it gives two deployment options:

- Create the first federation server in a federation server farm
- Add the federation server to a federation server farm

If WID is used with the first option, then WID will be deployed with scalability, which allows servers to be added to the farm later and replicate WID. The first server in the farm will be the primary server and it will host the read/write copy of the database.

When we use the second option, the newly added server will replicate the copy of WID from the primary server, and it will be maintained as a read-only copy. Any configuration change should be replicated from the primary server. In the event of a primary server failure, other servers in the farm continue to process requests as normal, but no configuration changes will be possible until the primary server is brought online. If it's not possible to bring it online, then a secondary server can be forcefully nominated as a primary server.

AD FS can store configuration data in Microsoft SQL. This enhances the performance of the AD FS farm, especially if it deals with larger processing as it can read and write data faster. Unlike WID, we can add high availability to the SQL instance by using the SQL cluster method or the Always On method. If AD FS uses MS SQL, then every server in the farm has read/write access to the database. It also enables support for features such as SAML artifact resolution and SAML/WS-Federation token replay detection. Both features require a configuration that is stored in the shared SQL database instead of WID.

AD FS deployment topologies

There are a few different deployment models we can use for AD FS deployment:

1. A single federation server
2. A single federation server and single Web Application Proxy server
3. Multiple federation servers and multiple Web Application Proxy servers with SQL Server

In this section, we are going to look into these different topologies and their characteristics.

A single federation server

This is the simplest AD FS deployment model available. It contains a single AD FS server. It doesn't have high availability (unless at the host level).

This is ideal for a lab environment or staging environment:

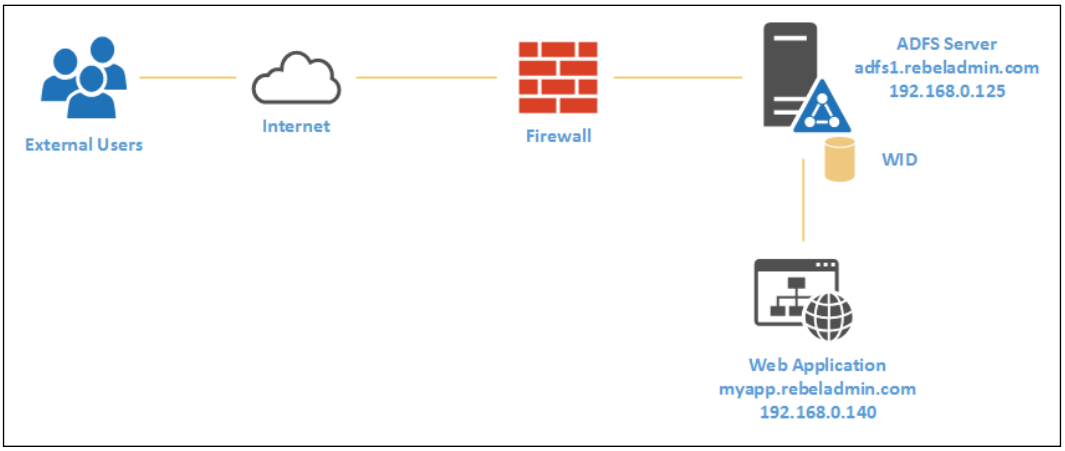


Figure 14.3: Single federation server deployment

In the preceding example, we have a web application, `myapp.rebeladmin.com`, that needs to allow access via AD FS. We have one AD FS server in the setup with WID. It is behind the corporate firewall and there are **Network Address Translation** (**NAT**) and access rules in place to do the following:

- Map a public IP address to `myapp.rebeladmin.com` so users can make initial requests from external networks. It is recommended to use TCP `443`.
- Map a public IP address to `secure.rebeladmin.com`, map it to the IP address of `adfs1.rebeladmin.com`, and open TCP `443` from the external networks to allow access.

The application should also have the relevant external and internal DNS records set up. If the request is coming from an external network, it should resolve to a public IP address and if the request is coming from an internal network, DNS should resolve to the internal IP addresses. This is also called a split-brain DNS setup.

My recommendation for this setup is to configure the AD FS server with a Windows **Network Load Balancer** (**NLB**). It is only going to cost you one additional IP address for the NLB cluster IP, but when we need to expand the AD FS farm, all we need to do is configure another server and add it to the NLB. The cluster IP will map to the public IP, and it will be used with an external DNS entry for AD FS.

In this setup, the Web Application Proxy hasn't been used:

Advantages	Disadvantages
Low cost to implement. Only one server required for AD FS and no SQL licenses have been used as it uses WID.	No redundancy. Single point of failure.
Easy to manage.	Poor performance as there is no way to share workloads.
No additional service role integration and therefore fewer dependencies.	Less secure as it is not possible to relay the requests to the AD FS server, and this method uses a direct connection point to process the requests (no Web Application Proxy).
Still can configure to support future expansions and can add servers to the AD FS farm whenever required.	N/A.

A single federation server and single Web Application Proxy server

This is an ideal setup to start with. This removes the security concerns we had with the single federation server. The Web Application Proxy server will be the initial connection point from the external network and it will relay requests into and out of the internal AD FS server.

This is still not going to provide high availability as each role holder only has one instance:

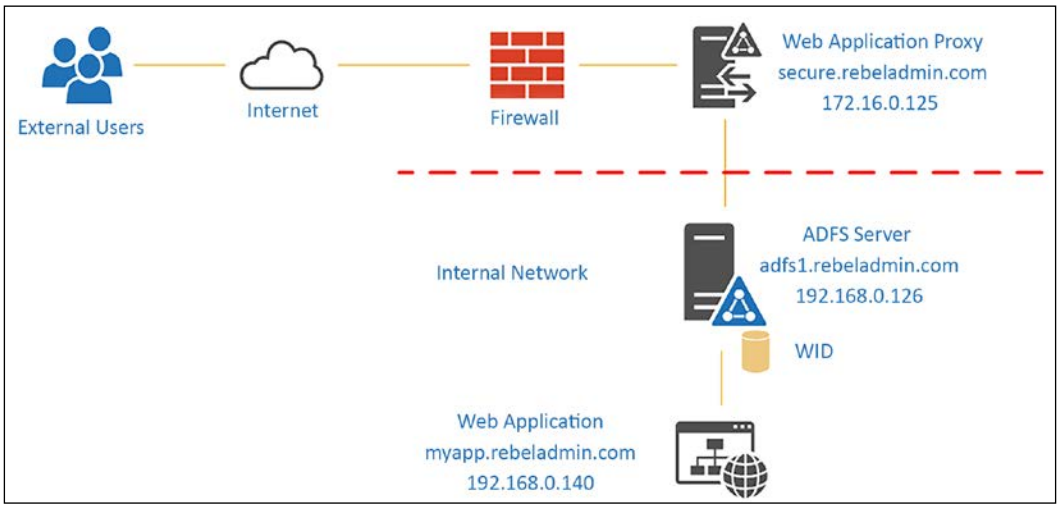


Figure 14.4: Single federation server and WAP deployment

In this setup, we can separate the network functionality between perimeter and corporate networks. The preceding setup needs NAT and access rules to accomplish the following communication requirements:

- Map a public IP address to `myapp.rebeladmin.com` so users can make initial requests from external networks. It is recommended to use TCP port `443`.
- Map a public IP address to `secure.rebeladmin.com`, map it to the IP address of the Web Application Proxy server, and open TCP port `443` to allow access.
- Allow access from the Web Application Proxy server to the AD FS server on TCP port `443`.

In the preceding example, once the external user accesses the application's URL, it will redirect to the Web Application Proxy server. This doesn't need to be domain joined as it operates from the perimeter network. Proxy servers should be able to resolve the DNS name for the AD FS servers from the perimeter network. This can be done by using a DNS server or a host file.

Similar to the previous model, this can be implemented with NLB to allow future expansions with minimum impact. We need two NLB clusters for that. The first NLB cluster is for the Web Application Proxy and the second NLB cluster is for AD FS servers. The only changes are in the DNS records and firewall rules. Instead of pointing DNS and firewall rules to the server IP addresses, we need to use NLB cluster IP addresses:

Advantages	**Disadvantages**
Improved security as the Web Application Proxy acts as an intermediate layer between external users and corporate networks.	No redundancy and a single point of failure.
Basic DoS protection by throttling and queuing connections.	The implementation cost is high compared to the single server model as additional servers need to be added.
This setup supports future expansions. It can easily add servers to the AD FS farm and the Web Application Proxy group when required.	Adding more roles also means more dependencies. Both roles need to function correctly to complete the process.

Both of the above models use the WID method for the AD FS database. But when it comes to high availability, we need to consider using the MS SQL method as multiple servers can write to the configuration database.

Multiple federation servers and multiple Web Application Proxy servers with SQL Server

So far, we have looked into models that benefit from easy implementation and improved security. But this model is focused on high availability. Each role will be configured with NLB clusters. The AD FS database will be hosted in the SQL Always On cluster environment for high availability. This model is ideal for SPs and other businesses that work with high volumes of AD FS requests.

The NLB cluster is a software-based load balancing solution that comes with the Microsoft Windows server OS. It is easy to implement with no additional licenses. However, hardware load balancers provide higher performance and fewer dependencies.

The following diagram provides a sample design for the given topology:

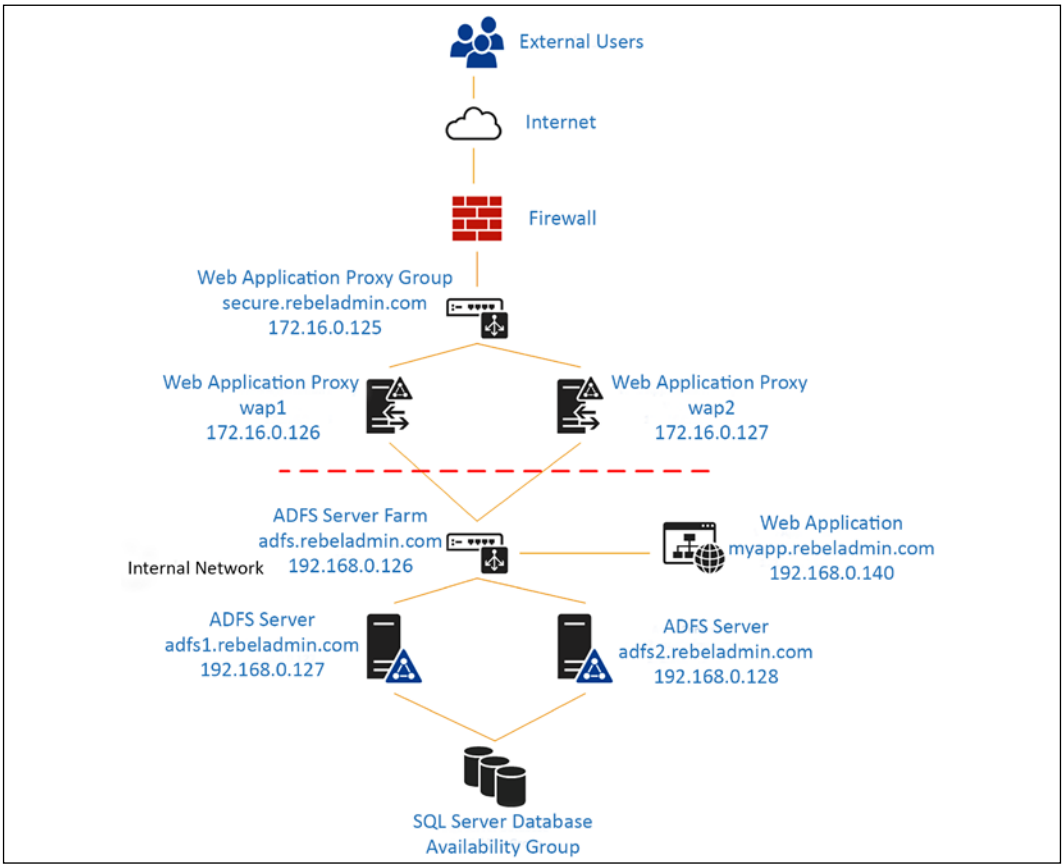


Figure 14.5: A multiple federation server and multiple WAP server deployment

Similar to the previous model, this model's operations are clearly divided into two network segments: perimeter and corporate. The network firewall should have NAT and access rules to support the following requirements:

- Map a public IP address to `myapp.rebeladmin.com` so that users can make initial requests from external networks. It is recommended to use TCP `443`.
- Map a public IP address to `secure.rebeladmin.com`, map it to the IP address of the Web Application Proxy server group's NLB cluster IP, and open TCP port `443` from the external networks to allow access.
- Allow access from Web Application Proxy servers to the AD FS farm NLB cluster IP on TCP port `443`.

For both NLB clusters, the initial connection point will be the NLB cluster IP. Apart from the application's external URL, the only external published URL will be the Web Application Proxy's URL. In the preceding example, `secure.rebeladmin.com` is mapped to the Web Application Proxy's NLB cluster IP.

AD FS servers use the Microsoft SQL Always On availability group to host the AD FS database. This is a read/write database for both hosts.

SQL Always On is a high-availability solution that runs on top of the Windows cluster. Windows Server 2022 supports a two-node cluster with Azure Cloud Witness. It reduces the number of servers that need to be used in a SQL Always On setup.

Advantages	**Disadvantages**
High availability: Each component hosts multiple servers with load balancers. AD FS databases also use a SQL high-availability environment.	High cost: It needs multiple servers and licenses (OS, SQL Server). This also increases the management cost.
High performance: Workloads are distributed between multiple hosts using load balancers.	Complex setup: The implementation is time-consuming and requires advanced skills for planning and configuration.
Support for features such as SAML artifact resolution and SAML/WS-Federation token replay detection.	Troubleshooting an issue is time-consuming and complex as there are many systems and application dependencies.

We just went through different deployment models and their pros and cons. The next step is to look into the AD FS configuration process.

AD FS deployment

In this section, we are going to look into AD FS deployment using a single federation server and a single Web Application Proxy server model. Before we move on to configuration, we need to sort out the following prerequisites:

- DNS records
- SSL certificates

Apart from that, we also need certain NAT and access rules in the firewall. But here, I am not going to talk about those in detail as I covered those when I explained the topologies in the previous section.

DNS records

We need to have a few DNS records (internal and external) set up prior to starting the deployment:

DNS Record	External	Internal
Application URL	Yes	Yes
WAP URL	Yes	N/A
AD FS URL	N/A	Yes

In the demo environment, the following URLs will be used:

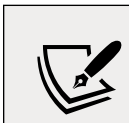
- `myapp.rebeladmin.com` will be the application, and it will have the external DNS record created and mapped to the public IP address. It will NAT to the application server IP address using a firewall. It will also have the internal DNS entry and point to the internal IP address of the application server.
- `secure.rebeladmin.com` will be the external WAP URL. WAP servers are in the perimeter network. It is not necessary to have the internal DNS record unless there are multiple WAP servers.
- `adfs.rebeladmin.com` will be the AD FS server DNS entry, and it does not need to have an external DNS entry. However, WAP servers need to communicate with AD FS servers via the TCP port. Since it's one server, there is no point deploying a DNS server in the perimeter network, and it can be done using the `hosts` file entry.

SSL certificates

AD FS deployment requires a few SSL certificates.

In this demonstration, we will use the following:

- `*.rebeladmin.com`: This is a wildcard SSL certificate for external URLs. This is used for the application and WAP.
- `rebeladmin.com`: This SSL is for AD FS service communication.



In the lab environment, we can create these certificates using an internal **Certification Authority** (**CA**). If the domain name is the same, then wildcard certificates are used internally and externally as well. Wildcard certificates simplify certificate management.

Installing the AD FS role

Before installation, the SSL certificate for `adfs.rebeladmin.com` needs to be installed in the computer account as it is required during the AD FS installation. This can be checked using the following command:

```
dir Cert:\LocalMachine\My
```

The AD FS server should be a member server of the domain and should log in as the domain administrator or the Enterprise Admin to do the installation.

The next step is to install the AD FS role service, which can be done by using the following PowerShell command:

```
Install-WindowsFeature ADFS-Federation -IncludeManagementTools
```

The following screenshot displays the output of the preceding command:

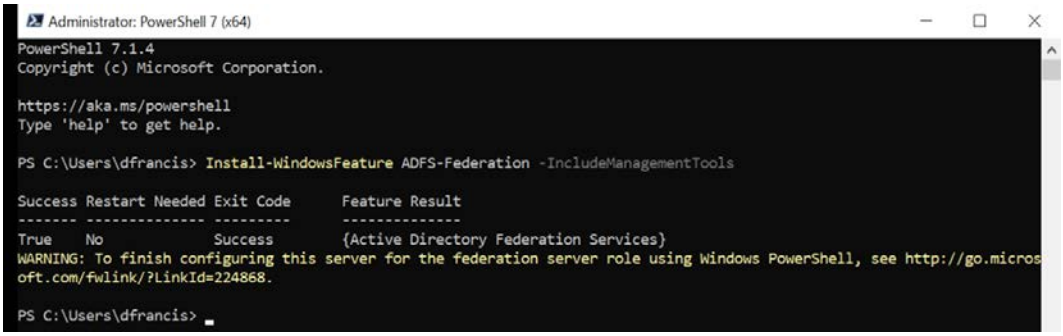


Figure 14.6: Install the AD FS role

Once this is completed, we need to configure the AD FS server. Let's use the following configuration for the demo setup:

```
Import-Module ADFS
$credentials = Get-Credential
Install-AdfsFarm `
-CertificateThumbprint:"938E369FA88B2F884A5BBC495F2338BE9FA0E0BB" `
-FederationServiceDisplayName:"REBELADMIN INC" `
-FederationServiceName:"adfs.rebeladmin.com" `
-ServiceAccountCredential $credentials
```

In this setup, we are using WID for AD FS, so there is no need for SQL configuration. In the preceding command, `CertificateThumbprint` specifies the SSL certificate (`adfs.rebeladmin.com`), and `FederationServiceDisplayName` specifies the display name of the federation service. `FederationServiceName` is the service name, and it should match the SSL we used. `ServiceAccountCredential` is used to define the service account details for the AD FS setup. In the end, the system needs to be restarted to apply the configuration:

```
PS C:\Users\dfrancis.REBELADMIN> Import-Module ADFS
WARNING: Module ADFS is loaded in Windows PowerShell using WinPSCompatSession remoting session; please note that all inp
ut and output of commands from this module will be deserialized objects. If you want to load this module into PowerShell
 please use 'Import-Module -SkipEditionCheck' syntax.
PS C:\Users\dfrancis.REBELADMIN> $credentials = Get-Credential

PowerShell credential request
Enter your credentials.
User: rebeladmin.com\dfrancis
Password for user rebeladmin.com\dfrancis: **************

PS C:\Users\dfrancis.REBELADMIN> Install-AdfsFarm `
>> -CertificateThumbprint:"69F65A89906485DFABF9DF8CC239508BF3256673" `
>> -FederationServiceDisplayName:"REBELADMIN INC" `
>> -FederationServiceName:"adfs.rebeladmin.com" `
>> -ServiceAccountCredential $credentials
WARNING: A machine restart is required to complete ADFS service configuration. For more information, see: https://go.mic
rosoft.com/fwlink/?LinkId=798725
WARNING: The SSL certificate subject alternative names do not support host name 'certauth.adfs.rebeladmin.com'. Configur
ing certificate authentication binding on port '49443' and hostname 'adfs.rebeladmin.com'.

RunspaceId : 1dbbcd54-0bc4-4994-ac17-602eaa129479
Message    : The configuration completed successfully.
Context    : DeploymentSucceeded
Status     : Success
```

Figure 14.7: Configure the AD FS role

The error about the alternative SSL name, `certauth.adfs.rebeladmin.com`, regards the certificate authentication. Before Windows Server 2016, this was an issue as the system didn't support different bindings for certificate authentication and device authentication on the same host. The default port `443` was used by the device authentication and couldn't have multiple bindings on the same channel.

In Windows Server 2016/2016/2019/2022, this is possible and now, it supports two methods. The first option is to use the same host (`adfs.rebeladmin.com`) with different ports (`443` and `49443`). The second option is to use different hosts (`adfs.rebeladmin.com` and `certauth.adfs.rebeladmin.com`) with the same port (`443`). This requires an SSL certificate to support `certauth.adfs.rebeladmin.com` as an alternate subject name.

Once the reboot completes, we can check whether the installation was successful by using the following command:

```
Get-WinEvent "AD FS/Admin" | Where-Object {$_.ID -eq "100"} | fl
```

This will print the content of event `100`, which confirms the successful AD FS installation.

Installing WAP

The next step of the configuration is to install WAP. This doesn't need to be a domain-joined server and should be placed on the perimeter network. Before the installation process, install the required SSL certificates. In my demo, it is for `*.rebeladmin.com`. We can verify this by using this:

```
dir Cert:\LocalMachine\My
```

Before proceeding, we also need to check whether a server can resolve to `adfs.rebeladmin.com` as WAP needs to connect to AD FS.

Once everything is confirmed, we can install the WAP role:

```
Install-WindowsFeature Web-Application-Proxy -IncludeManagementTools
```

Once it's completed, we can proceed with the configuration by using the following:

```
$credentials = Get-Credential
Install-WebApplicationProxy
-FederationServiceName "adfs.rebeladmin.com"
-FederationServiceTrustCredential $credentials
-CertificateThumbprint "3E0ED21E43BEB1E44AD9C252A92AD5AFB8E5722E"
```

In the preceding commands, `FederationServiceName` is used to define the AD FS service name, and it needs to match the name provided on the AD FS setup. `FederationServiceTrustCredential` is used to provide an account, which is authorized to register a new proxy server with AD FS. The account that is used here should have permissions to manage AD FS.

The `CertificateThumbprint` parameter is used to define the certificate for WAP. In our demo, it's the `*.rebeladmin.com` certificate. At the end of the configuration, we need to restart the system to apply the changes.

Once the reboot is completed, we can confirm the health of the configuration using the following event log in the AD FS server:

```
Get-WinEvent "AD FS/Admin" | Where-Object {$_.ID -eq "396"} | fl
```

Configuring the claims-aware application with new federation servers

At the start of this chapter, I explained that not every application can use AD FS for authorization. It should be a claims-aware application. I have an application called `myapp.rebeladmin.com` that is already set up. In the configuration, I set it up to use the existing STS and added the new AD FS server's metadata URL, which is `https://adfs.rebeladmin.com/federationmetadata/2007-06/federationmetadata.xml`.

If the configuration is successful, AD FS installs the metadata XML, and you should be able to view this using the web browser. If it cannot load, then you need to check it before this step.

Once the application is configured, when I go to my application, which is `https://myapp.rebeladmin.com/myapp` (internally), I can see the following error. This was expected as the AD FS setup does not know about my application yet:

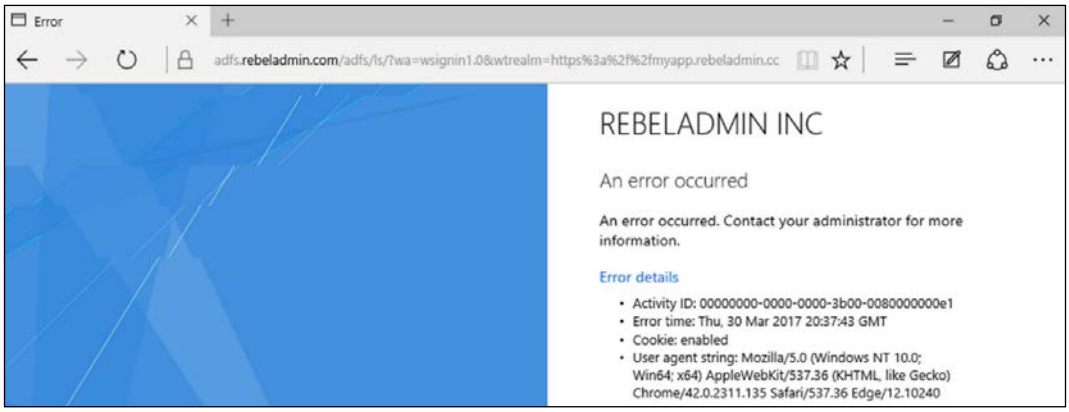


Figure 14.8: AD FS login page

Creating a relying party trust

To get our application working, we need to create a relying party trust between our application and the AD FS setup. Then, only the AD FS setup will know about the application.

In order to do that, perform the following steps:

1. Log in to the AD FS server as an administrator.
2. Go to **Server Manager | Tools | AD FS Management**.
3. Go to **Relying Party Trusts (1)**, and then click on **Add Relying Party Trust... (2)**:

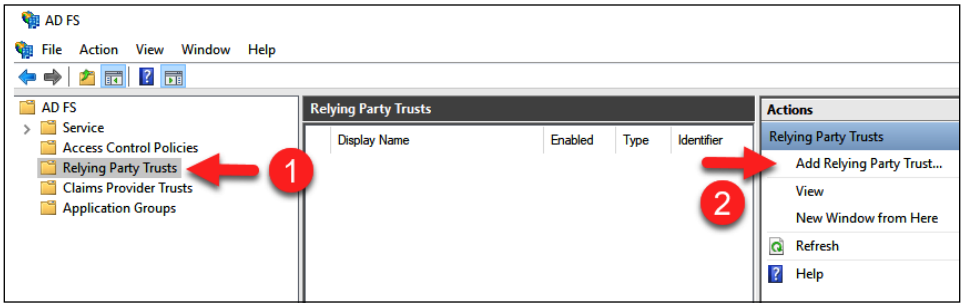


Figure 14.9: Add Relying Party Trust

4. The system will open the Add Relay Party Trust Wizard. Select **Claims Aware** and click **Start**.

5. On the **Select Data Source** page, select **Import data about the relying party published online or on a local network** and enter the metadata URL for the application. For my application, I have created the metadata file under `https://myapp.rebeladmin.com/myapp/federationmetadata/2007-06/federationmetadata.xml`:

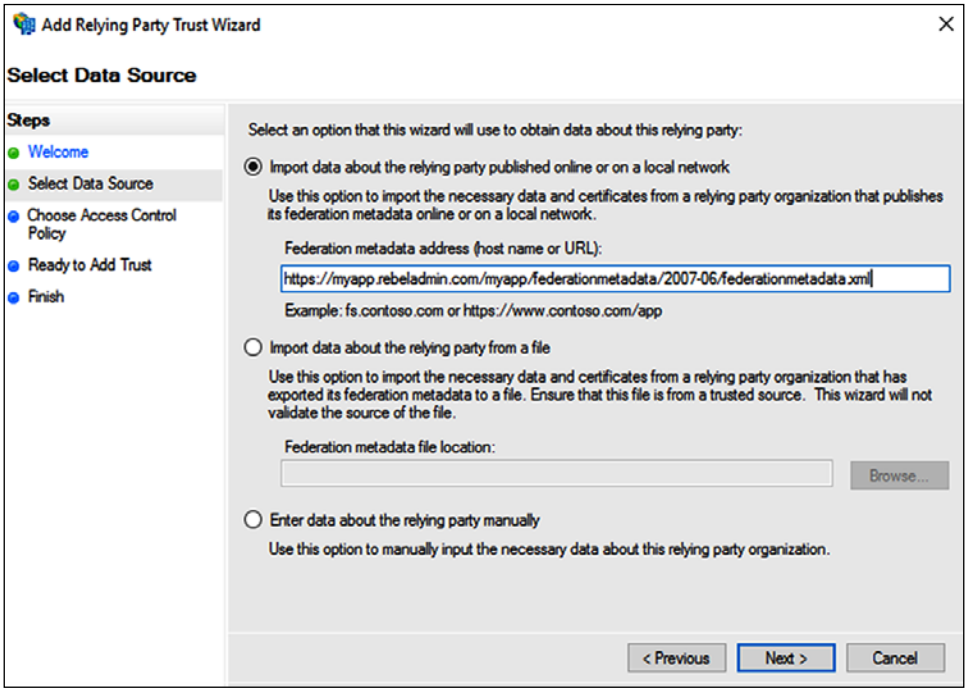


Figure 14.10: Configure the relying party trust

When we establish a trust, we need to provide certain information to match the SP's application configuration. This is painful as even a small mistake can cause issues. Also, if these settings are changed on the SP's side, we will not know until they inform us of what to modify on the AD FS side. Therefore, service providers use metadata XML files to publish these required settings. This simplifies the configuration. If there is no metadata file, we still can create the trust using the required custom settings.

6. On the next page, go to **Specify Display Name** for the claim and click on **Next**.

7. On the **Choose an access control policy** page, select **Permit Everyone**, and click on **Next**. This is a new feature of AD FS 2016/2016/2019/2022, and it allows us to create access policies easily. It is possible to add customer access policies as well. In this demo, I am not going to use any MFA as it's a test lab:

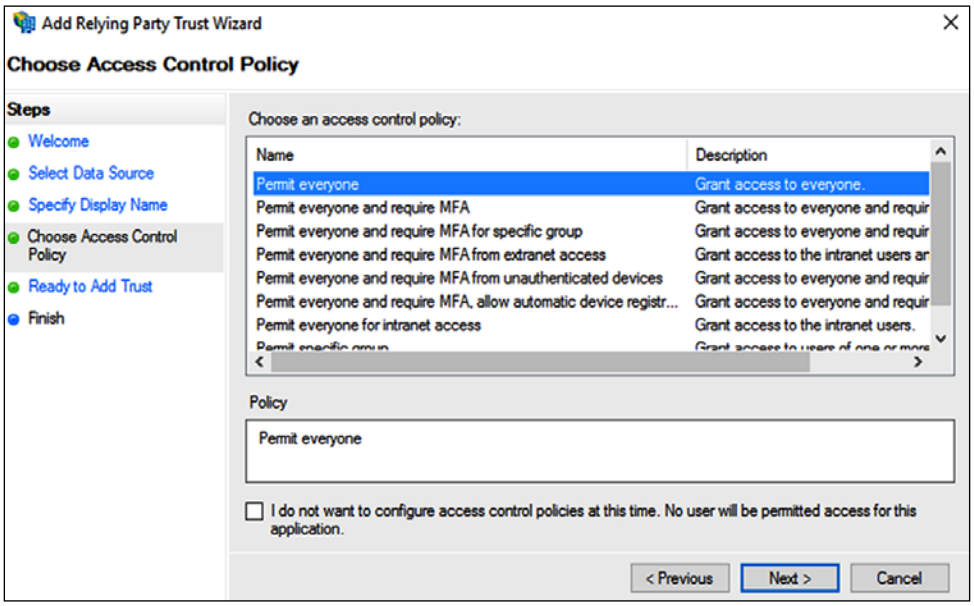


Figure 14.11: Relying party trust access control policies

8. In the next window, we can review the settings and click on **Next** to continue.
9. On the **Finish** page, keep the check for **Configure claims issuance policy for this application** and click on **Close**:

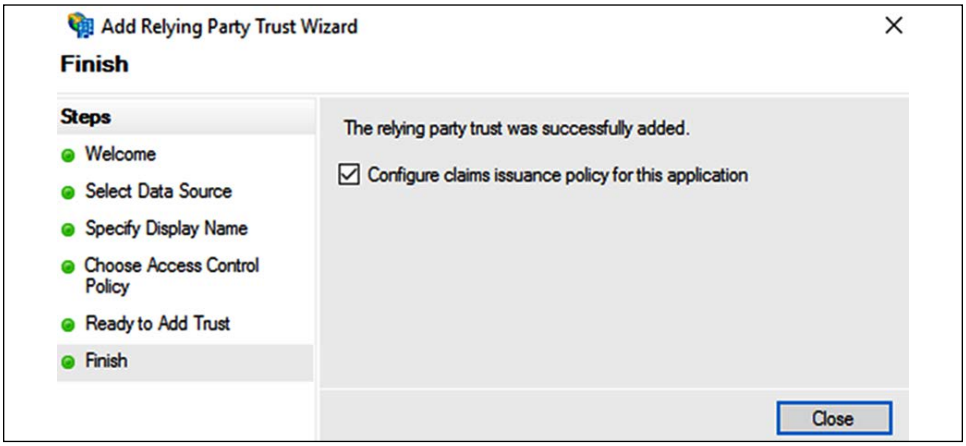


Figure 14.12: Completing the relaying party trust wizard

10. The **Edit Claim Issuance policy** window will open. If it does not, click on **Edit Claim Issuance policy** on the **action** pane. Once the window is open, click on the **Add Rule** button.
11. In **Add Transform Claim Rule Wizard**, select **Send Claims Using a Custom Rule** and click on **Next**.
12. On the next page, type the name for the custom rule and input the claim rule. The claim rule depends on the application requirements. Most application vendors specify what kind of claim rule you need to have. Once complete, click on **Finish**.
13. Once finished, click **OK** to exit from the wizard.

Configuring the Web Application Proxy

Now, we have the application configured with AD FS. But our requirement is to use the Web Application Proxy to publish the application to the public.

In order to do that, log in to the Web Application Proxy server as an administrator and execute the following command:

```
Add-WebApplicationProxyApplication
-BackendServerUrl 'https://myapp.rebeladmin.com/myapp/'
-ExternalCertificateThumbprint
'3E0ED21E43BEB1E44AD9C252A92AD5AFB8E5722E'
-ExternalUrl 'https://myapp.rebeladmin.com/myapp/'
-Name 'MyApp'
-ExternalPreAuthentication AD FS
-ADFSRelyingPartyName 'myapp.rebeladmin.com'
```

In the preceding command, `ExternalUrl` specifies the external URL for the application. `BackendServerUrl` specifies the internal URL for the application. `ExternalCertificateThumbprint` is the certificate to use from external networks. The `Name` parameter specifies the custom name for the app, which will be displayed on the proxy page. `ExternalPreAuthentication` defines the authentication mode. On our setup, we're using AD FS mode. It also supports pass-through mode. `ADFSRelyingPartyName` specifies the AD FS relying party name that will be used for this application.

The Web Application Proxy can translate hostnames used in the external URL and the backend URL. However, it cannot translate paths.

Once all is done, when we access the application from the external URL `https://myapp.rebeladmin.com/myapp/`, it successfully proxies to AD FS, and after successful authentication, the application page is displayed:

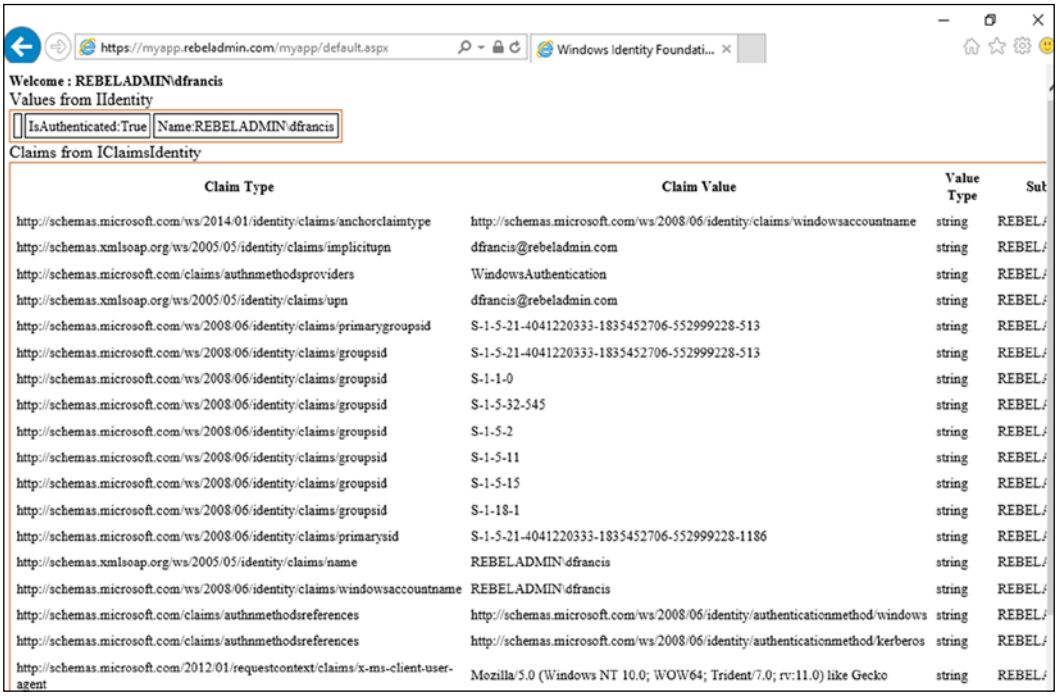


Figure 14.13: Application access

Integrating with Azure MFA

MFA is a common requirement today for online services, which could be for any hosted solution in organizations such as Citrix, RDS, or any other web application. MFA is also used in hybrid cloud environments to provide the same level of protection on-prem and in the cloud.

There are many MFA service providers in the market. Some of those are on-prem solutions, where we can install an appliance and use MFA services. Others are cloud-based solution providers and sell MFA services as subscriptions. Customers can simply install an agent on-prem and connect it to these cloud-based solutions. Azure MFA was first introduced to be used with Azure services and later improved to support on-prem workload protections too. Users can use SMS, calls, or a PIN on the Microsoft Authenticator application to authenticate. Most of the MFA SPs have a separate agent, which needs to be installed in the AD FS servers in order to connect it with the MFA services.

Azure MFA integration was complicated in Windows Server 2012 R2 environments. It needed an agent as well as an on-prem Azure MFA server. A big change with AD FS 2016/2019/2022 was the Azure MFA integration enhancement. With AD FS 2016/2019/2022, we no longer need to install these components. The AD FS Azure adapter allows integration with Azure AD in order to pull the configuration. In this section, we are going to look at how we can integrate the AD FS setup with Azure MFA.

Prerequisites

To configure Azure MFA, we need a few things:

- A valid Azure subscription.
- Azure Global Administrator privileges.
- The Azure AD federated setup. Azure AD needs to integrate with AD FS on-prem and synchronize identities to Azure. This will be covered later in this chapter.
- Windows Server 2022 AD FS in the local infrastructure.
- Enterprise Admin privileges for AD FS servers to configure MFA.
- Azure MFA needs to be enabled. The users that sync from on-prem AD need to have MFA enabled. I wrote an article about this before, and you can access it via `https://bit.ly/3kZSTIc`.
- The Windows Azure AD module for Windows PowerShell in AD FS servers. This can be downloaded from `https://bit.ly/3l43zFG`.

Creating a certificate in an AD FS farm to connect to Azure MFA

First, we need to create a certificate, which will be used by the AD FS farm. This needs to run from the AD FS server:

```
$certbase64 = New-AdfsAzureMfaTenantCertificate -TenantID 05c6f80c-
61d9-44df-bd2d-4414a983c1d4
```

The preceding command generates the new certificate. `TenantID` is the subscription ID you have from Azure. This can be found by running this:

```
Login-AzureRmAccount
```

The preceding command will ask for the credentials for Azure and once we provide them, it will list `TenantId`:

```
Environment           : AzureCloud
Account               : dcadmin@REBELADMIN.onmicrosoft.com
TenantId              : 05c6f80c-61d9-44df-bd2d-4414a983c1d4
SubscriptionId        :
SubscriptionName      :
CurrentStorageAccount :
```

Figure 14.14: Azure TenantId

This will create a certificate under **Certificates (Local Computer)**:

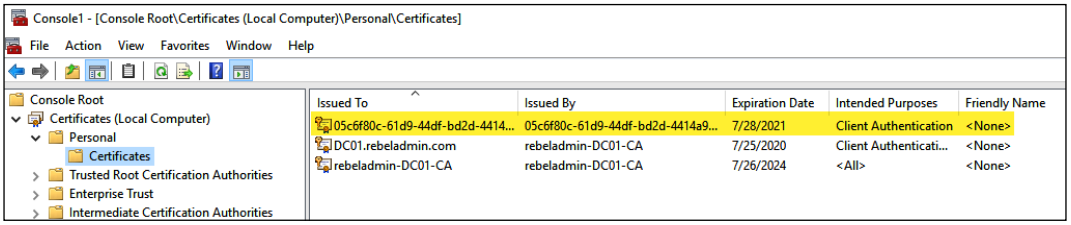


Figure 14.15: New certificate

Enabling AD FS servers to connect with the Azure MFA client

Now, we have the certificate, but we need to tell the Azure MFA client to use it as a credential to connect with AD FS.

Before that, we need to connect to Azure AD by using Azure PowerShell. We can do that by using the following command:

```
Connect-MsolService
```

Then, it will prompt you for your login and use your Azure Global Administrator account to connect.

After that, we can pass the credentials by using the following command:

```
New-MsolServicePrincipalCredential -AppPrincipalId 981f26a1-7f43-403b-
a875-f8b09b8cd720 -Type asymmetric -Usage verify -Value $certbase64
```

In the preceding command, `AppPrincipalId` defines the **Globally Unique Identifier** (**GUID**) for the Azure MFA client.

Enabling the AD FS farm to use Azure MFA

The next step of the configuration is to enable the AD FS farm to use Azure MFA. This can be done by using the following command:

```
Set-AdfsAzureMfaTenant -TenantId 05c6f80c-61d9-44df-bd2d-4414a983c1d4
-ClientId 981f26a1-7f43-403b-a875-f8b09b8cd720
```

In the preceding command, `TenantId` refers to the Azure tenant ID and `ClientId` represents the Azure MFA client's GUID.

Once the command successfully runs, we need to restart the AD FS service on each server in the farm:

```
PS C:\Users\administrator.REBELADMIN\Desktop> Set-AdfsAzureMfaTenant -TenantId 05c6f80c-61d9-44df-bd2d-4414a983c1d4
entId 981f26a1-7f43-403b-a875-f8b09b8cd720
WARNING: PS0177: The authentication provider configuration data was successfully updated.  Before your changes take
effect, you must restart the AD FS Windows Service on each server in the farm.
PS C:\Users\administrator.REBELADMIN\Desktop>
```

Figure 14.16: Enabling Azure MFA for AD FS

Enabling Azure MFA for authentication

The last step of the configuration is to enable Azure MFA globally for the AD FS server.

In order to do that, log in to the AD FS server as the Enterprise Admin. Then, go to **Server Manager | Tools | AD FS Management**.

Then, in the console, navigate to **Service | Authentication Methods**. Then, in the **Actions** panel, click on **Edit Primary Authentication Method**:

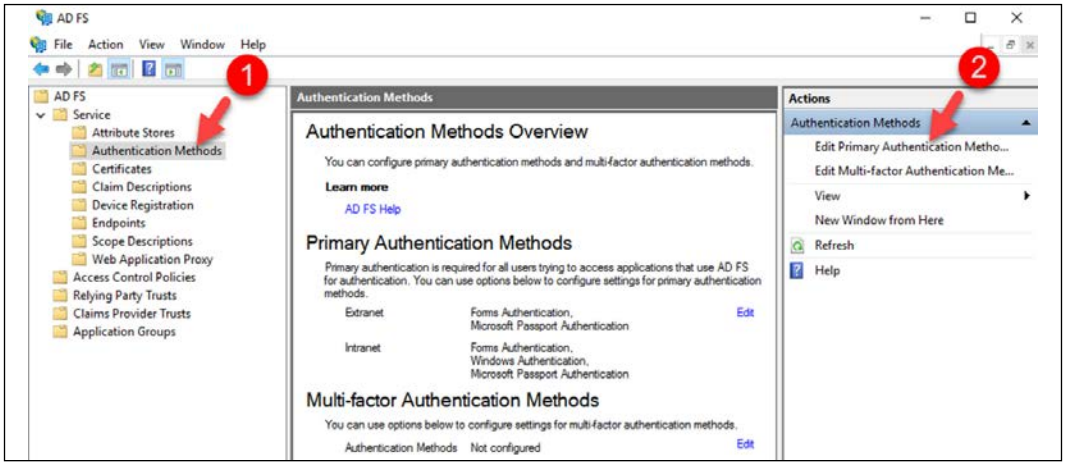


Figure 14.17: AD FS Authentication Methods

This opens up the window to configure global authentication methods. It has two tabs, and we can see **Azure MFA** on both. If Azure MFA is used as a primary method by removing other options, then AD FS will not ask for logins and will use MFA as the only authentication method.

Its operation boundaries can be set to intranet or extranet:

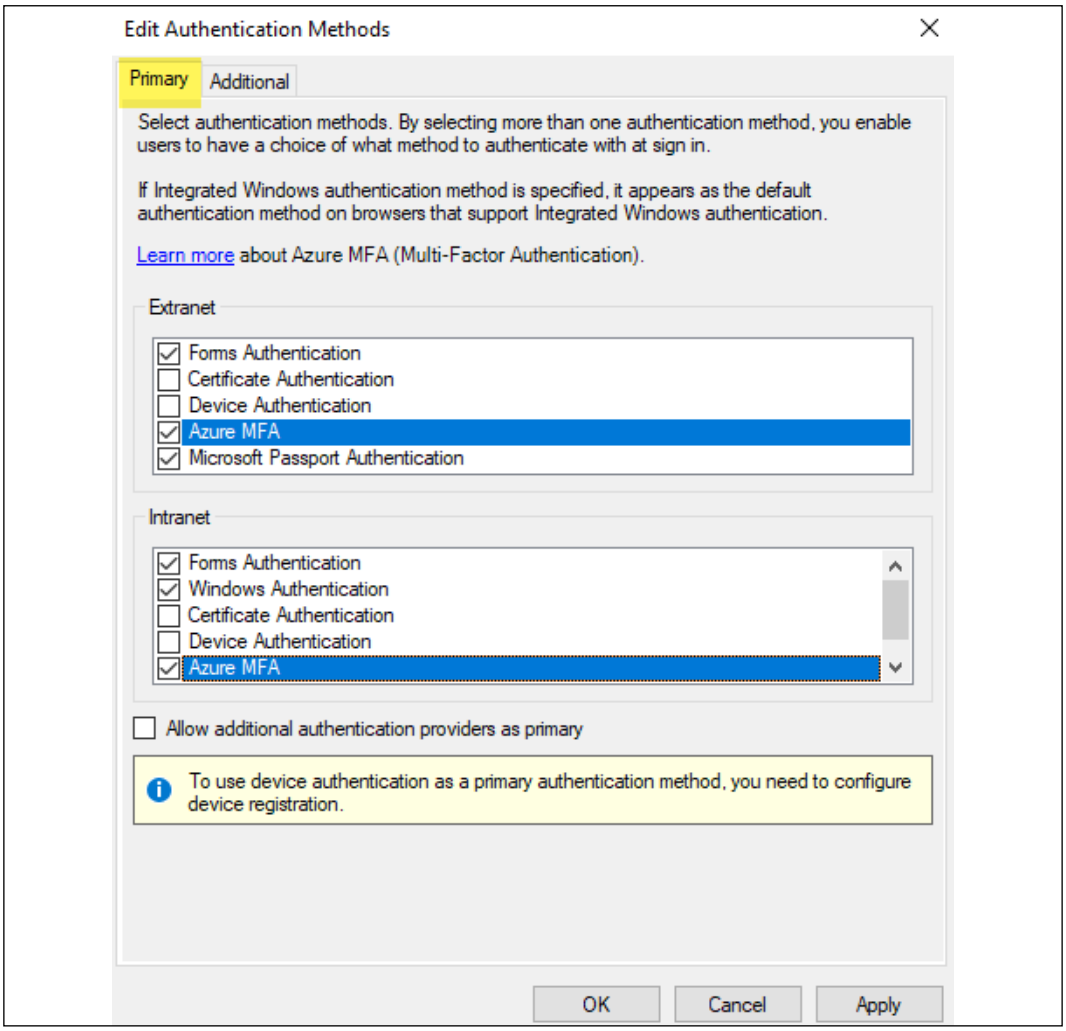


Figure 14.18: AD FS Primary authentication methods

Another option is to select MFA as the secondary authentication method:

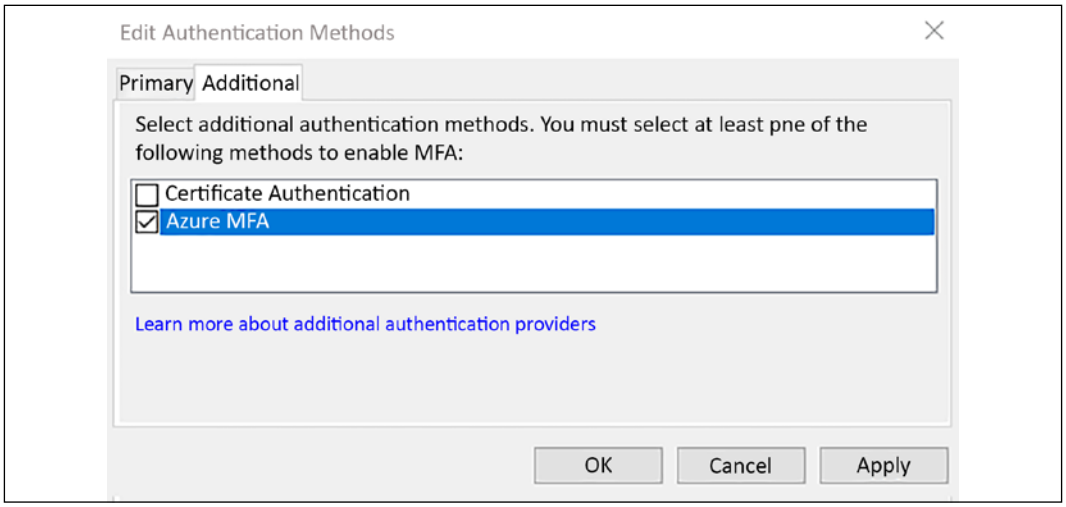


Figure 14.19: AD FS Additional authentication methods

This finishes the Azure MFA integration, and users can use MFA based on the options selected in the preceding wizards.

Azure AD federation with AD FS

Azure AD supports various integration methods with on-prem Active Directory. We can configure federation between on-prem AD FS and Azure AD to enable integration between two systems. When federation sign-in is in place, users can log in to Azure AD using the same on-prem Active Directory user name and password. This method ensures the user authentication occurs on-prem. We can use Azure AD Connect to configure the federation. During the configuration process, we can either deploy a new AD FS server/farm or configure existing AD FS servers. In this section, I am going to demonstrate how we can configure federation sign-in between AD FS and Azure AD. Before we go into that, it is important to understand how exactly the federation sign-in method works.

Federation sign-in with Azure AD

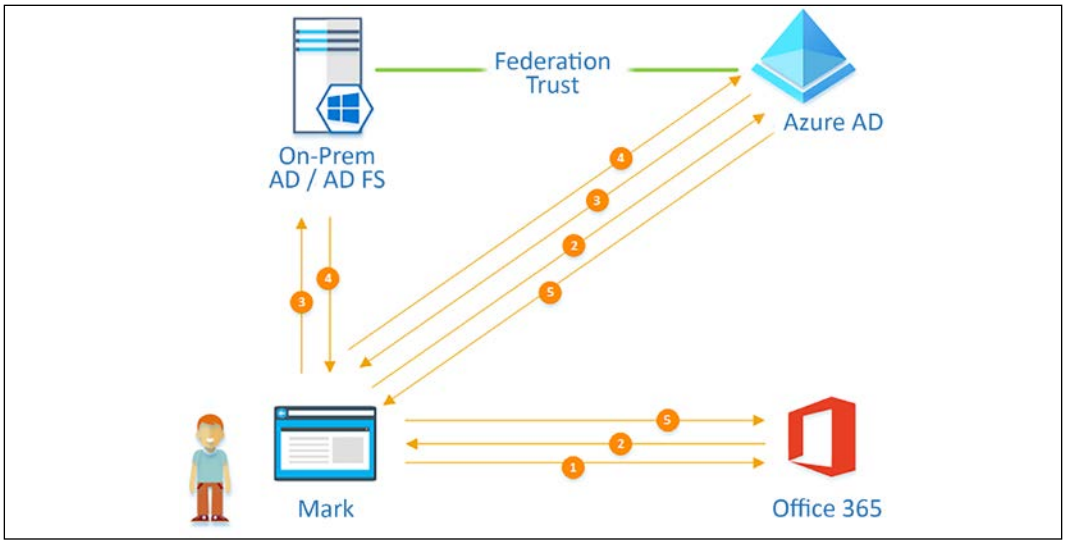


Figure 14.20: How AD FS federation works with Azure AD

Rebeladmin Inc. has federation between Azure AD and on-prem AD FS. The user Mark is trying to access the Office 365 portal using his browser. Let's see how the authentication and authorization work with federation sign-in:

1. Mark opens his browser and types `https://bit.ly/3oPeCUc` to access the Office 365 portal.
2. Office 365 has a trust relationship with Azure AD. Therefore Office 365 sends an **authentication request** to Azure AD via Mark's browser. Mark can see the Azure AD sign-in page in his browser. He goes ahead and enters his user name, `mark@rebeladmin.com`, and clicks on **Next**. Azure AD uses `@rebeladmin.com` as a hint and checks its configuration to see if `rebeladmin.com` is a federated domain. In this scenario, `rebeladmin.com` is a federated domain and Azure AD knows where to send the **sign-in request**. This process is also called **Home Realm Discovery**.

3. Azure AD sends a **sign-in** request to the on-prem AD FS server via Mark's browser.
4. Mark completes the authentication process with AD FS (Forms-based or Kerberos) by using his Active Directory user name and password. Upon successful authentication, AD FS responds to the original sign-in request by Azure AD with a token. Let's call it **token A**. This token represents the user. As `rebeladmin.com` is federated, Azure AD can verify the authenticity of the token by validating the signature.
5. Azure AD responds to the sign-in request it received from the Office 365 portal with a new token. Let's call it **token B**. Office 365 has trust with Azure AD so it can also validate the token signature. After the validation process, Mark will have access to the Office 365 portal. It is important to understand there is no relationship between token A and token B. Both tokens are used by different systems in different stages of the federation sign-in process.

In the next section, I am going to demonstrate how to create federation trust between AD FS and Azure AD.

Creating federation trust between Azure AD and AD FS

The following illustration explains the planned demo setup for this exercise.

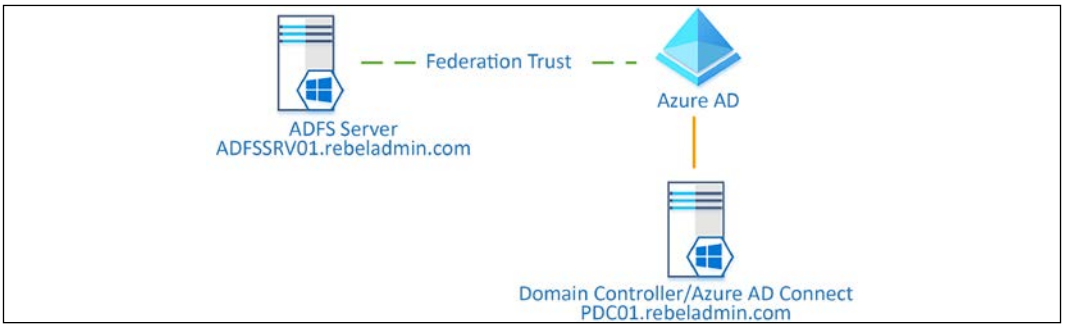


Figure 14.21: Planned demo setup

In the demo environment, I already have the Active Directory domain controller (`PDC01.rebeladmin.com`) up and running. It is using `rebeladmin.com` as the Active Directory domain name.

1. The domain controller will also be used for **Azure AD Connect**.
2. I have added the `rebeladmin.com` domain to the Azure AD custom domain list and completed the verification process already.

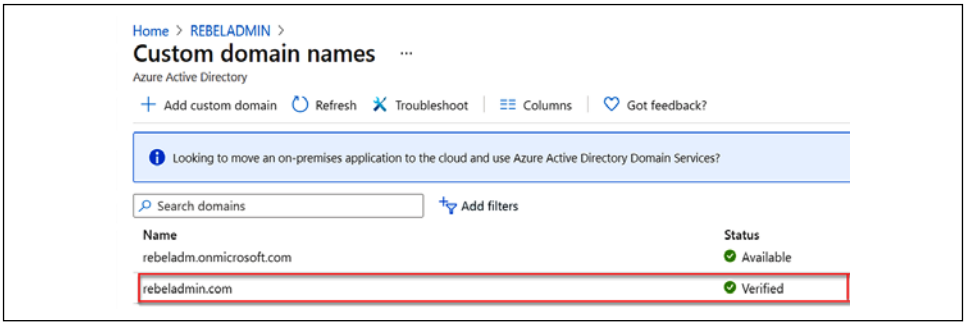


Figure 14.22: Azure AD custom domain status

   In this demo, we are going to use a single AD FS server (`ADFSSRV01.rebeladmin.com`). We will not use a web application proxy (WAP).
3. The federation service name is set to `adfs.rebeladmin.com` and relevant public DNS records and firewall rules are in place to allow `HTTPS` (443) access to the server from the internet.
4. I am using a valid wildcard SSL certificate, which is issued to `*.rebeladmin.com` for AD FS.

In the *AD FS Deployment* section of this chapter, we've already demonstrated how to set up the AD FS service, so we won't repeat that here. We already have AD FS configured on `ADFSSRV01.rebeladmin.com`.

We can view the current configuration of the AD FS server by using `Get-AdfsProperties`.

```
PS C:\Windows\System32> Get-AdfsProperties

RunspaceId                          : 38d17077-6170-4352-be9c-7087e2275dae
AcceptableIdentifiers               : {}
AddProxyAuthorizationRules          : exists([Type ==
                                      "http://schemas.microsoft.com/ws/2008/06/identity/claims/groupsid", Value
                                      == "S-1-5-32-544", Issuer =~ "^AD AUTHORITY$"]) => issue(Type =
                                      "http://schemas.microsoft.com/authorization/claims/permit", Value =
                                      "true");
                                                      c:[Type ==
                                      "http://schemas.microsoft.com/ws/2008/06/identity/claims/primarysid",
                                      Issuer =~ "^AD AUTHORITY$" ]
                                                                          => issue(store="_ProxyCredentialStore",types=("http://schemas.micr
                                      osoft.com/authorization/claims/permit"),query="isProxyTrustManagerSid({0})
                                      ", param=c.Value );
                                                      c:[Type ==
                                      "http://schemas.microsoft.com/ws/2008/06/identity/claims/proxytrustid",
                                      Issuer =~ "^SELF AUTHORITY$" ]
                                                                          => issue(store="_ProxyCredentialStore",types=("http://schemas.micr
                                      osoft.com/authorization/claims/permit"),query="isProxyTrustProvisioned({0}
                                      )", param=c.Value );
ArtifactDbConnection                : Data Source=np:\\.\pipe\microsoft##wid\tsql\query;Initial
                                      Catalog=AdfsArtifactStore;Integrated Security=True
AuthenticationContextOrder          : {urn:oasis:names:tc:SAML:2.0:ac:classes:Password,
                                      urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport,
                                      urn:oasis:names:tc:SAML:2.0:ac:classes:TLSClient,
                                      urn:oasis:names:tc:SAML:2.0:ac:classes:X509…}
AuditLevel                          : {Basic}
AutoCertificateRollover             : True
CertificateCriticalThreshold        : 2
CertificateDuration                 : 365
CertificateGenerationThreshold      : 20
CertificatePromotionThreshold       : 5
CertificateRolloverInterval         : 720
CertificateSharingContainer         : CN=30a49201-c09b-46c6-b238-8a1ca6f35b8a,CN=ADFS,CN=Microsoft,CN=Program
                                      Data,DC=rebeladmin,DC=com
CertificateThresholdMultiplier      : 1440
ClientCertRevocationCheck           : None
ContactPerson                       :
DisplayName                         : REBELADMIN INC
IntranetUseLocalClaimsProvider      : False
ExtendedProtectionTokenCheck        : Allow
FarmRoles                           : Microsoft.IdentityServer.PolicyModel.Configuration.FarmRolesConfiguration
FederationPassiveAddress            : /adfs/ls/
HostName                            : adfs.rebeladmin.com
HttpPort                            : 80
HttpsPort                           : 443
TlsClientPort                       : 49443
Identifier                          : http://adfs.rebeladmin.com/adfs/services/trust
IdTokenIssuer                       : https://adfs.rebeladmin.com/adfs
InstalledLanguage                   : en-US
LogLevel                            : {Errors, FailureAudits, Information, Verbose…}
MonitoringInterval                  : 1440
NetTcpPort                          : 1501
NtlmOnlySupportedClientAtProxy      : False
OrganizationInfo                    :
PreventTokenReplays                 : False
ProxyTrustTokenLifetime             : 21600
ReplayCacheExpirationInterval       : 60
SignedSamlRequestsRequired          : False
SamlMessageDeliveryWindow           : 5
SignSamlAuthnRequests               : False
SsoLifetime                         : 480
PersistentSsoLifetimeMins           : 129600
KmsiLifetimeMins                    : 1440
PersistentSsoEnabled                : True
```

Figure 14.23: Existing AD FS configuration

As per the above screenshot, we can see the server is configured with the `adfs.rebeladmin.com` service name. I also enabled the AD FS sign-in page for testing purposes by using `Set-AdfsProperties -EnableIdpInitiatedSignonPage $true`.

After that, I can access the `https://bit.ly/2ZgF40k` sign-in page over the internet and test logging in using the domain user account.

Figure 14.24: Initiated sign-in page

The next stage of the configuration is to configure Azure AD Connect for federation.

Configuring Azure AD Connect

In this demo setup, I am going to install Azure AD Connect on the `PDC01.rebeladmin.com` server. To do that:

1. Log in to the VM as **Domain Administrator**.
2. Download the latest Azure AD Connect file. Then double-click on `AzureADConnect.msi`.
3. This will open up a new wizard. On the **Express Settings** page, click on **Customize**.

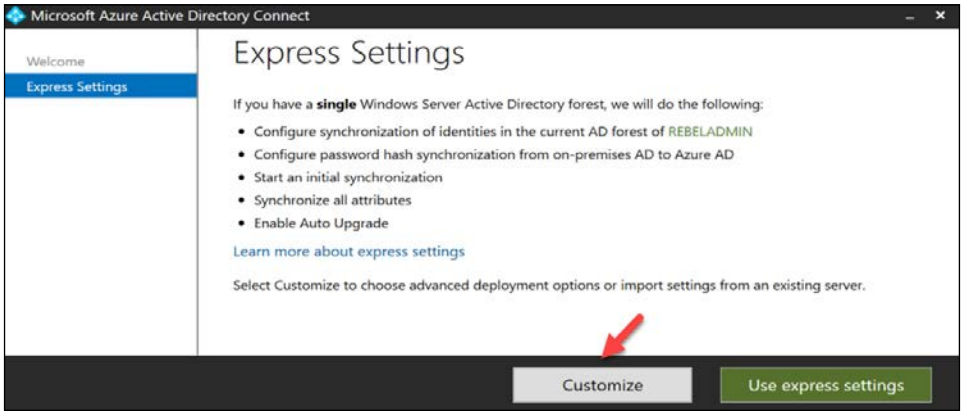


Figure 14.25: Azure AD Connect Express Settings page

4. On the **Install required components** screen, click the **Install** button.
5. This will install **Azure AD Connect**.
6. After the installation, on the user sign-in page select **Federation with AD FS** and click **Next**.

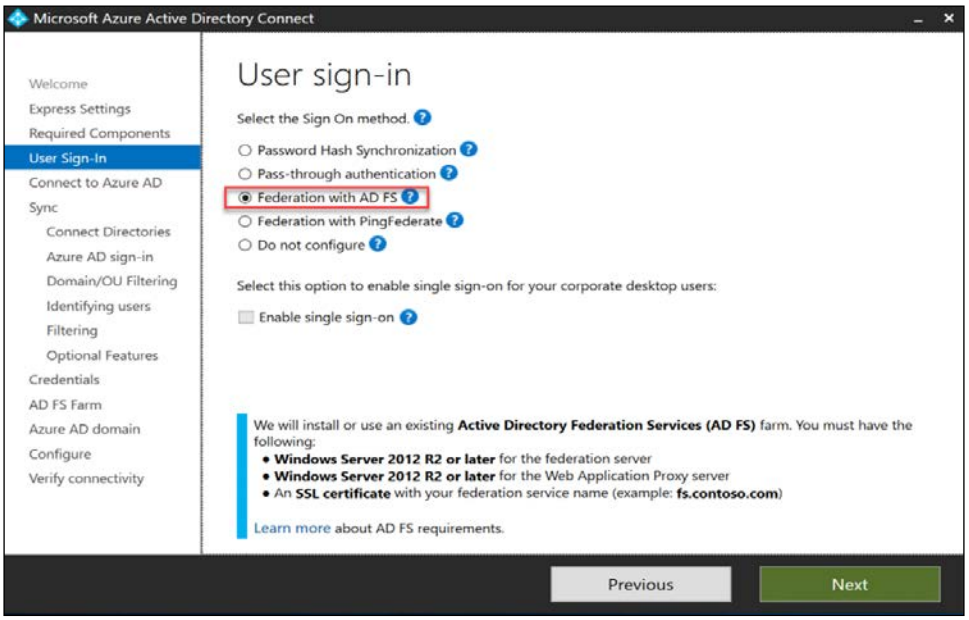


Figure 14.26: User sign-in method selection

7. On the next page, provide the Azure Global Administrator user name and password to connect to Azure AD.
8. On the **Connect your directories** page, click the **Add Directory** button to create a new service account and add the existing domain to the configuration. Once the settings are in place, click on **Next** to proceed.
9. On the Azure AD sign-in configuration screen, I can see `rebeladmin.com` is already listed as the verified domain. So, we do not need to make any changes. Click on **Next** to proceed.
10. On the **Domain and OU filtering**, **Identifying users**, **Filtering**, and **Optional Features** pages, I am using default values.
11. Then, on the **Credentials** page, we need to provide domain administrator account details and click **Next**.

12. On the **AD FS farm** page, use the **Use an existing AD FS farm** option and then select the current AD FS server. After the information is in place, click on **Next**.

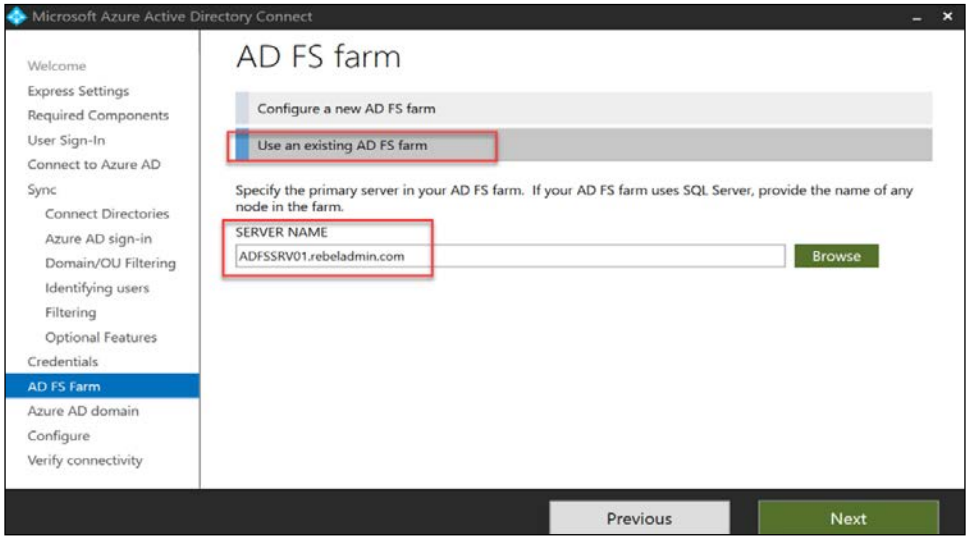


Figure 14.27: AD FS farm configuration

13. On the **Azure AD domain** page, I select `rebeladmin.com` as the domain to federate and click on **Next** to proceed.

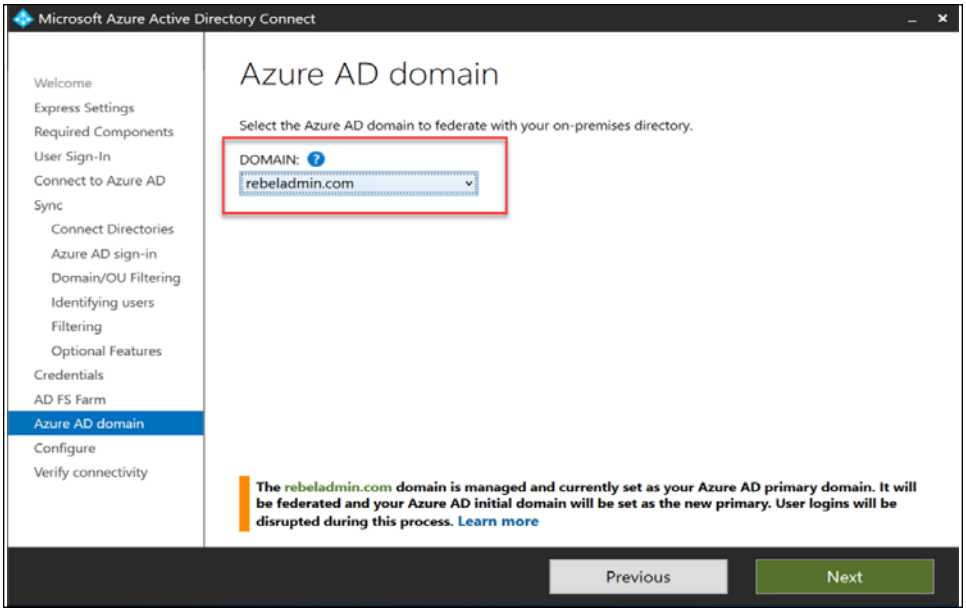


Figure 14.28: Select the domain to federate

14. Then, on the **Ready to configure** screen, click the **Install** button.
15. This will start the configuration process and can take a few minutes to complete.
16. Once the configuration is completed, exit the wizard and allow a few minutes to complete the initial synchronization process.
17. The next stage of the configuration is to do testing to verify the federation setup.

Testing

We can start the testing process by confirming the Azure AD Connect configuration status. To do that:

1. Log in to the Azure portal as **Global Administrator**.
2. Then go to **Azure Active Directory | Azure AD Connect**. Then click on **Federation** under the **user-sign in** section. In there, we can see `rebeladmin.com` is recognized as the federated domain.

Figure 14.29: Rebeladmin.com recognized as the federated domain

Now we need to test the login process. To do that:

1. Go to `https://office.com`.
2. Then enter `usera@rebeladmin.com` as the user name. This account is synced to Azure AD from on-prem Active Directory.

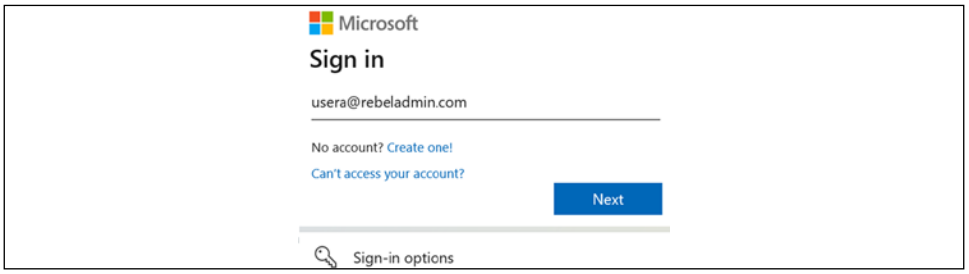


Figure 14.30: Azure AD user login page

3. When I clicked on **Next**, as expected, I was redirected to the AD FS login form page. In there, I have entered the user name and password and clicked on the **Sign in** button.

Figure 14.31: User redirected to the AD FS login form

Note: The way the user is authenticated depends on the AD FS authentication method. In this demo setup, I am only using **Forms Authentication** for extranet and intranet.

4. After a successful login, I was redirected to the `Office.com` home page as expected.

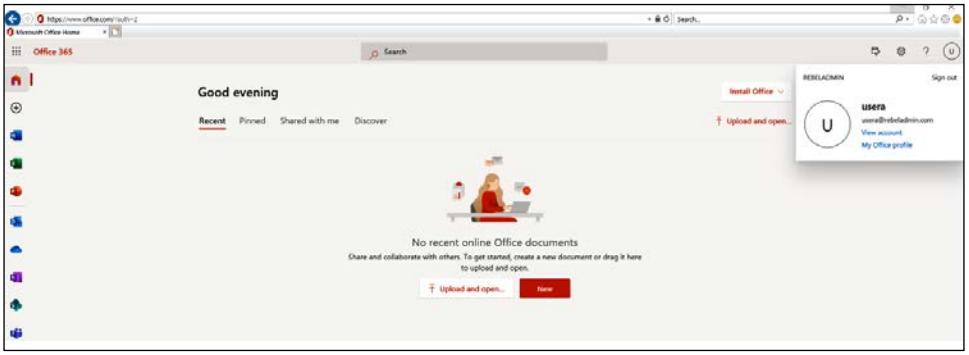


Figure 14.32: User access to the Office 365 portal

As we can see here, the Azure AD federation with AD FS is working successfully. This marks the end of the configuration process as well as the chapter.

Summary

We started this chapter by learning about the characteristics of each of the AD FS versions. This allows us to plan for version upgrades and get the benefits from the new features. AD FS deployment topologies change according to business requirements.

In this chapter, we also learned about different topologies, their characteristics, and their advantages and disadvantages. With the help of that, you have now learned about how to select the best topology based on business requirements. Not only did we go through the theory, but we also went through AD FS deployment using a single federation server and a single web application proxy server model.

MFA is a basic security requirement for public-facing web services. Azure MFA was first introduced to provide multi-factor protection to Azure services and later developed further to support on-prem workload protections. Prior to AD FS 2016/2019/2022, it was a complicated process to implement Azure MFA for AD FS. AD FS 2016/2019/2022 now has built-in Azure MFA integration support. Toward the end of the chapter, we learned about how we can integrate AD FS 2022 with Azure MFA successfully.

Last but not least, we learned how to enable federation between Azure AD and on-premises AD FS.

In the next chapter, we will look into another AD role service, the **Active Directory Rights Management Service** (**AD RMS**), which we can use to protect sensitive data in an infrastructure.

15

Active Directory Rights Management Services

Following the invention of the computer, people started to transform analog data into digital formats. It also transformed the way that people accessed data. If someone is in possession of valuable documents, they can put them in a safe, or in any other secure place. In order to access this valuable data, someone needs to physically be there. Digital data is completely different. Even without physically being there, someone could steal valuable data from a computer infrastructure. This is why data security and data governance are so important when it comes to digital data. When the wrong people have access to the wrong data, the consequences can have an impact on people, organizations, or even countries.

The famous WikiLeaks phenomenon is a good example of this. WikiLeaks got access to state secrets, and some of that data was in digital format, such as emails and scanned files. Someone with authority over that data had passed it to WikiLeaks, or they stole the data illegally. Either way, it's obvious that WikiLeaks should not have had access to that data in the first place. Even now, they keep releasing data to the public, and some of that data is even powerful enough to have a negative impact on foreign affairs between countries.

The other concern about data security is related to intellectual property. For example, let's consider a Covid-19 vaccine manufacturer. They invest a lot of money and effort to develop a vaccine. Then someone else steals the research data, including formulas, and develops another version of the vaccine purely to make money. In such a situation, the efforts of the scientists involved in developing the original vaccine will be in vain. Also, what about the patients who risk their lives in clinical trials? Most importantly, these fake products can create long-term health issues as the quality of the product degrades.

We all know that there are laws in place to protect companies from these kinds of situations, but companies still have to constantly fight to protect their products from illegal distribution. The same theory is applicable to films, music tracks, box sets, and so on. I've been maintaining my own technical blog (`https://bit.ly/3xkymTS`) for the last 7 years, and I have written more than 400 articles. I also write for Microsoft TechNet. I maintain this blog just to share my experience and knowledge with the public. Over the years, I have seen people use my articles on their own websites without asking permission.

Some time ago, a gentleman sent me a very interesting email. In his email, he complained that I was copying other people's blog posts. He even said I was including the same text as in other blog posts. If I include anything from a website, a white paper, or a book, I always reference the source. So, I was very interested to investigate the complaint further. I asked him for more details, and he sent me links to a post from a website. When I checked, yes, the text was exactly the same, but the gentlemen who made the complaint didn't check the published dates. This site had copied my articles and published them as their own work. These articles even included the same screenshots that I used! In the end, I explained and proved that I was the original author; but that wouldn't be necessary if other people didn't use my posts without my permission. It's just another example of the misuse of digital data.

Governments, data security agencies, application vendors, IT communities, and professionals have come up with laws, rules, applications, services, methodologies, and best practices to protect digital data, but one thing we need to understand is that none of these solutions are going to completely protect the data in an infrastructure. We need to expect a breach somewhere. It is not only external threats that we need to worry about. We also need to consider internal data theft. Many major data breaches have support from internal stakeholders. **Active Directory Rights Management Services** (**AD RMS**) is a server role that helps to protect sensitive data from forwarding, printing, or copying. It also allows us to enforce these rules using corporate policies, in order to ensure that data governance is carried out effectively. In this chapter, we are going to cover the following topics:

- What is AD RMS?
- AD RMS components
- How does AD RMS work?
- How do we deploy AD RMS?
- **Azure Information Protection** (**AIP**) implementation

Before we go into AD RMS configuration, it is important to know the role of AD RMS.

What is AD RMS?

Microsoft took their first approach to **information rights management** (**IRM**) by introducing **Windows Rights Management Services** (**Windows RMS**) with Windows Server 2003. This was fully compliant with **Federal Information Processing Standard** (**FIPS**) **140-1**. The updated version of Windows RMS was renamed AD RMS and reintroduced with Windows Server 2008. It continued to grow with features and was included in later versions. Microsoft also released **Azure Information Protection** (**AIP**), which can be used in hybrid/cloud-only environments to protect confidential data.

As I stated earlier, AD RMS is not the solution for all data security requirements in an organization. There are other areas we need to secure along with data, such as networks, identities, roles, and permissions; traditional perimeter-based defense is no longer valid when it comes to the protection of data and identities. We have to embrace the zero-trust security approach to fight against modern threats. We need to verify access to data and resources explicitly. There shouldn't be differences in the verification process based on user, role, location, department, and so on. We also need to make sure users have least-privilege access to data or resources. Even if there is a breach, this will prevent lateral movement within the infrastructure and getting access to more sensitive data. Last but not least, we need to assume a breach and be prepared for it. We need to think about how we can detect whether there is a breach. How we can prevent these attackers from getting access to sensitive data? More importantly, how can we recover if there is a breach? How will we respond to a breach? The zero-trust approach will not only protect sensitive data in an infrastructure but also improves the overall security posture of the organization.

Then the question will be what AD RMS can do to protect sensitive data. Let's go ahead and look into this with an example. Rebeladmin Inc. has a sales department. The CEO creates a Word document that includes sensitive data about last year's sales, and saves it in a network folder. The only people who have access to it are the CEO and the sales manager. The CEO sends an email to the sales manager, informing him about the file. Access to the network share is protected by **Access Control Lists (ACLs)**. However, if the sales manager can access it, what is preventing him from emailing it to a person in the technical department, or bringing it home with him and sharing it with a third party? AD RMS controls the behavior of data once users have access to it. But this will not prevent data leakage via digital photographs, third-party screen capturing, hard copies, or viruses and malware.

AD RMS can do the following:

- **Follow data with policies (persistent usage rights and conditions)**: NTFS permission and ACLs can only be valid within their operation boundaries.

In my previous example, when the report is inside the `Sales` folder, it can be accessed only by the CEO and the sales manager. However, if it's copied to a local disk and forwarded as an email, it will bypass the NTFS permissions and ACLs. AD RMS uses persistent usage policies that follow the data. Even when it's moved or forwarded, the policies will follow it.

- **Prevent confidential emails from getting into the wrong hands**: Emails are commonly involved in data leakages. Constant news about data leakages confirms the extent to which emails are involved. Once an email has left the outbox, we do not have control over the data, and we do not have any guarantee that it will only be accessed by the intended recipient. AD RMS can prevent the recipient from forwarding, modifying, copying, or printing confidential emails.
- **Prevent data from being accessed by unauthorized people**: Similar to emails, AD RMS can also protect confidential files and reports from being modified, copied, forwarded, or printed by unauthorized users.
- **Prevent users from capturing content using the Windows print screen feature**: Even if users do not use forwarding or copying methods to send data, they can still use the **print screen** option to capture the data in another form. AD RMS can prevent users from using the Windows print screen tool to capture data.
- **File expiration**: AD RMS allows you to set a validity period for files. When the time is up, the file will not be able to be accessed.
- **Protect data on mobile devices and macOS**: People use mobile devices to access corporate services and data. AD RMS's mobile extension allows you to extend its data protection capabilities to mobile devices. It supports any mobile device that runs Windows, Android, or iOS. To use it, the device should have the latest RMS clients and RMS-aware apps installed. This also applies to macOS devices with Office 2016/2019 for macOS and RMS-aware applications.
- **Integration with applications**: AD RMS not only supports Microsoft Office files but also supports a wide range of applications and file types. For example, AD RMS can directly integrate with SharePoint (version 2007 onward) in order to protect the documents published on an intranet site.

AD RMS supports many other third-party applications and file types such as `.pdf`, `.jpg`, `.txt`, and `.xml`. This allows a corporate network to protect more and more data types in the infrastructure.

An AD RMS operation is supported in three different operation boundaries:

- **Internal**: In this scenario, AD RMS is used only to protect data in an internal network. Policies are applied to internal applications, files, and users. It will not consider how the data will be protected when it leaves the company premises. This will not support corporate data on mobile devices. This mode is suitable for a testing environment or small businesses.
- **Partner networks**: Businesses are collaborating with each other more and more now. Accelerated digital transformation plays an important role there. Based on the business requirements, there can be AD trusts between multiple forests. In this scenario, the company will focus on protecting data that is exchanged between partners. This can be one-way or both ways. If it's both ways, both infrastructures will have to have their own AD RMS servers.
- **External**: In this scenario, AD RMS operations are focused on protecting the company's data, which is shared across infrastructures. It can either be a completely remote infrastructure or a federated infrastructure. Corporate data protection on mobile devices also needs this type of AD RMS deployment. In this mode, certain AD RMS components will need to be hosted in the perimeter network, and allow service URLs to be accessed via the internet.

Even though there are three main AD RMS deployment modes, it doesn't prevent organizations from setting up AD RMS environments to cover two, or all three, deployment modes. It is easily possible to extend AD RMS operation boundaries based on the organization's requirements. This means that organizations can start with protecting internal data, and then extend it to protect data in a partner environment, or external environments, when required.

The following table lists the applications, file types, and scenarios we can use to protect data using AD RMS:

Requirements	Application	Protected file types

Protect confidential files	**Microsoft Outlook Windows (version 2003 onward)**
	Office for macOS 2016:
	Word
	Excel
	PowerPoint
	PDF
	Image processing
	XPS Viewer
	Gaaiho Doc
	GigaTrust Desktop PDF Client for Adobe
	Foxit PDF Reader
	Nitro PDF Reader
	Siemens JT2Go
		`.doc, .docm, .docx, .dot, .dotm,`
		`.dotx, .potm, .potx, .pps, .ppsm,`
		`.ppsx, .ppt, .pptm, .txt, .xml, .jpg, .jpeg, .pdf, .png, .tif, .tiff, .bmp, .gif, .jpe, .jfif, .jt, .xps`

Protect confidential emails	**Microsoft Outlook Windows (version 2003 and newer)** **Office for macOS 2016** **Microsoft Exchange 2007 SP1**	`.msg`

Protect content on the intranet	**Microsoft SharePoint 2007 and newer**	`.doc, .docm, .docx, .dot, .dotm,`
		`.dotx, .potm, .potx, .pps, .ppsm,`
		`.ppsx, .ppt, .pptm, .txt, .xml, .jpg, .jpeg, .pdf, .png, .tif, .tiff, .bmp, .gif, .jpe, .jfif, .jt, .xps`

Protect mobile data	**Microsoft Word** **Microsoft Word app** **Microsoft Excel app** **Microsoft Outlook app** **Microsoft PowerPoint app** **WordPad** **Office for macOS 2016** **Word Online** **TITUS Docs** **Foxit Reader**	`.doc, .docm, .docx, .dot, .dotm, .dotx, .potm, .potx, .pps, .ppsm, .ppsx, .ppt, .pptm, .txt, .xml, .jpg, .jpeg, .pdf, .png, .tif, .tiff, .bmp, .gif, .jpe, .jfif, .jt, .msg, .xps`

The above list can regularly be updated with new releases. Also, more third-party applications can appear in the list.

AD RMS components

AD RMS has its own role services and related components that need to work together in order to maintain a healthy AD RMS environment:

- Active Directory Domain Services (AD DS)
- The AD RMS cluster
- Web server
- SQL Server
- The AD RMS client
- Active Directory Certificate Service (AD CS)

Let's look into each of these components in detail.

Active Directory Domain Services (AD DS)

AD RMS is one of the AD role services. AD RMS can be installed only in an AD DS environment. As a part of the setup, a **service connection point** (**SCP**) will need to be published via AD. It will help users to discover the service URLs for the AD RMS environment.

The AD RMS cluster

The AD RMS cluster is a single RMS server or a group of servers that share certificates and licensing requests from their clients. Even though it is named *cluster*, it is different from a typical Windows failover cluster. The failover cluster needs at least two nodes. However, with RMS, even though it has a single server, it is called a **cluster**. But there is one requirement for the AD RMS cluster if there are multiple servers involved. AD RMS supports two types of databases similar to **Active Directory Federation Services** (**AD FS**). By default, it uses **Windows Internal Database** (**WID**), and it also supports Microsoft SQL Server databases. If the AD RMS cluster is going to have multiple servers, it must use Microsoft SQL.

There are two types of clusters in AD RMS:

- **The root cluster**: When we deploy the first AD RMS server in the infrastructure, it becomes the root cluster. By default, it responds to both licensing and certificate requests from clients. When required, additional RMS servers can be added to the cluster. Only one root cluster can exist in one AD forest.
- **The licensing cluster**: If an organization has multiple sites, it is always best to use local services, especially if the sites are connected through slow links. It improves service performance, as well as reliability. In such scenarios, organizations can deploy licensing-only clusters on remote sites to serve licensing requests.

It is recommended that the root cluster is used whenever possible, as it will automatically load balance both certificates and licensing requests. When a system has two clusters, load balancing is handled by each cluster separately, even though they are components of one system.

Web server

AD RMS requires **Internet Information Services** (**IIS**) with the following role services:

- Web server (IIS)
- Web server:
    - Common HTTP features:
        - Static content
        - Directory browsing
        - HTTP errors
        - HTTP redirection

        - Performance:
            - Static content compression
        - Health and diagnostics:
            - HTTP logging
            - Logging tools
            - Request monitor
            - Tracing
        - Security:
            - Windows authentication
- Management tools:
    - IIS Management Console
    - IIS 6 Management Compatibility:
        - IIS 6 Metabase Compatibility
        - IIS 6 WMI Compatibility

These features can be added during the role installation.

SQL Server

AD RMS supports WID and Microsoft SQL Server databases (SQL Server 2005 onward). If an AD RMS cluster is going to use multiple servers, the database must be running on Microsoft SQL. AD RMS uses three databases:

- **Configuration database**: This includes configuration data that is related to the AD RMS cluster, Windows users' identities, and the AD RMS certificate key pair that is used to create a cluster.
- **Logging database**: This contains the logging data for the AD RMS setup. By default, it will install it in the same SQL Server instance that hosts the configuration database.
- **Directory service database**: This database maintains cached data about users, SID values, group membership, and related identifiers. This data is collected by the AD RMS licensing service through **Lightweight Directory Access Protocol** (**LDAP**) queries that run against the global catalog server. By default, it's refreshed every 12 hours.

AD RMS supports SQL high-availability solutions, including SQL failover clustering, database mirroring, and log shipping. However, it *does not* support SQL Server Always On.

Earlier in this chapter, I mentioned that mobile device extensions can be used to extend AD RMS in order to manage corporate data in mobile devices. If you are going to use this feature, AD RMS databases must use Microsoft SQL.

The AD RMS client

We need the AD RMS client to communicate with the AD RMS cluster. This is included in all recent operating systems that were released after Windows XP. However, it still needs to be installed manually on macOS and mobile devices to use AD RMS.

Active Directory Certificate Service (AD CS)

AD RMS uses several certificates to protect communication between AD RMS components and clients. Most of these can be issued using a corporate trusted **certificate authority** (**CA**). For example, the AD RMS cluster can be built using an SSL certificate to protect communication between servers in a cluster. If the AD RMS setup is required to externally publish service URLs, then it will require a certificate from the public CA.

AD RMS itself uses various **eXtensible rights Markup Language** (**XrML**)-based certificates to protect communication between components and data. These certificates are different from AD CS certificates.

How does AD RMS work?

By now, we know the components of AD RMS and their responsibilities. In this section, we are going to learn in detail how all these components work together in order to protect sensitive corporate data.

Before we start the data protection process, we need a healthy AD RMS cluster, AD RMS clients (author and recipient), and a reliable connection between these components. Once these prerequisites are fulfilled, the data protection process will go through three main stages: protecting the author's content, publishing the protected content, and accessing the protected content (recipient).

Let's assume Peter is trying to protect a document using AD RMS. He is going to send it to Adam, but he does not want him to edit or print it. This is the first time he is going to use AD RMS. In an AD RMS environment, the user, Peter, will be referred to as an **information author**. In his first authentication into the AD RMS cluster, a **rights account certificate** (**RAC**) is created, which will be the user's identity in AD RMS. This is a one-time process.

This certificate contains Peter's public key, and his private key (which is encrypted by his computer's public key). When Peter registers with the AD RMS cluster, another certificate called the **client licensor certificate** (**CLC**) is also created. The CLC includes the CLC's public key and private key, which are protected by Peter's public key. It also includes the AD RMS cluster's public key, which is signed by the AD RMS private key.

Peter decides what data needs to be protected first. Then, a symmetric key (random) is generated, which encrypts the data that needs to be protected. It uses AES-256 standards to encrypt the data. When the first AD RMS server is added to the cluster, it creates another certificate called the **server licensor certificate** (**SLC**). This represents the identity element of the AD RMS server. This is shared with clients so that they can use it to exchange confidential data in a secure way. The SLC includes the public key of the AD RMS server. In the next step, the system will encrypt the symmetric key that is used for the encryption of the data using the AD RMS server public key. Therefore, only the AD RMS cluster can open it.

After that, the RMS client creates the **publishing license** (**PL**). The PL is used to indicate to the allowed recipients what rights they have, and what conditions will apply to the protected data. The PL includes an encrypted symmetric key that can be used to decrypt the protected data. All this data is then encrypted with the SLC's public key. Apart from that, the AD RMS client will also sign the encrypted data with the private key of the SLC. In the end, this protected data will be attached to the PL. It also includes the copy of the symmetric key that is encrypted with the SLC public key. This confirms Peter's authority over the protected document, so he can decrypt the document without using another license. Once all these encryptions and signings are done, the document is ready to be sent over to Adam.

Once Adam receives the document, his AD RMS-aware application tries to open it, and finds out that it is a protected document. Similar to Peter, Adam already has his RAC and CLC from the AD RMS cluster. To open the protected document at once, does it need to be decrypted or signed with any of Adam's certificates? No, it does not. But his AD RMS client knows who needs to be contacted in order to sort it out for him. To open the protected document, Adam should have a **Use License** (**UL**). This is issued by the RMS cluster. So, the AD RMS client request for the license also includes the encrypted PL, the encrypted symmetric key, Peter's CLC, and the public key of Adam's RAC. The protected document will not be sent over with this request to the RMS cluster.

To decrypt the protected document, Adam needs the symmetric key that was used by Peter to encrypt the document. As a first step, the server needs to know whether Adam is permitted to access the document; if he is permitted, what sort of conditions and rights will apply? This information is in the PL. It is encrypted using the public key of the SLC.

The AD RMS server is the private key owner for it, and can easily extract it. If Adam is not allowed in the PL, he will be declined access to it. If he is allowed in the PL, it creates a list that states Adam's rights over the document.

The most important part of the decryption process is to retrieve the symmetric key. This is also encrypted by the SLC's public key. Once it is extracted, it will be re-encrypted using Adam's RAC public key. This was a part of the UL request, and it ensures that only Adam's system can see the key. Since the server has all the required information, it generates the UL, including the permission list and the encrypted symmetric key. Then, it sends it over to Adam's RMS client. Once it reaches Adam's system, it can decrypt the symmetric key using the RAC's private key. Then, the RMS-aware application will decrypt the document and attach the rights information retrieved from the UL. In the end, voilà! Adam can see the content of the document.

In the preceding example, we saw different certificates, licenses, and data encryption and decryption. But I think it would still be better to explain it at a higher level, in order to recap the things you learned:

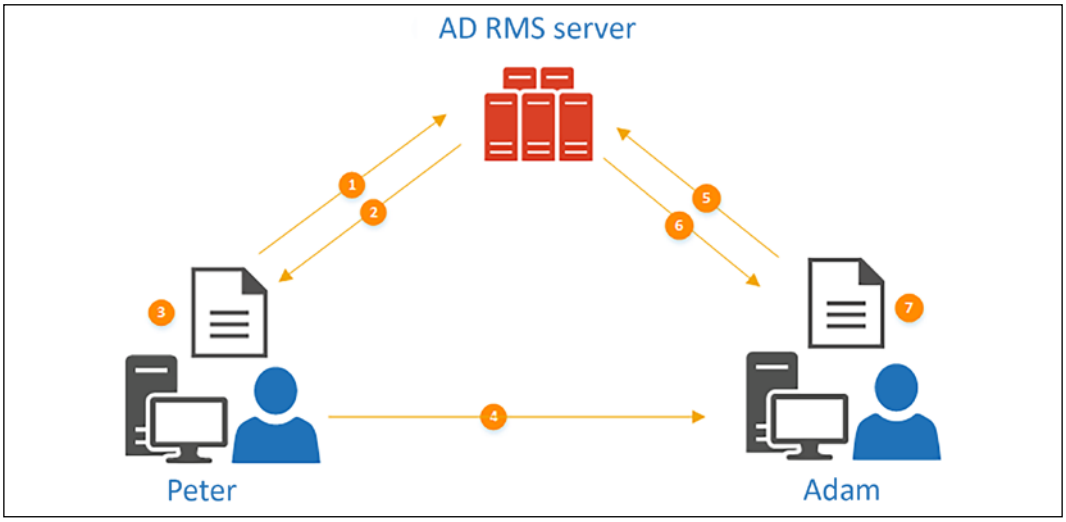


Figure 15.1: AD RMS in action

Peter wants to send the protected document to **Adam**. **Adam** should have read-only permissions for the document; he should not be able to modify or print it:

1. This is the first time that **Peter** is going to use the **AD RMS server**. As soon as he tries to protect the document, the RMS client initiates a connection to the **AD RMS server** (cluster).

2. The **AD RMS server** replies with the RAC and the CLC. This is a one-time process.
3. In **Peter**'s system, a random symmetric key is generated, which is used to encrypt the document. Then, this symmetric key is encrypted using the SLC's public key. After that, it is attached to a PL, which includes **Adam**'s rights for the protected document. After that, the PL is attached to the encrypted document.
4. **Peter** sends the protected document (along with this additional info) to **Adam**.
5. **Adam**'s RMS-aware application tries to open it and finds that it needs the UL from the **AD RMS server**. Then, the RMS client requests it from the RMS server.
6. The RMS server decrypts the symmetric key and the PL. After that, the server checks whether the requester matches the PL. In our scenario, it matches, so it goes ahead and creates the UL. This includes the symmetric key (it re-encrypts using **Adam**'s RAC public key), and a list that contains the rights described in the PL. Then, it delivers it to **Adam**'s system.
7. Once **Adam**'s system receives the UL, it retrieves the symmetric key and decrypts the document. Then, **Adam** opens the document and uses it according to the rights described in the PL.

In the next section of this chapter, we are going to look into different models that we can use to deploy AD RMS.

How do we deploy AD RMS?

AD RMS deployment topologies are a bit different from other AD role service deployments. Other AD role service deployment topologies are mostly focused on high availability or scalability. But AD RMS deployments are more about addressing different types of business requirements. Let's look into these topologies in detail.

Single forest-single cluster

This is the most commonly used deployment topology. In this setup, AD RMS operations will be limited to an AD forest. The deployment will only have one AD RMS cluster to process certificates and licensing requirements. The cluster can contain any number of servers, and load balancing is handled at the cluster level. If it has multiple servers, the AD RMS cluster should use a Microsoft SQL Server database, instead of WID. This deployment model will not consider extending data protection to non-corporate networks.

The following table lists the advantages and disadvantages of a single forest-single cluster:

Advantages	**Disadvantages**
Easy to implement. Fewer resources (servers and licenses) are used.	Can protect data only within a limited environment.
System maintenance and management are easy due to a smaller operation boundary.	N/A.
Support for extending into any other deployment topologies.	N/A.

Even though this is the easiest model to implement and maintain, not every organization is going to fit into this model. Let's go ahead and see some other models that can contribute to complex business needs.

Single forest-multiple clusters

This is the extended configuration of the single forest-single cluster topology. There are two types of AD RMS clusters. The AD RMS root cluster is the default cluster, and it answers both certificates and licensing requests. The licensing-only cluster can also be deployed in the same forest, and it will respond to licensing requests only. This suits infrastructures that have sites in different geographical locations. Then, it will rule out the requirement of contacting RMS clusters via slow links. Instead, it will use a license-only cluster on each site for licenses. There is only one root cluster for the forest but it can have multiple licensing-only clusters.

Based on the role installed in the AD RMS server, it will decide which cluster it will be a part of. However, unless there is a special requirement, it is recommended that you use the root cluster only. Load balancing between member servers is handled at the cluster level, and its configurations are independent. They cannot be shared between different clusters. In this topology, it is important to also have Microsoft SQL high availability.

The following table lists the advantages and disadvantages of this topology:

Advantages	**Disadvantages**
Remote sites do not need to depend on the site links and bandwidth. The local RMS license-only cluster will process the license requests.	Complexity – deployment requires advanced planning and configuration.
Complies with government rules to use localized encryption tools and technologies.	High cost – need to use additional resources and licenses for deployment. Also increases the maintenance cost.

N/A.	Distributed management – when placing clusters on remote sites, it may also need to grant privileges to the site's IT team in order to manage and maintain the system. This can have an impact on security and system integrity.

Still, we only considered single AD forest environments. But what if an organization has multiple forests?

AD RMS in multiple forests

If the organization has multiple AD forests, and if AD RMS needs to be used between them in order to protect data, this deployment method can be used. Each forest can only have one RMS root cluster. Therefore, in multiple forest environments, each domain should have its own AD RMS cluster. The AD RMS cluster uses AD DS to query an object's identity. When there are multiple forests, it needs to have contact objects of users and groups for the remote forest. The following elements are required for AD RMS deployment in multiple forests:

- An AD RMS root cluster in each forest
- Contact objects for remote users and groups (from the different forests) need to be set up
- Schema extensions need to be in place to trace back to the mother forest of the contact objects
- Attribute values of contact objects are needed to sync with the mother forest, so that when required, these values can be used to trace back to the original object

AD trust between forests is not a must. It can have two-way trust, one-way trust, or no trust at all. If there is trust, it will simplify the process of managing the permissions and validations of cross-forest objects.

The following table lists the advantages and disadvantages of this:

Advantages	**Disadvantages**
Extended data protection boundaries	Complexity – deployment requires advanced planning and conflagration.
Standardized data protection with partners in order to protect confidential data	Dependencies – the success of the solution depends on the system configuration and the availability of the AD RMS components of the partner's forest. It can also have dependencies where the primary forest does not have control.

In this topology, AD RMS trust policies control how licensing requests and data protection between different AD RMS clusters are handled. There are two types of trust policies:

- **Trusted AD RMS domains**: This allows the AD RMS root cluster to process requests for the CLC or the UL from users who have an RAC issued by a different AD RMS root cluster. We can add a trusted AD RMS domain by importing the SLC of the AD RMS cluster to be trusted.
- **Trusted publishing domains**: This allows one AD RMS cluster to issue ULs for PLs that were issued by a different AD RMS cluster. We can add a trusted publishing domain by importing the SLC and the private key of the server to be trusted.

AD RMS in a multiple-forest topology needs an AD RMS root cluster in each forest. This topology doesn't require trust between forests but if there is trust, it makes it easier to manage permissions. But, not every partner or business wants trust between forests. They may want to use AD RMS, but they may not want to maintain the AD RMS cluster or create trust between forests. So in such a situation, how we can do this?

AD RMS with AD FS

AD FS allows an organization to use the already deployed AD RMS cluster in a remote forest. AD FS allows user accounts to use their own credentials, established by a federated trust relationship.

Before we set it up, we need to fulfill certain prerequisites that are required between federated infrastructures. Refer to `https://technet.microsoft.com/en-us/library/dn758110(v=ws.11).aspx` for more details on the configuration.

The following table lists the advantages and disadvantages of this:

Advantages	**Disadvantages**
No need to maintain multiple AD RMS clusters between organizations. It can use an already existing federation trust to use AD RMS in other forests.	There are security concerns, as it's possible to spoof someone's user account and access protected data through a federation proxy.
Extended data protection boundaries and implementation is less complex.	If an internal CA is used to make SSL-based trusts (AD RMS and AD FMS), the federated domain should be configured to trust the root CA (using GPO to publish the root cert). Otherwise, they may need to invest in using public certificates.

Fewer system dependencies between infrastructures.	N/A.

If it is a hybrid environment, the use of Azure RMS and AIP on corporate data is recommended. These can be used to protect data in both environments. Since it is a hosted service, we do not need to worry about resources, maintenance, and dependencies.

AD RMS configuration

In this section, we are going to look into the configuration of AD RMS 2022, and see how it really works. The demonstration environment that it uses was built using Windows Server 2022 with the latest updates. The AD DS forest and domain functional levels are also set to Windows Server 2016.

Setting up an AD RMS root cluster

AD RMS can only be installed on a domain member server. I have a demonstration server, which is already a member server of the domain. The first AD RMS server that is added to the forest creates the AD RMS cluster.

Installing the AD RMS role

The following are the steps that are needed in order to install the AD RMS role:

1. Log in to the server as Enterprise Admin.
2. Install the AD RMS role and related management tools using the following command:

```
Install-WindowsFeature ADRMS -IncludeManagementTools
```

Once the role is installed, we can proceed with the initial configuration.

Configuring the AD RMS role

The following are the steps that are needed in order to configure the AD RMS role:

1. Launch Server Manager and go to the notifications icon by navigating to **Configuration required for AD Rights Management Services | Perform additional configuration**; this will open the AD RMS configuration wizard.

Click on **Next** to start the configuration:

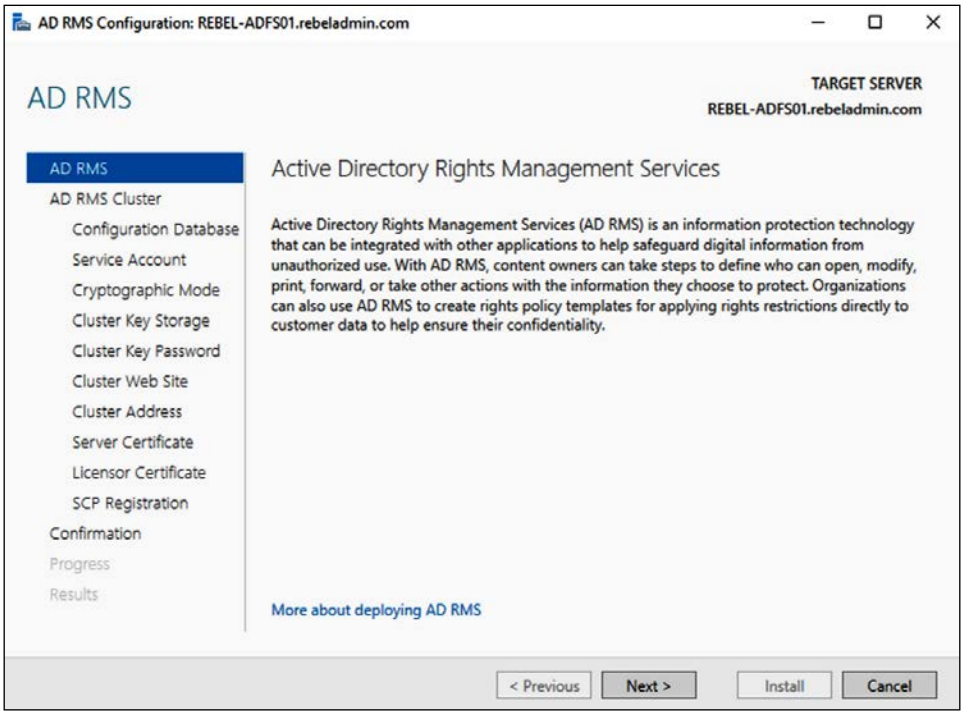


Figure 15.2: AD RMS configuration wizard

2. On the next screen, there is the option to create a new AD RMS root cluster or to join it to the existing AD RMS cluster. Since it is a new cluster, select the **Create a new AD RMS root cluster** option and click on **Next**.

3. The next screen defines the AD RMS database configuration. If it's going to use the MS SQL Server, select **Specify a database server and a database instance**, or else select **Use Windows Internal Database on this server**. Note that if **WID** is used, it cannot have any more AD RMS servers, and it cannot have the AD RMS mobile extension either. Since this is for testing purposes, I am going to use WID. Once the selection has been made, click on **Next** to move to the next step:

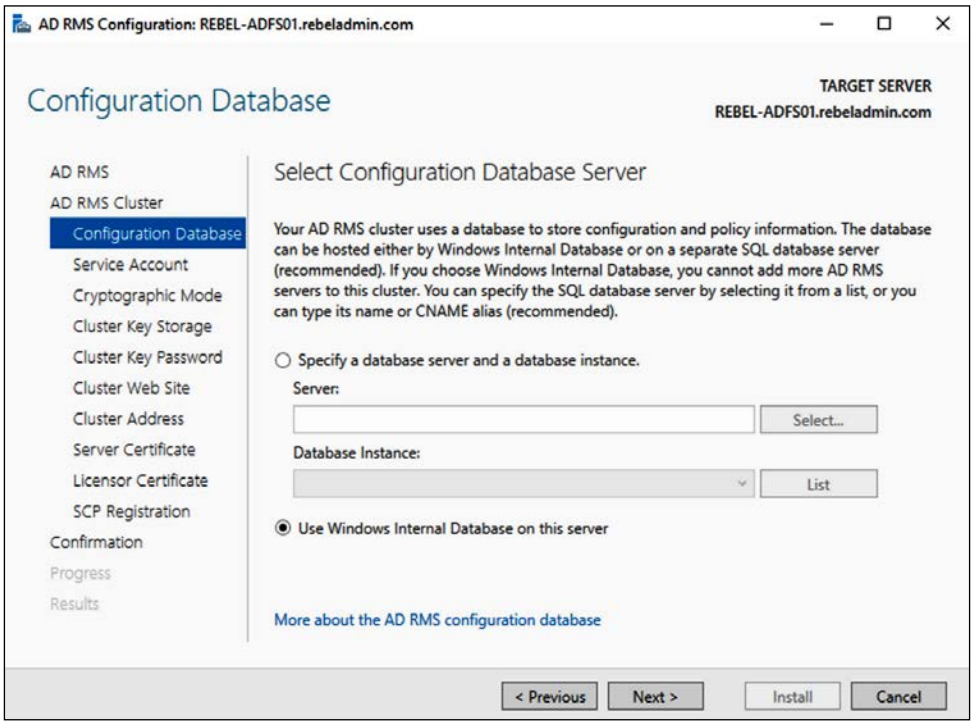


Figure 15.3: Creating WID for AD RMS

4. In the next window, we need to define the service account. This service account is used to communicate with other services and computers. This doesn't need to have Domain or Enterprise Admin rights. To configure it, click on **Specify...** and provide the username and password for the account. Then, click on **Next** to proceed:

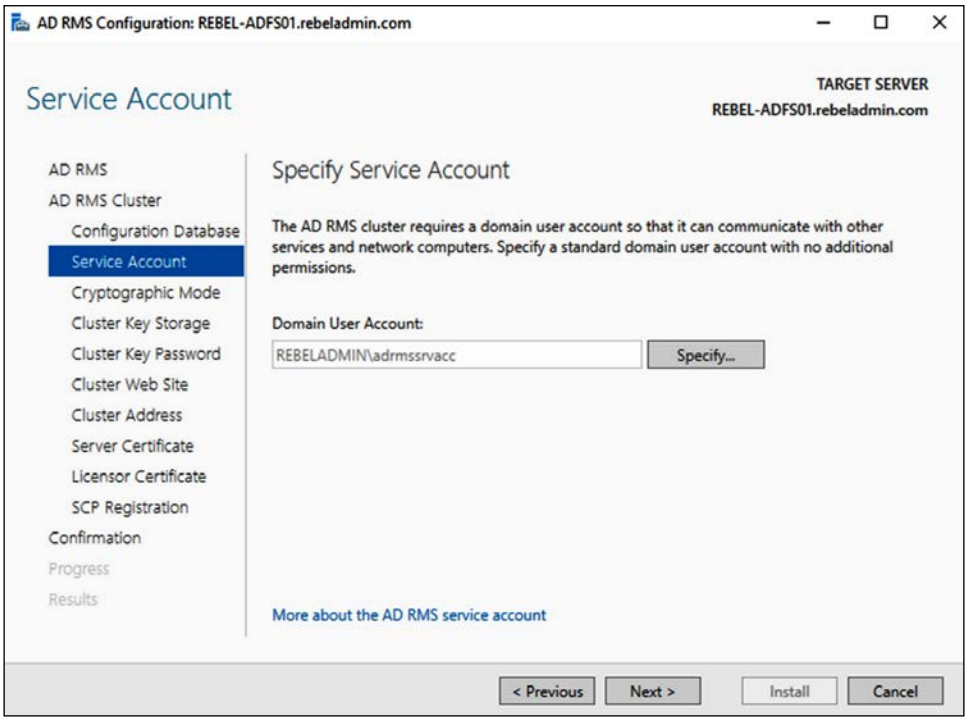


Figure 15.4: Defining the AD RMS service account

5. On the next screen, we need to select the cryptographic mode. This defines the strength of the hashes. AD RMS supports two modes, which are SHA-1 and SHA-256. It is highly recommended to use **Cryptographic Mode 2 (RSA 2048-bit keys/SHA-256 hashes)**, which uses SHA-256 for stronger hashing. However, if multiple clusters are in place, all clusters need to use the same cryptographic mode. In our setup, I am going to use the default SHA-256. Once the selection is made, click on **Next** to proceed:

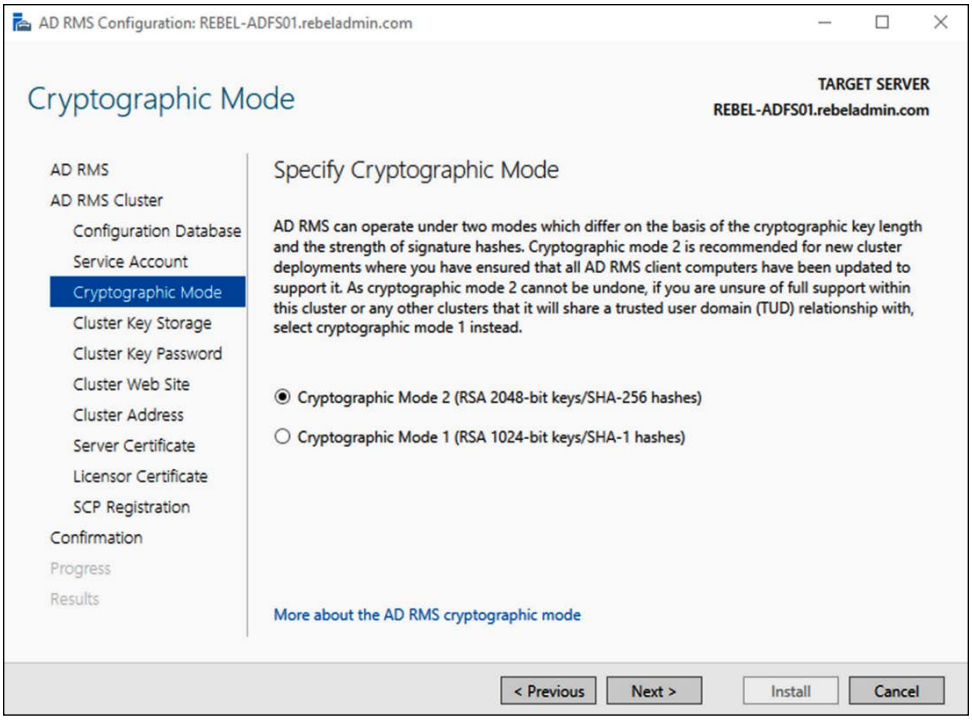


Figure 15.5: Defining the AD RMS cryptographic mode

6. AD RMS uses the cluster key to sign the certificate and licenses that it issues. This is also required when AD RMS is restored, or when the new AD RMS server is added to the same cluster. It can be saved in two places. The default method is to save the cluster key in a centrally managed key store. It also supports the use of the **cryptographic service provider** (**CSP**) as storage, but this requires the manual distribution of a key when adding another AD RMS server to the cluster. In this demonstration, we will use the **Use AD RMS centrally managed key storage** option. Once the selection is made, click on **Next** to proceed.
7. AD RMS also uses a password to encrypt the cluster key described in the preceding step. This cluster key password will be used when adding another AD RMS server to the cluster, or when restoring AD RMS from a backup. This password cannot be reset. Therefore, it is recommended that you keep it recorded in a secure place. Once you define the AD RMS cluster key password, click on **Next** to proceed.

8. In the next step, we need to define the IIS virtual directory for the AD RMS website. Unless there is a specific requirement, always use the default, and click on **Next**:

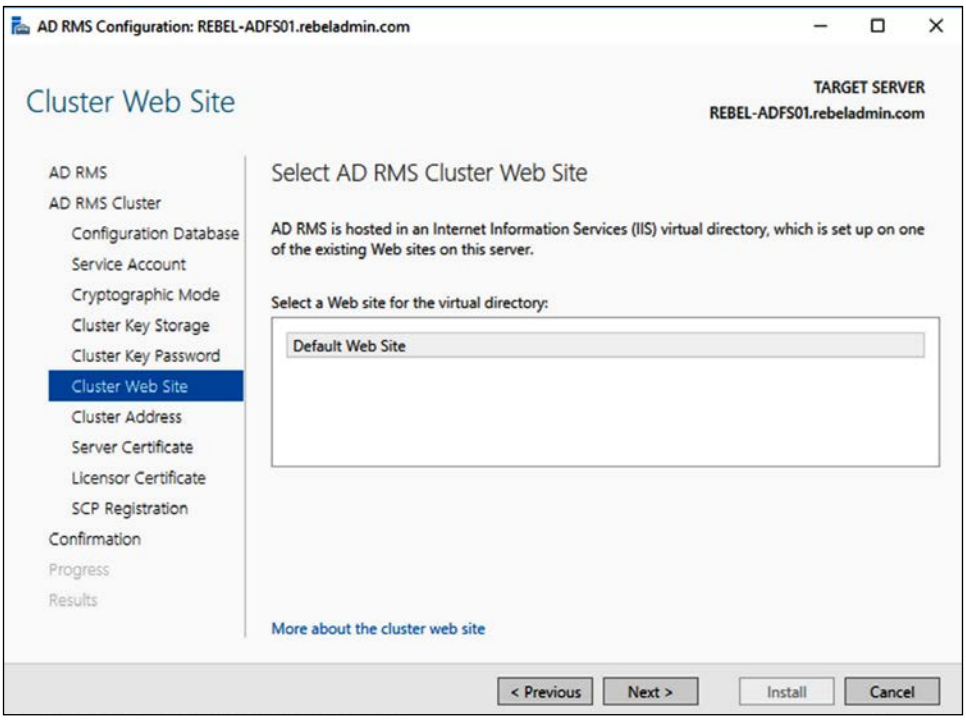


Figure 15.6: AD RMS cluster website settings

9. In the next step, we need to define an AD RMS cluster URL. This will be used by AD RMS clients to communicate with the AD RMS cluster. It is highly recommended that you use SSL for this, even if it allows it to be used with the HTTP-only method. The related DNS records and firewall rules need to be created to provide a connection between AD RMS clients and this URL (internally or externally).

Once the configuration values are provided, click on **Next** to proceed. One thing you need to note is that once this URL is specified, it cannot be changed. In this demonstration, I used `https://rms.rebeladmin.com` as the RMS URL:

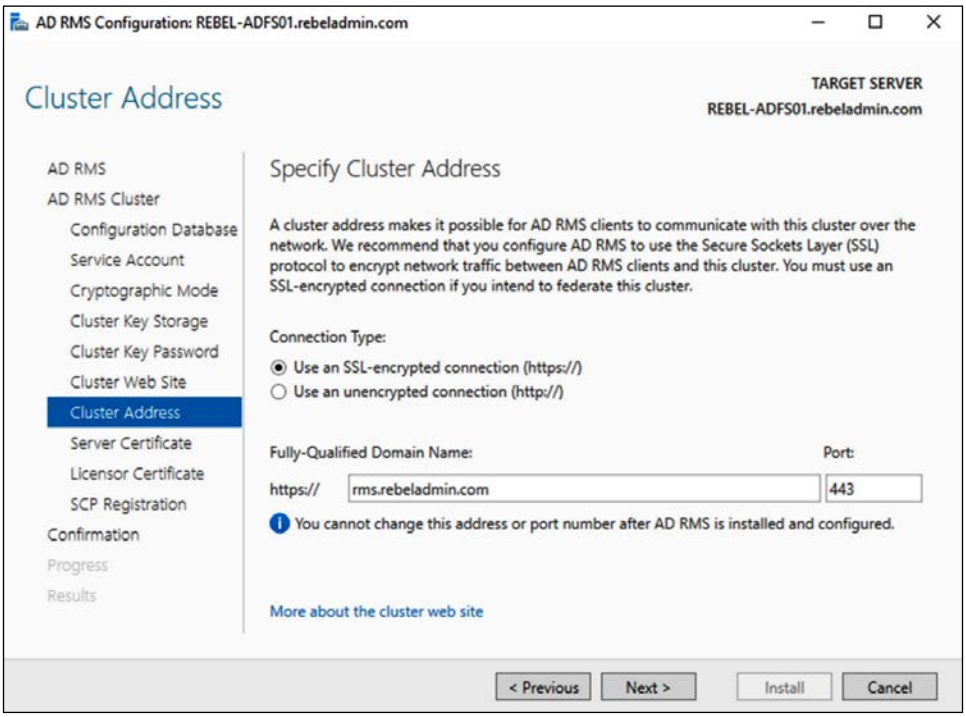


Figure 15.7: AD RMS cluster address

10. In the next step, we need to define a server authentication certificate. This certificate will be used to encrypt the network traffic between RMS clients and the AD RMS cluster. In a testing environment, it can use a self-signed certificate, but this is not recommended for production. If it uses an internal CA, client computers should be aware of the root certificate.

In the wizard, it automatically takes the list of the SSL certificates that are installed on the computer, and we can select the certificate from there. It is also possible to configure this setting at a later time. Once the settings are defined, click on **Next** to proceed:

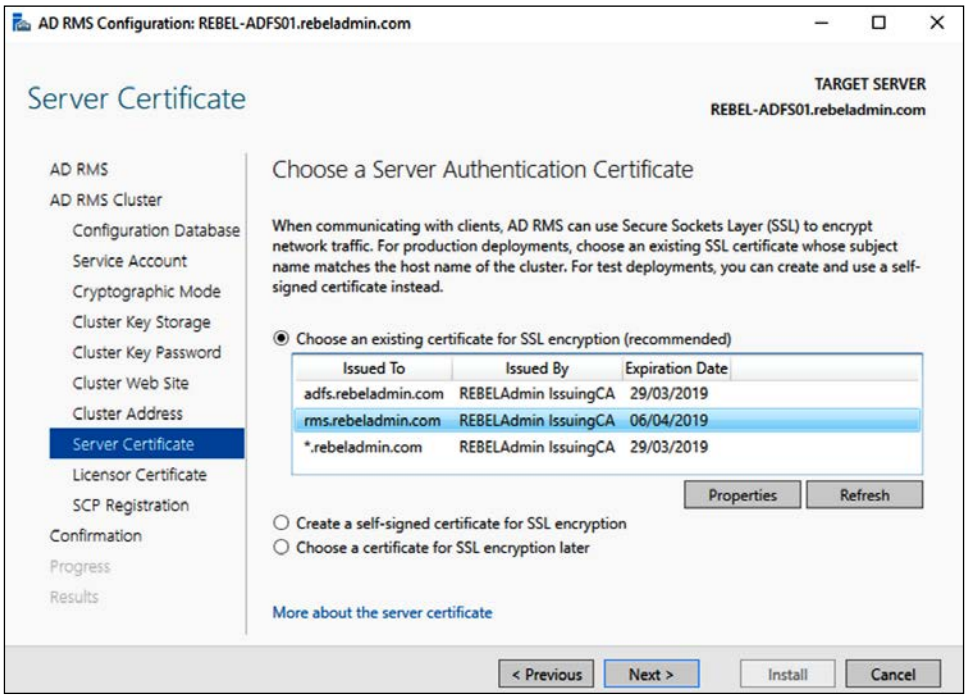


Figure 15.8: Selecting the server authentication certificate

11. On the next window, it asks us to provide a name for the SLC. This certificate is used to define the identity of the AD RMS cluster, and it is used in the data protection process between clients to encrypt/decrypt symmetric keys. Once you've defined a meaningful name, click on **Next** to proceed.

12. The last step of the configuration is to register an AD RMS SCP with AD DS. If needed, this can also be configured later. You need Enterprise Admin privileges to register it with AD DS. In this demonstration, I have already logged in as the Enterprise Admin, so I am using the **Register the SCP now** option. Once this option is selected, click on **Next**:

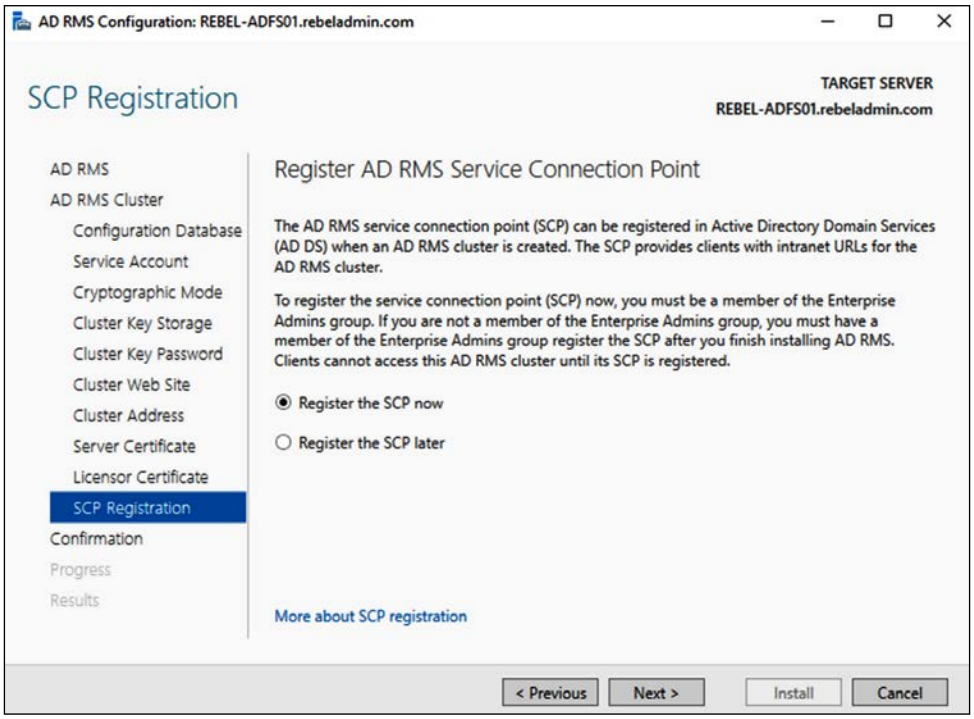


Figure 15.9: Registering an AD RMS SCP

13. After the confirmation, the installation will begin. If all is successful, log out and log back in to the AD RMS server.

14. Once logged back in, select **Server Manager** and navigate to **Tools | Active Directory Rights Management Service** in order to access the AD RMS cluster:

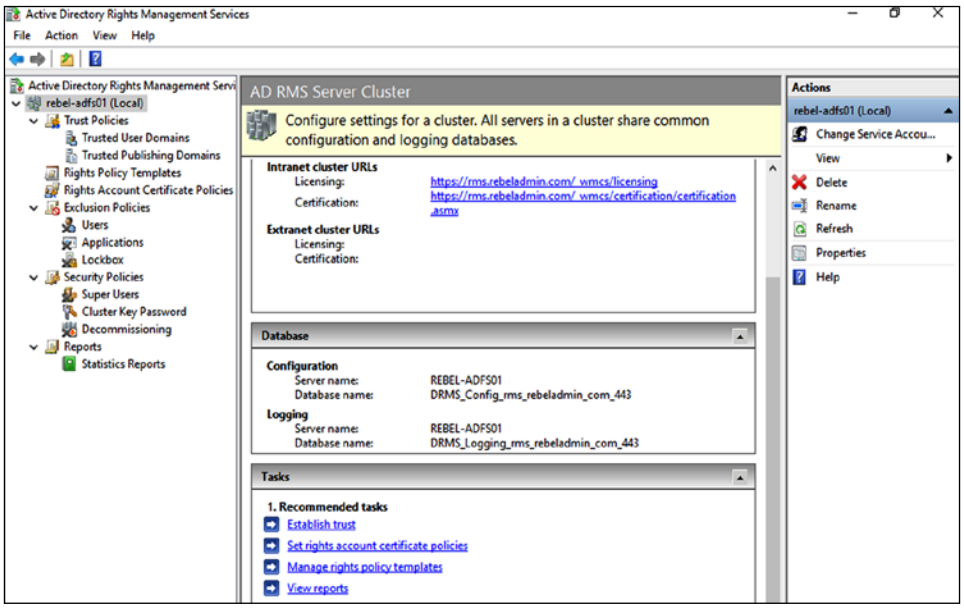


Figure 15.10: AD RMS server cluster

This completes the AD RMS configuration process. The next step will be to test and confirm the solution in place.

Testing – protecting data using the AD RMS cluster

The next step of the demonstration is to test the functionality of the AD RMS cluster by protecting sensitive data. For that, I will use two user accounts, as shown in the following table:

User	**Email address**
Peter	`peter@rebeladmin.com`
Adam	`adam@rebeladmin.com`

The email address field is a must, and if the user doesn't have an email address defined, they will not be allowed to protect the document.

The end user computers must add the `https://rms.rebeladmin.com` RMS service URL to Internet Explorer's local intranet trusted site lists. This can also be done via GPO. If it's not added, when you go to protect the document, you will get the following error:

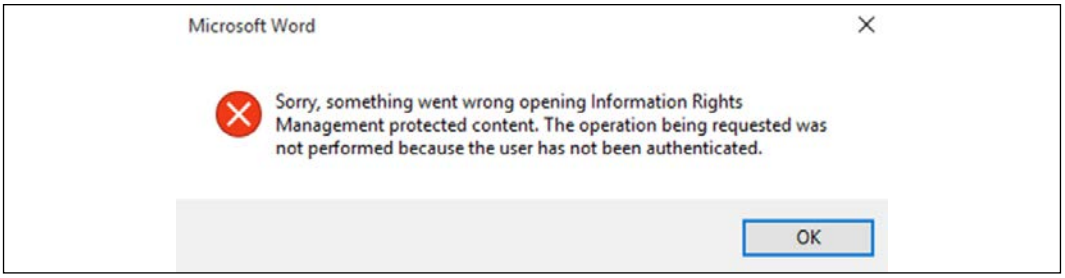


Figure 15.11: Service error due to trusted sites

In this demonstration, Peter is going to create a protected document using Word 2013. He wants Adam to have read-only access to the file. At the same time, he wants to make sure no one else has access to the content of the document.

Testing – applying permissions to the document

The following steps need to be performed in order to protect the document:

1. Log in to the Windows 10 (domain member) computer as the user Peter.
2. Open Word 2013 and type some text.
3. Then, go to **File** | **Protect Document** | **Restrict Access** | **Connect to Rights Management Servers and get templates**:

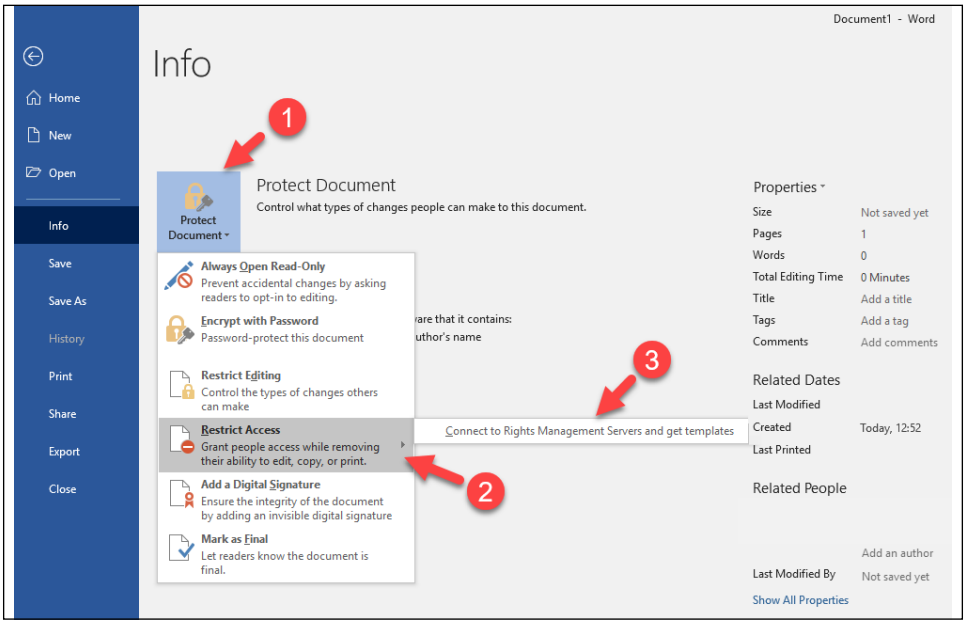


Figure 15.12: Connecting to the AD RMS server to get templates

4. Once you have successfully retrieved the templates, go back to the same option and select **Restricted Access**:

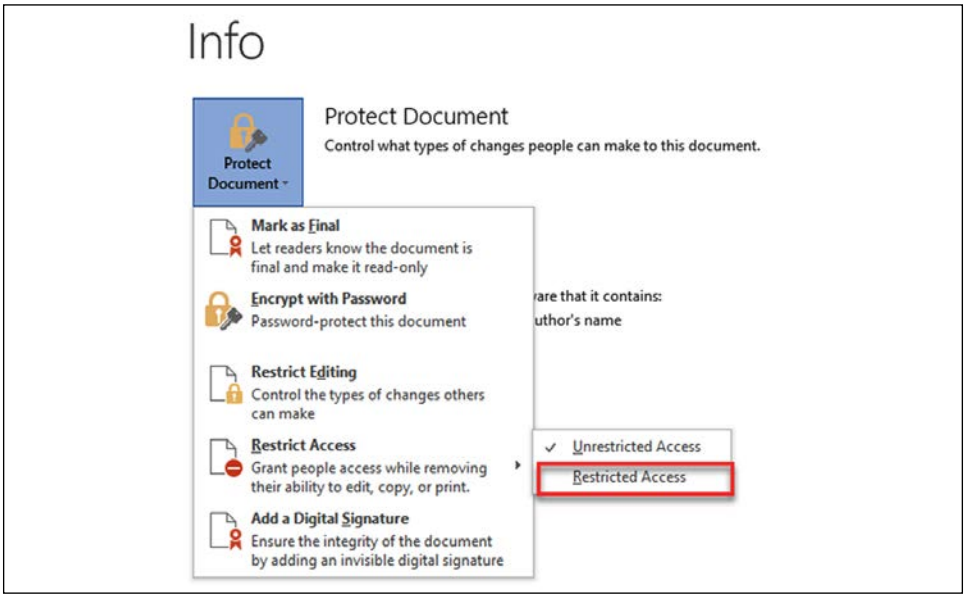


Figure 15.13: Protecting a Word document with AD RMS

5. Then, it will open a new window. Once there, for the read permissions, type `adam@rebeladmin.com` to provide read-only permission to the user Adam. Then, click on **OK**:

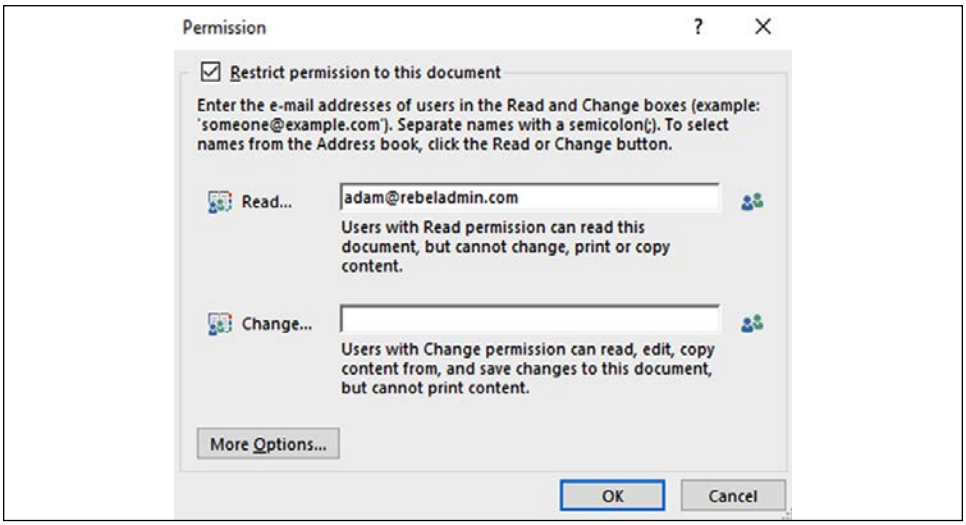


Figure 15.14: Document permissions

6. After that, save the document. In the demonstration, I used a network share that the user Adam also has access to.
7. Now, log in to another Windows 10 computer as the user Adam.
8. Then, browse to the path where the document was saved and open it using Word 2013.
9. In the opening process, the system asks to authenticate for AD RMS to retrieve the licenses. After that, the system opens the document. At the top of the document, it says the document has limited access. When you click on **View Permission...**, it lists the allowed permissions and matches what we set on the author side:

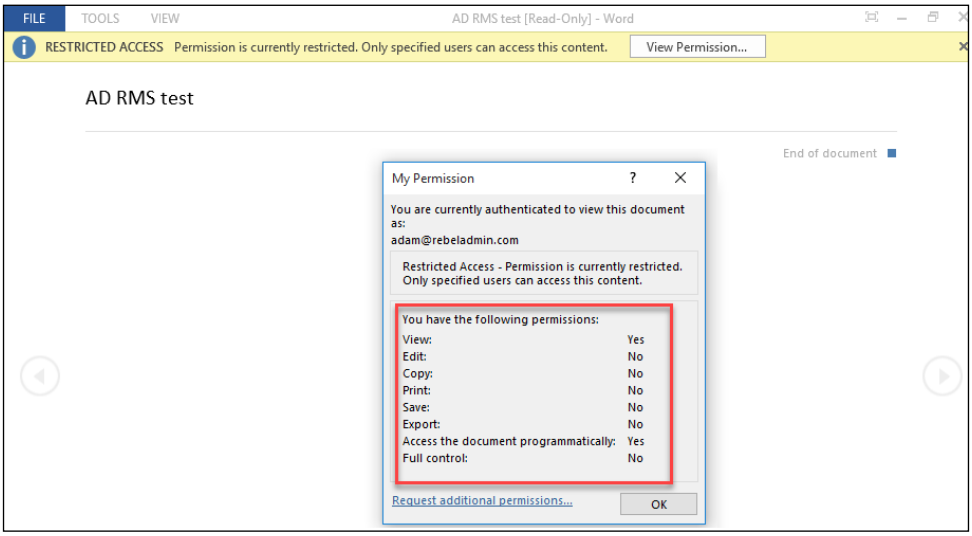


Figure 15.15: Viewing permissions

10. Further into testing, I log in to the system as another user (Steve), and when I access the file, I get the following prompt:

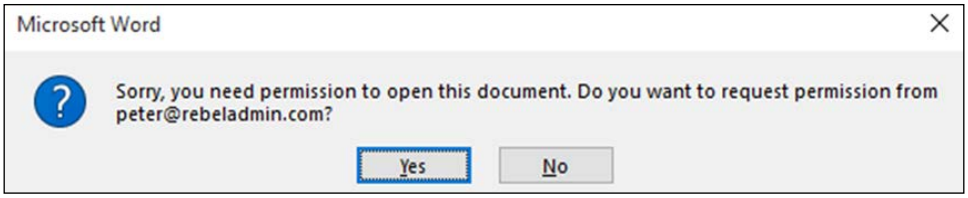


Figure 15.16: Error message

This error is expected as we did not assign permission for the user Steve. This ends the configuration and the testing of the AD RMS cluster. In this demo, I explained how we can set up the AD RMS cluster using minimum resources and configuration.

I only used the default configuration of the AD RMS cluster, and no custom policies were applied. Understanding the core functions allows us to customize our setup to meet our organization's requirements.

Azure Information Protection (AIP)

AIP is a cloud-based solution that helps to discover, classify, and protect sensitive data in a cloud or hybrid environment. AIP uses *labels* to classify data. Once data has been classified, we can protect the data using policies.

Data classification

A famous quote from former US Secretary of State Dean Rusk is "If you protect your paper clips and diamonds with equal vigor, you'll soon have more paper clips and few diamonds." On that particular occasion, he was talking about national security, but when it comes to data protection, the same statement is true. If we need to protect sensitive data, first we need to identify sensitive data. In an infrastructure, this sensitive data can be in different formats and in different locations, such as applications, network shares, and devices. Once sensitive data is identified, we can protect it using relevant services and policies. This is why data classification is so important. AIP uses *labels* to classify data.

My daughter Salena is in Year 4 now. She loves to read. From time to time, I take her to the library to pick up books. When she selects books, she mostly looks at the title of the book and the *label* on it. This *label* is nothing but a color code. This color code references Oxford reading levels. Each level has its own color:

Age 4–5	Age 5–6	Age 6 – 7
Reading at school School Year: Reception Band: Lilac | Level: 1 Band: Pink | Level: 1+ Band: Red | Level: 2 Band: Yellow | Level: 3	**Reading at school** School Year: 1 Band: Blue | Level: 4 Band: Green | Level: 5 Band: Orange| Level: 6	**Reading at school** School Year: 2 Band: Turquoise | Level: 7 Band: Purple | Level: 8 Band: Gold | Level: 9 Band: White| Level: 10 Band: Lime | Level: 11
Reading at home *Read with Oxford*: Stage 2 eBook library: eBooks for 4–5 year olds	**Reading at home** *Read with Oxford*: Stage 3 & Stage 4 eBook library: eBooks for 5–6 year olds	**Reading at home** *Read with Oxford*: Stage 4, Stage 5 & Stage 6 eBook library: eBooks for 6–7 year olds

Figure 15.17: Oxford reading levels

She knows her level and the color code for it. So, without going through the content, she can easily find books that match her reading level. In addition, the children's section of the library is a bit messy, since kids do not return books to the correct shelves. However, even when these books are on the wrong shelves, thanks to the labels, she can still recognize the right books. Data classifications and labels in AIP work in a similar way. In AIP, we can create labels and use them to categorize/group data. Once labels are created, we can apply them to data using manual or automatic methods. In manual methods, we have to go to the file and label it:

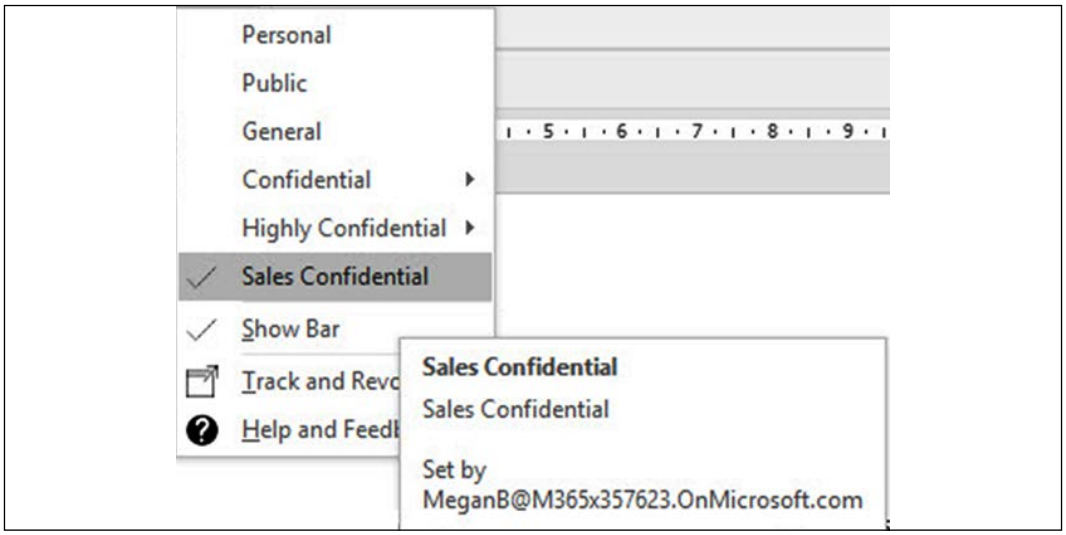


Figure 15.18: Example of labels

When it comes to the automatic method, AIP allows us to perform classification based on custom conditions or predefined information types, such as the following:

- Canadian bank account number
- Canadian driver's license number
- Canadian health service number
- Canadian passport number
- Canadian **Personal Health Identification Number** (**PHIN**)
- Canadian social insurance number
- Chilean identity card number
- Chinese resident identity card number
- Credit card number
- EU debit card number
- EU driver's license number

- EU national identification number
- EU passport number
- EU social security number or equivalent ID
- EU tax identification number
- UK driver's license number
- UK electoral roll number
- UK National Health Service number
- UK **National Insurance Number** (**NINO**)
- US/UK passport number
- US bank account number
- US driver's license number
- US **Individual Taxpayer Identification Number** (**ITIN**)
- **US Social Security Number** (**SSN**)

The following example shows how the automated classification rule labels documents based on their content:

Figure 15.19: Automated classification

AIP supports four methods that can be used to identify, classify, and protect sensitive data:

1. **Office applications**

   Azure RMS, which provides the data protection for AIP, supports the following applications:

   - Microsoft 365 apps (Word, Excel, email, PowerPoint, Visio)
   - Office 2019 (Word, Excel, email, PowerPoint, Visio)
   - Office 2016 (Word, Excel, email, PowerPoint, Visio)
   - Office 2013 (Word, Excel, email, PowerPoint)
   - Office 2010 (Word, Excel, email, PowerPoint)

   By using these applications, we can label, classify, and protect data without the help of plugins.

2. **AIP unified labeling client**

   This client extends labeling, classification, and protection capabilities to File Explorer and PowerShell. Once the client is installed, we can go to File Explorer, select the file, and then use the **Classify and protect** option to label and protect sensitive data. This client can be downloaded using `https://bit.ly/3DNa5It`.

3. **AIP unified labeling scanner**

   In a hybrid cloud environment, sensitive data is also saved in on-prem servers. The AIP scanner allows us to scan the networks to identify sensitive data and apply labels according to organizations' classification policies:

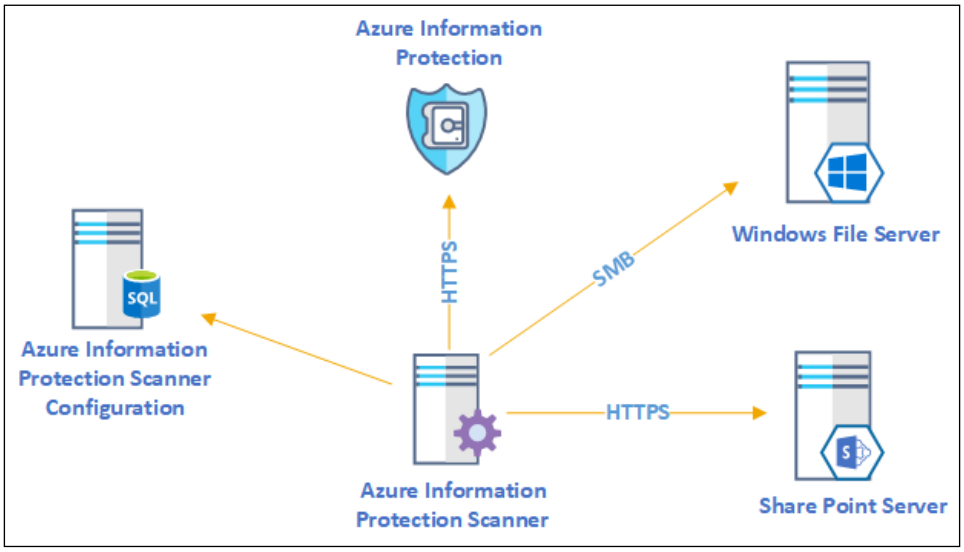


Figure 15.20: AIP scanner

The AIP scanner comes as part of the AIP client. The AIP scanner uses an SQL database to store the scanner configuration. The AIP scanner can automatically discover and classify the data in a selected data repository or run it in discovery mode and report back the findings. Data classification via the AIP scanner is not performed in real time, so you need to decide how often you want to run the scanner.

I have written a step-by-step guide that explains the implementation steps of the AIP scanner. You can access it using `https://bit.ly/30WMt5U`.

4. **The Microsoft Information Protection (MIP) SDK**

   The MIP SDK extends labeling, classification, and protection capabilities to third-party applications. Developers can use the SDK to develop/update applications to natively support data classification and protection. The MIP SDK also supports different platforms such as Ubuntu, Debian, Red Hat Enterprise, macOS, Android, and iOS.

I have already written a couple of blog posts about classification. You can access them by using the following links:

- *Step-by-Step Guide: Protect confidential data using AIP*: `https://bit.ly/3DN722F`
- *Step-by-Step Guide: automatic data classification via aip*: `https://bit.ly/3xk5vi9`

With AIP, we see the term *Azure RMS* quite often. What is the connection between these two services?

Azure Rights Management Services (Azure RMS)

AIP uses Azure RMS as its protection technology. Azure RMS is a cloud-based service that uses identity, encryption, and authorization policies to protect sensitive files and emails across different types of devices (such as PCs, laptops, tablets, and mobile). It can protect data even after it leaves the company premises. It sounds similar to AD RMS. Can Azure RMS replace AD RMS? There are certain similarities as well as differences between these two products. Let's go ahead and try to compare the two products:

Area	**AIP (Azure RMS protection technology)**	**AD RMS**
Infrastructure	AIP is a cloud-native managed service and it works based on subscriptions. We do not have to deploy additional servers to use the services.	The AD RMS role can't work on its own. It is dependent on many different server roles, such as AD FS, IIS, AD CS, and AD DS. Therefore, implementation requires planning, skills, and resources.

Authentication	AIP uses Azure AD for authentication and users can authenticate from anywhere. They do not need to be in the internal network to authenticate. This also makes it easier to share sensitive data with other organizations as most organizations have Azure AD integration.	Prefers internal authentication, and if we need to share content, we need to configure trusts with other organizations.
Classification and labeling	AIP uses labels to classify sensitive data and protect those based on policies. AIP supports both automatic and manual classification. AIP can label and classify data with Office applications, File Explorer, PowerShell, and on-prem data stores.	AD RMS does not support data classification or labeling.
Device support	AIP supports mobile devices (Android, iOS) and Mac computers.	To support mobile devices, AD RMS requires AD FS federation, mobile device extension, and public DNS records to be in place.
Templates	AIP creates default templates that can be used to protect sensitive data in organizations immediately.	AD RMS does not have any default templates.
Document tracking and revocation	AIP supports document tracking and revocation.	AD RMS does not support these features.
Information rights management (IRM)	Supports on-prem server products and Microsoft online services.	Supports on-prem server products and Microsoft online services.
Multi-Factor Authentication (MFA)	Supports MFA.	Supports smart card authentication only if IIS is configured to request certificates.
Cryptographic mode	Supports cryptographic mode 2 by default.	Supports cryptographic mode 1 by default. Requires additional configuration to support mode 2.

Based on the above comparison, we can see that AIP is more feature-rich compared to AD RMS. It is the more appropriate solution for modern data protection requirements.

How does Azure RMS work?

So far, we have talked about AIP and Azure RMS's capabilities. But how do they work? What is the technology behind them? In this section, we are going to look into this in detail.

At a high level, I can explain the Azure RMS document protection process as follows:

- When a user protects a document, Azure RMS encrypts the content of the file and attaches an access policy to it. This policy decides what other users can do with the protected data.
- When other users access the file (after successful Azure AD authentication), Azure RMS decrypts the file and applies an access policy to it.

It sounds simple, but let's go through this process in detail so we know the technology behind it.

The best way to understand the technology behind the encryption and decryption process is to go through a scenario. **Andrew**, a Rebeladmin Inc. employee, is sending a document with sensitive data to another employee, **Selena**. He does not want anyone else in the sales team to have it, so he is going to use AIP to protect the document:

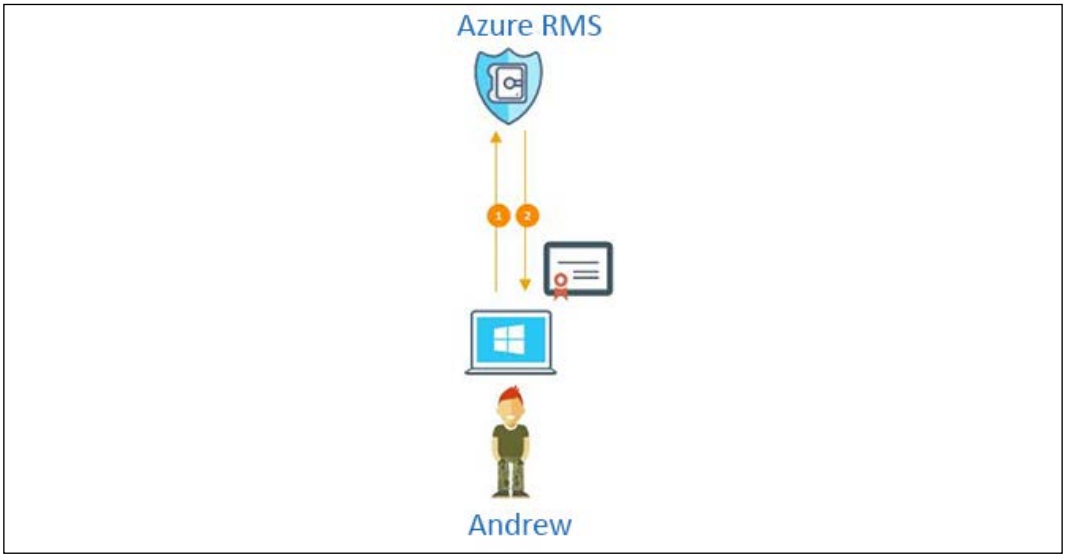


Figure 15.21: Initial communication with Azure RMS

Since this is the first time that **Andrew** and **Selena** are using AIP, they both need to go through the one-time user environment preparation process:

1. First, **Andrew** installs the AIP client on his PC. This can be downloaded via `https://bit.ly/3DQWZKi`.
2. Then, he authenticates into AIP using his Azure AD account. After successful authentication, the session is redirected to the AIP tenant. Then, it issues a certificate that **Andrew** will use to authenticate into **Azure RMS** in the future. This certificate will be automatically renewed by the AIP client after 31 days. A copy of this certificate is also stored in Azure. If the user changes the device, then **Azure RMS** recreates the certificate using the same keys:

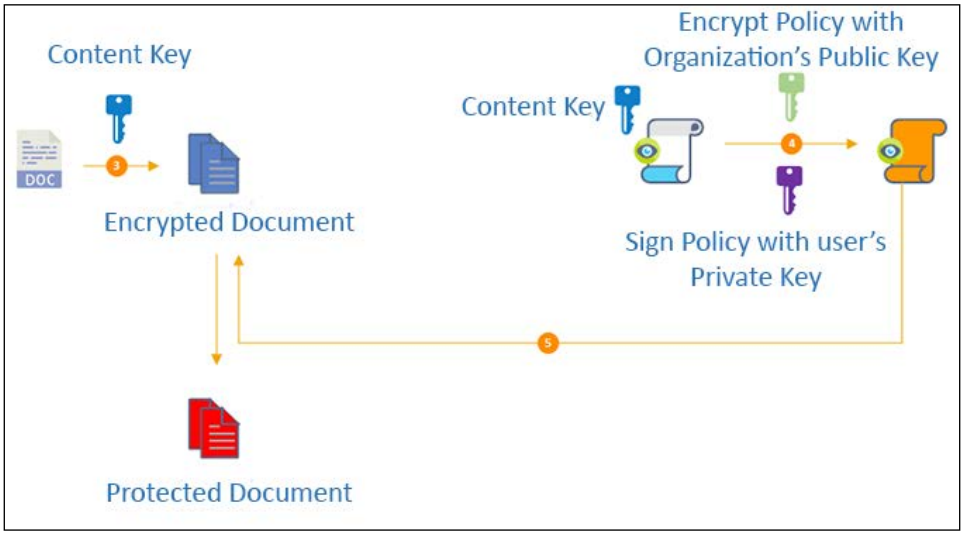


Figure 15.22: Preparing for document protection

Now, the user environment preparation process is done. The next step is to protect the Word document.

3. **Andrew** goes ahead and requests for the AIP client to protect the document. The AIP client creates a random AES key and encrypts the document content using it. This key is called the **Content Key** and it uses the AES symmetric encryption algorithm. The key length is 128 bits or 256 bits.
4. Then, the AIP client creates a policy that contains the access rights for the recipient **Selena**. This can be done using a policy template that has already been created by an administrator; otherwise, the user can create an ad hoc policy. Once the policy is in place, the system will encrypt the policy and the symmetric content key by using the organization's public key. This key was retrieved by the AIP client during the initial user environment preparation process. Then, the policy and the content key are signed by **Andrew**'s certificate, which was obtained during the same preparation process.

5. In this next step, the AIP client creates a **Protected Document**, which includes the **Encrypted Document** and the policy that has already been encrypted and signed.

   Once the system has created the **Protected Document**, **Andrew** sends it to **Selena** via email:

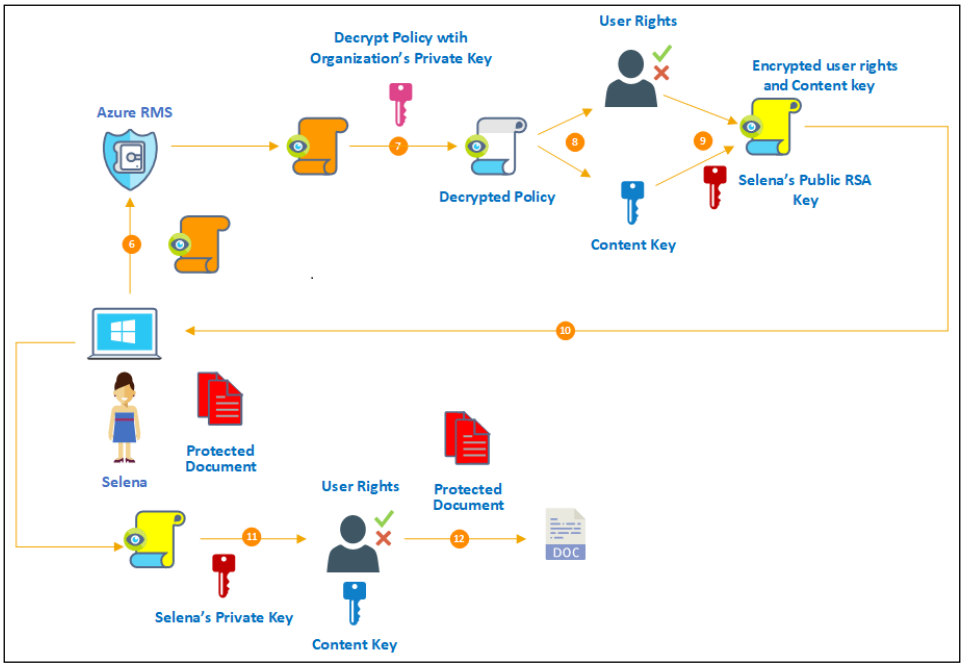


Figure 15.23: Data protection with Azure RMS

   Once **Selena** receives the **Protected Document**, she tries to open it. As this is the first time she has tried to open the **Protected Document**, she needs to complete the user environment preparation process before she can start using AIP.

6. After **Selena** authenticates successfully, the system retrieves the policy and **Andrew**'s certificate from the **Protected Document** and then forwards it to **Azure RMS**.
7. The service decrypts the policy using the **Organization's Public Key**.
8. The **Decrypted Policy** contains **Selena**'s **User Rights** and **Content Key**. The system evaluates the permissions in order to understand the rights associated with the document.
9. In this step, the **Content Key** re-encrypts using **Selena**'s public RSA key. Then, it is attached to the **User Rights**.

10. In this step, the files created in *steps 8* and *9* are delivered to **Selena**'s computer.
11. The AIP client decrypts the **User Key** and the **Content Key** using **Selena's Private Key**, which was retrieved via the initial user environment preparation process. This process reveals the **User Rights** list and the **Content Key**.
12. With the help of the **Content Key**, the AIP client decrypts the **Encrypted Document**. The AIP client also passes the rights list to the application and the application decides what **Selena** can do with the document.

This completes the decryption process and, in the end, **Selena** was able to open the **Protected Document**. The previous scenario shows us exactly what is happening behind the scenes when we protect and consume sensitive data using **Azure RMS**.

AIP implementation

AIP implementation is a vast topic that is beyond the scope of this chapter. I have written a series of blog posts covering the implementation of AIP:

- *Step-by-Step Guide: Protect confidential data using Azure Information Protection*: `https://bit.ly/30NxJG1`
- *Step-by-Step Guide: Automatic Data Classification via Azure Information Protection*: `https://bit.ly/30ZZGe5`
- *Step-by-Step Guide: On-premise Data Protection via Azure Information Protection Scanner*: `https://bit.ly/3nKfiei`
- *Step-by-Step Guide: How to protect confidential emails using Azure Information Protection?*: `https://bit.ly/3l4LU0x`
- *Step-by-Step Guide: How to track shared documents using Azure Information Protection?*: `https://bit.ly/3FYNGZt`

Summary

Data protection is crucial in modern infrastructures, as more and more analog data is being transformed into digital data. There are different laws, products, technologies, and methodologies to improve data protection in infrastructures.

AD RMS is Microsoft's solution that can be used to manage the operational behavior of confidential data in an infrastructure.

In this chapter, we learned about AD RMS and its related characteristics. Then, we moved on to understanding how AD RMS works, and how it protects data.

After that, we looked at different AD RMS deployment topologies. Later, we worked on AD RMS installation, configuration, and testing.

AIP is a cloud-native service that can be used to protect sensitive data in a cloud or hybrid environment. In this chapter, we learned about the differences between AIP and AD RMS. Last but not least, we also learned how AIP protects sensitive data.

This ends the third part of this book. The fourth part of the book is going to start with a chapter that is focused on AD security best practices.

16

Active Directory Security Best Practices

Hackers are breaking the systems for profit. Before, it was about intellectual curiosity and pursuit of knowledge and thrill, and now hacking is big business.

- Kevin Mitnick

Kevin Mitnick was once known as the world's most wanted hacker. Most of the cyber security laws that exist today were first introduced to the world because of him. For anyone who is interested in cyber security, I highly recommend his book *Ghosts in the Wires*. It not only talks about his life as a hacker, but it also explains the evolution of the "hacker" over time. He started hacking into phone systems first and then, as technology developed, he challenged himself with breaking into computer systems. For him, it wasn't about profit, but today, things are way more complicated. According to `https://vz.to/2Zm81bn`, 90% of data breaches are finically motivated. Hackers are targeting more valuable assets such as intellectual property, state secrets, and identities.

In a computer system, we can no longer identify *good* or *bad* people. Most recent security breaches have involved some sort of support from *inside users*. This is why a *zero-trust* approach is vital when it comes to security. We can no longer say that anything *inside* the perimeter is trustworthy and anything *outside* is a risk. In an infrastructure, **Active Directory** (**AD**) is mainly responsible for managing identities. The protection of identities is crucial for any infrastructure. However, there are so many other things that we need to secure, such as network, storage, and data, in order to protect identities.

If you need to maintain a car properly, then you can't just pay attention to the engine. It is the heart of the car, but if you really need it to be a good car, you need to equally take care of, as well as investing in, every other component of the car.

In this chapter, we are going to discuss many different AD features that can be used to improve the security of the AD infrastructure. This chapter will cover the following topics:

- How does AD authentication work?
- Delegating permissions
- The AD Protected Users security group
- Restricted RDP mode
- Authentication policies and authentication policy silos
- Secure LDAP
- Microsoft LAPS
- Azure AD Password Protection

In an AD environment, we use usernames and passwords to authenticate. But what exactly happens during the authentication process? It is important to know that we are looking to protect user accounts.

AD authentication

AD uses Kerberos version 5 as the authentication protocol between the domain controller and the clients. Kerberos v5 became the default authentication protocol for Windows Server from Windows Server 2003. It is not a Microsoft proprietary protocol; it is an open standard. Therefore, AD can work with any application or service that supports the same standard.

The Kerberos protocol

The Kerberos protocol is built to protect authentication between the server and the client in an open network.

The main concept behind authentication is that two parties first agree on a password (secret) and then use it to identify and verify their genuineness:

Figure 16.1: Authentication by using a secret

In the preceding example, **Dave** and **server A** have a communication link. They often exchange confidential data. To protect this communication, they agree to use a common secret code (**1234**) to verify their identities before exchanging data. When **Dave** makes the initial connection, he passes his secret to **server A** and says **Hey! I'm Dave**. Then, **server A** checks the secret to see whether it's true. If it's correct, it identifies him as **Dave** and allows further communication.

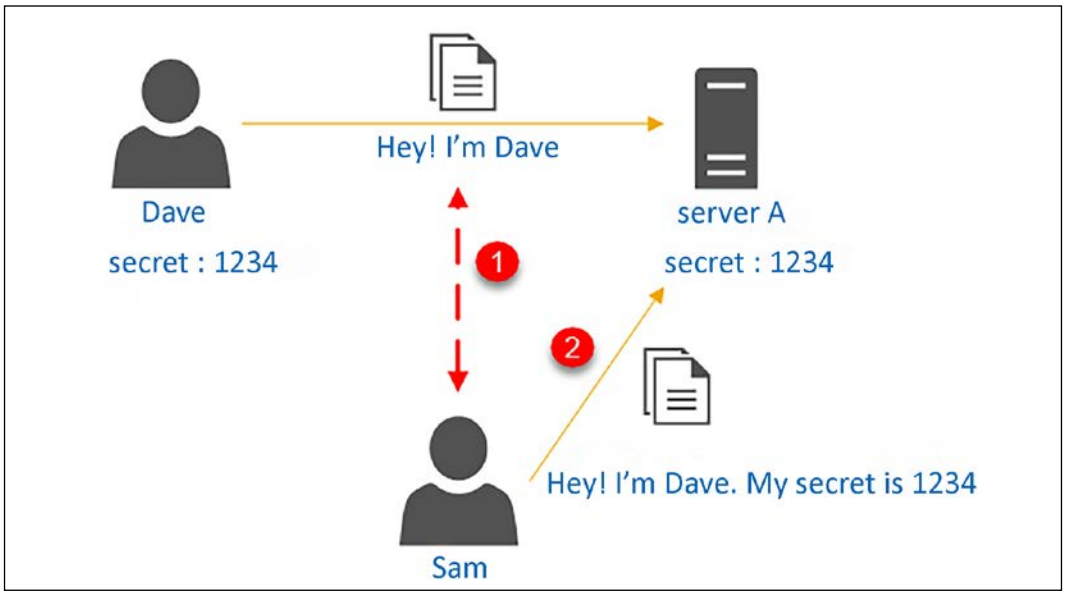


Figure 16.2: Man-in-the-middle attack

Communication between **Dave** and **server A** happens in an open network, which means that other systems and users are also using the same network. **Sam** is another user in the same infrastructure. He is aware of the communication that happens between **Dave** and **server A**. The data that is exchanged between the two parties has a high value and Sam is trying to get his hands on it. He started sniffing in the network to find out the secret they use. Once he has found it, he can communicate with **server A** and pretend to be **Dave** by providing the secret code, **1234**. However, **server A** doesn't see any difference between requests from **Dave** and **Sam**, as both provide the correct secret.

Kerberos solves this security challenge by using a shared symmetric cryptographic key instead of secrets. It also uses this key for encryption and decryption. The name *Kerberos* comes from a powerful three-headed dog in Greek mythology. Just like the mythical three-headed dog, the Kerberos protocol has three main components:

- A client
- A server
- A trusted authority to issue secret keys

This trusted authority is called the **Key Distribution Center** (**KDC**). Before we look into Kerberos in detail, let's first look at how a typical key exchange works:

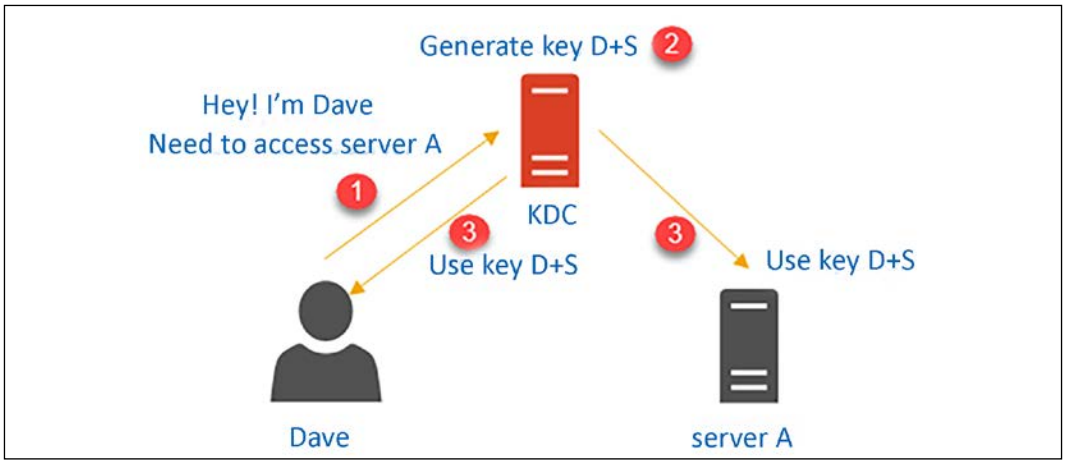


Figure 16.3: KDC in action

If we revisit the previous scenario with a **KDC** in place, instead of communicating with **server A** directly, **Dave** goes to the **KDC** and says he needs to access **server A** (**Step 1**). Dave needs a symmetric key to start communication with **server A**. This key should only be used by **Dave** and **server A**. The **KDC** accepts the request, generates a key (**key D+S**), and then issues it to **Dave** and **server A** (**Step 2**).

By the looks of it, this seems quite straightforward, but from **server A**'s point of view, there are a few challenges.

To accept a connection from **Dave**, **server A** should have a record regarding **key D+S**. The easiest way to do this is to store the key in **server A**. We are considering storing only one connection in **server A**, but if there were a hundred connections, **server A** would need to store all hundred keys. This doesn't seem to be practical. Also, if **server A** were compromised, attackers would have access to all the keys. So, is there a better way of doing this?

In an AD environment, the **KDC** is installed as part of the domain controller. The **KDC** is responsible for two main services: one is the **Authentication Service** (**AS**) and the other is the **Ticket-Granting Service** (**TGS**).

In the preceding example, when **Dave** logs in to the system, it needs to be proved to the KDC that he is the exact person he claims to be. When he logs in, it sends the username to the **KDC** along with a *long-time key*. The long-time key is generated based on **Dave**'s password. The Kerberos client on **Dave**'s PC accepts his password and generates the cryptographic key. The **KDC** also maintains a copy of this key in its database. Once the **KDC** receives the request, it checks the user's username and the long-term key with its records. If it's all good, the **KDC** responds to **Dave** with a *session key*. This is called a **Ticket-Granting Ticket** (**TGT**).

The TGT contains two things:

- A copy of a session key that the **KDC** uses to communicate with **Dave**. This is encrypted with the **KDC**'s long-term key.
- A copy of a session key that **Dave** can use to communicate with the KDC. This is encrypted with **Dave**'s long-term key, so only **Dave** can decrypt it.

Once **Dave** receives this key, he can use his long-term key to decrypt the session key. After that, all future communication with the **KDC** will be based on this session key. This session key is temporary and has a **Time-to-Live** (**TTL**) value.

This session key is saved in **Dave**'s computer's volatile memory. The next step in the process is to request access to **server A**. To do this, **Dave** has to contact the **KDC** again, but this time he uses the session key provided by the **KDC**. This request includes the TGT and the timestamp encrypted by the session key and the service ID (the service that is running on **server A**). Once the **KDC** receives it, it uses its long-term key to decrypt the TGT and retrieve the session key. Then, using the session key, it decrypts the timestamp. If the time difference is less than 5 minutes, then it proves that it came from **Dave**.

Once the **KDC** confirms the server access request is a legitimate request, it creates another ticket, and this is called a **service ticket**. It contains two keys: one for **Dave** and one for **server A** (**Step 3**). Both keys include the requester's name (**Dave**), the recipient, the timestamp, the TTL value, and a new session key (which will be shared between **Dave** and **server A**). One of these keys is encrypted using **Dave**'s long-term key. The other key is encrypted using **server A**'s long-term key. In the end, both are encrypted together using the session key between the **KDC** and **Dave**. Finally, the ticket is ready and sent over to **Dave**. **Dave** decrypts the ticket using the session key. He also finds *his* key and decrypts it using his long-term key.

This process reveals the new session key that needs to be shared between **Dave** and **server A**. Then, **Dave** creates a new request, which includes **server A**'s key that was retrieved from the service ticket, and the timestamp that was encrypted using the new session key created by the **KDC**. Once **Dave** sends the key over to **server A**, it decrypts its key using its long-term key and retrieves the session key. Using the session key, it can decrypt the timestamp to verify the authenticity of the request. As we can see, in this process, it is not **server A**'s responsibility to keep track of the key used by the client and it's not the client's responsibility to keep the relevant keys.

Authentication in an AD environment

Let's go ahead and revisit our previous example, to see how the authentication process works in an AD environment:

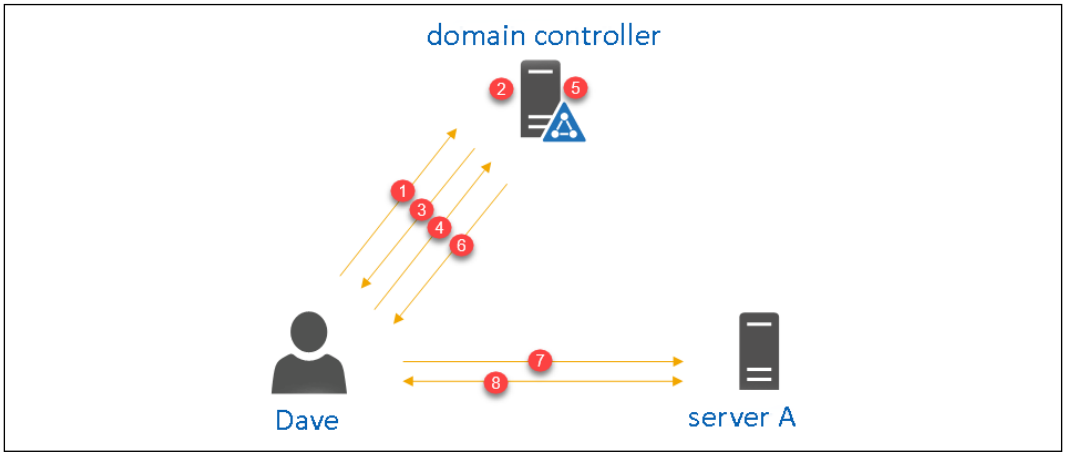


Figure 16.4: Authentication process in AD

The following list summarizes the steps involved in the authentication process:

1. **Dave** sends the username and his long-term key to the KDC (**domain controller**).

2. The KDC checks the username and long-term key with its database and verifies the identity. Then, it generates a TGT. It includes a copy of a session key, which the KDC uses to communicate with **Dave**. This is encrypted using the KDC's long-term key. It also includes a copy of a session key that **Dave** can use to communicate with the KDC.
3. The KDC responds to **Dave** with its TGT.
4. **Dave** decrypts his key using his long-term key and the retrieved session key. His system creates a new request, which includes the TGT and the timestamp encrypted by the session key and service ID. Once the request is generated, it is sent to the KDC.
5. The KDC uses its long-term key to decrypt the TGT and retrieve the session key. Then, the session key can be used to decrypt the timestamp. Next, it creates a service ticket. This ticket includes two keys: one for **server A** and one for **Dave**. **Dave**'s key is encrypted using his long-term key and **server A**'s key is encrypted using its long-term key. In the end, both are encrypted using the session key that is used by the KDC and **Dave**.
6. The KDC sends the service ticket to **Dave**.
7. **Dave** decrypts the ticket using the session key and retrieves his key. In the end, the system creates another request, including **server A**'s key (which was created earlier) and the timestamp that is encrypted by the session key that **Dave** decrypted earlier in this step. Once everything is ready, the system sends the request to **server A.**
8. **server A** goes ahead and decrypts **Dave**'s key using its long-term key and the retrieved session key. Then, using it, **server A** can decrypt the timestamp to verify the request's authenticity. Once everything is green, a connection between **Dave** and **server A** is established.

There are a few other things that need to be fulfilled to complete this process:

- **Connectivity**: The server, client, and KDC need to have a reliable connection between them to process requests and responses.
- **DNS**: Clients use DNS to locate the KDC and servers. Therefore, a functioning DNS with the correct records is required.
- **Time synchronization**: As we can see, the process uses the timestamp to verify the authenticity of the requests. It allows up to 5 minutes time difference. Therefore, it's a must to have accurate time synchronization with the domain controllers.
- **Service Principal Names** (**SPNs**): Clients use SPNs to locate services in the AD environment. If there is no SPN for the services, the clients and the KDC cannot locate them when required. When setting up services, make sure that you set up SPNs as well.

In this section, we have learned about what the Kerberos protocol is and how it works with AD authentication. Kerberos has built-in security features to protect user identities during the authentication process, but this is not enough to protect identities from emerging threats. Therefore, in the next section, we are going to look into the different methods, tools, and features that we can use to improve the security of our AD environment further.

Delegating permissions

The Kerberos protocol itself was built to prevent identity compromise. This is all good on paper, but in reality, attackers use many different methods and tools to attack AD environments. Therefore, it is important to know the features, techniques, and tools that we can use to protect AD environments further.

In an AD environment, there are different types of management tasks. Managing domain controllers, adding, modifying, and removing users, adding, managing, and removing groups, resetting passwords, and adding devices to a domain are just some examples. In a structured IT department, these management tasks can be assigned to different job roles.

As an example, let's assume that Rebeladmin Corp.'s IT department has first-line (first support contact), second-line (intermediate), and third-line (senior) IT teams. When considering the AD management tasks, first-line engineers are usually involved with tasks such as user password resets, setting up new user accounts and groups, and adding devices to domains.

Second-line engineers are involved with additional tasks such as Group Policy setup and Group Policy troubleshooting. Third-line engineers usually work on tasks such as advanced troubleshooting, domain controller installations, schema changes, and physical and logical design changes.

In this way, we can group AD management tasks according to the responsibilities of different engineers' roles. This allows different job roles to take *ownership* of AD management tasks. At the same time, if a certain job role is assigned the ownership of a task, there should be a mechanism to prevent other teams from interfering with that particular task. As an example, if third-line engineers are responsible for AD schema changes, then there should be a way to prevent first-line and second-line engineers from doing them. If we need to prevent users or groups from accessing a folder in a file server, or if we need to grant them access, we can do so using *permissions*. In the same way, AD also allows you to manage users' and groups' authority over objects or management tasks based on permissions.

Managing permissions for the IT team is a difficult task, as it is not just about permission. It has a social aspect too. In general, we accept that administrators are trustworthy people; most of them are but we never know. Therefore, it is best to take precautions and manage permissions sensibly. There are a few ways to manage permissions for AD management tasks:

- By using predefined AD administrator roles
- By using object **Access Control Lists** (**ACLs**)
- By using the delegate control method in AD

Let's go ahead and explore further the methods mentioned above.

Predefined AD administrator roles

AD has predefined administrator roles. Each of these roles has predefined permissions attached to them. If a user needs these role permissions, their account needs to be added to the relevant security group. These security groups are predefined groups:

- **Enterprise Admins**: This is the highest AD role permission that can be applied in the AD forest. The accounts that are part of this group can modify the logical and physical topology of the AD infrastructure. This also allows you to perform schema changes. This role is capable of managing other role memberships (Enterprise Admins, Schema Admins, and Domain Admins).
- **Schema Admins**: Members of this group can modify the AD schema. This is only included in the forest root domain as the schema is handled on the forest level.
- **Domain Admins**: This is the highest AD role permission that can be applied in an AD domain. When adding the first domain controller to the forest, the default administrator account will be part of the Domain Admins and Enterprise Admins groups. Domain admins have permission to add/remove other user accounts from the Domain Admins security group.

These roles are considered to be privileged roles in the AD environment. Therefore, rather than keeping permanent memberships, it's recommended that you use **Privileged Access Management** (**PAM**) to provide time-based group memberships. This was described in detail in *Chapter 2, Active Directory Domain Services 2022*.

Using object ACLs

User or group access permissions to a shared folder are managed by the ACL. Similarly, we can define permissions to AD objects by using ACLs. This can be applied to individual objects or to the AD site/domain/**Organizational Unit** (**OU**), and then the same permissions can be forced onto lower-level objects.

As an example, I have a security group called `First Line Engineers`, and Liam is a member of this group. Liam is an engineer in the Europe office. In the AD environment, Liam should be allowed to add user objects under any sub-OU that is under the `Europe` OU. However, he should not be allowed to delete any objects that are under it. Let's see how we can do this using ACLs:

1. Log in to the domain controller as a domain admin/enterprise admin.
2. Review the group membership using the following command:

```
Get-ADGroupMember "First Line Engineers"
```

3. Go to **Active Directory Users and Computers** (**ADUC**), right-click on the `Europe` OU, and click on **Properties**. Then, go to **Security**.
4. In the **Security** tab, click on **Add**.
5. In the new window, type `First Line Engineers` and click on **OK**. Afterward, in the **Security** tab, select **First Line Engineers** and click on **Advanced**:

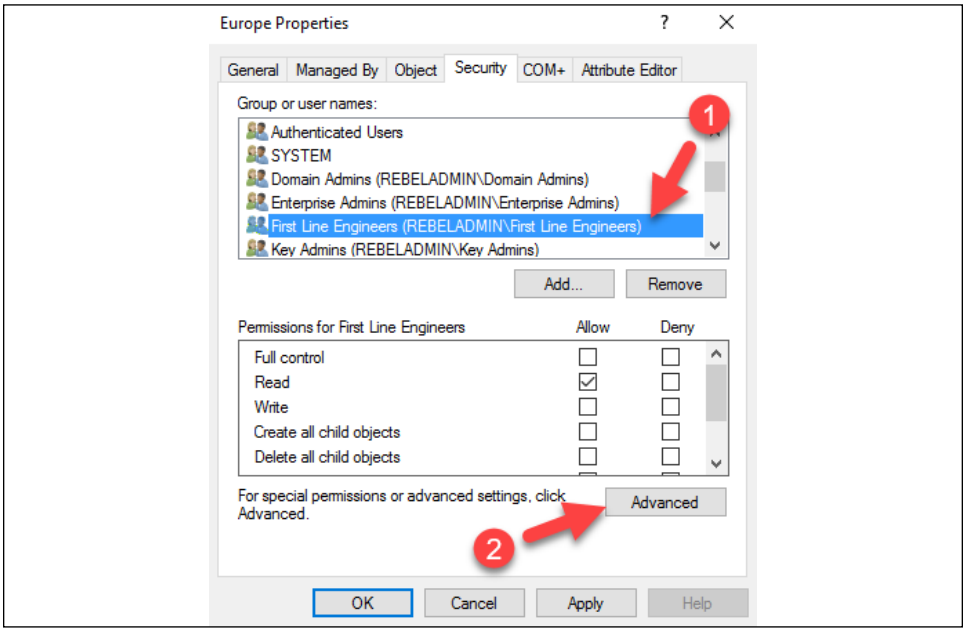


Figure 16.5: Open advanced permission settings for a group

6. In the next window that appears, select **First Line Engineers** from the list and click on **Edit**.
7. From the **Applies to** list, select **This object and all descendant objects** to apply the permission to all child objects:

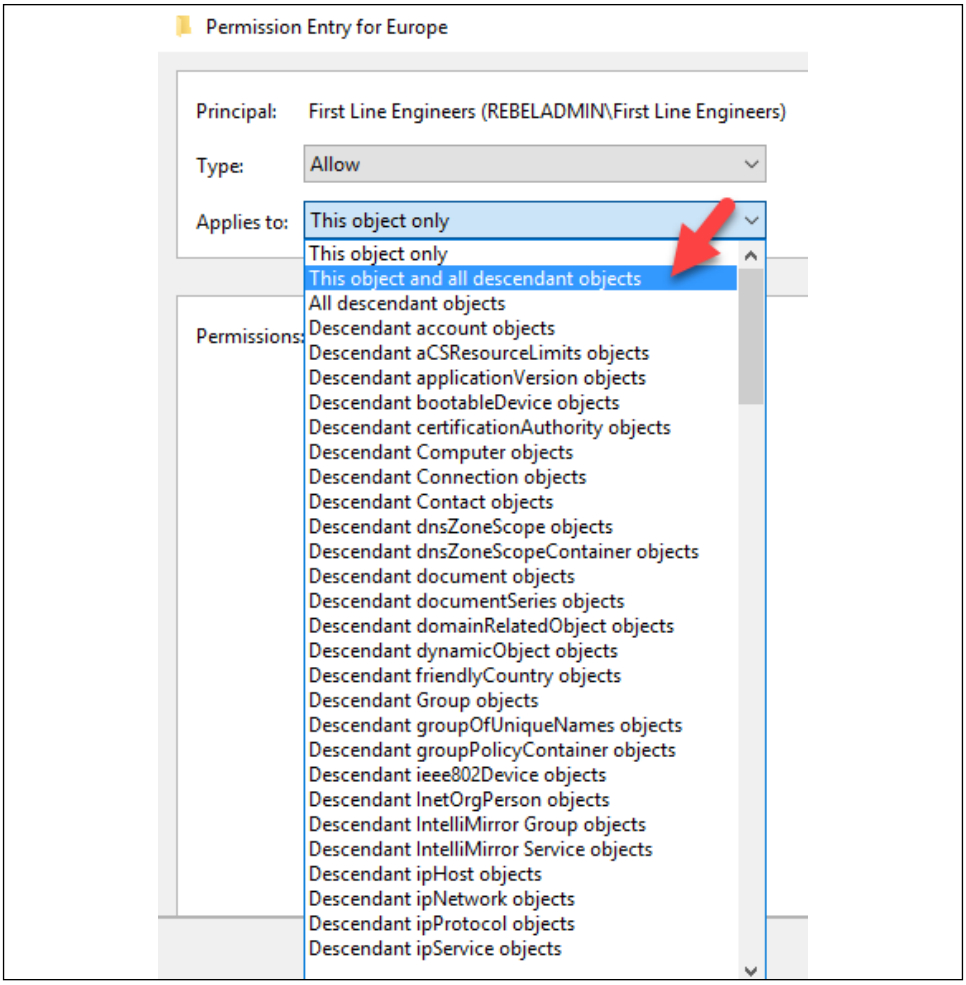


Figure 16.6: Advanced permission settings for a group

8. Under the **Permissions** section, check **Create all child objects** and click **OK**.
9. Then, keep clicking on **OK** until all permission windows are closed.
10. Then, I log in to a Windows 10 computer as user `Adam`. This computer already has RSAT tools (`https://bit.ly/3CGBCda`) installed.

11. According to the permissions, we should be able to add the user account under the `Europe` OU:

```
New-ADUser -Name "Dale"
-Path "OU=Users,OU=Europe,DC=rebeladmin,DC=com"
```

12. This successfully adds the user. Let's see whether we can add another user on a different OU:

```
New-ADUser -Name "Simon"
-Path "OU=Users,OU=Asia,DC=rebeladmin,DC=com"
```

13. As soon as I run the preceding code, I get an `Access is denied` error:

Figure 16.7: ACL permissions are preventing the provision of a new user account

14. According to the applied permissions, Liam should not be able to delete any object under `OU=Users,OU=Europe,DC=rebeladmin,DC=com` either. Let's check this using the following command:

```
Remove-ADUser -Identity "CN=Dishan Francis,
OU=Users,OU=Europe,DC=rebeladmin,DC=com"
```

15. As soon as I run the preceding command, I get an `Access is denied` error:

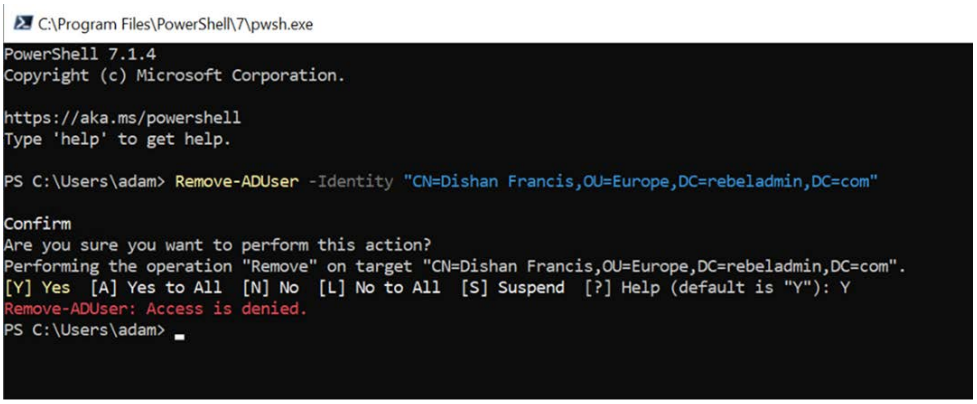


Figure 16.8: ACL permissions are preventing the provision of a new user account

This confirms that we can manage permissions for AD management tasks using ACLs.

Using the delegate control method in AD

The delegate control method also works similarly to ACLs, but it simplifies privilege management as it uses the following:

- The **Delegation of Control Wizard** can be used to apply delegated permissions
- A predefined task list is available via the wizard and we can easily map permissions to these tasks

This wizard contains the following predefined tasks, which can be used to assign permissions:

- Create, delete, and manage user accounts
- Reset user passwords and force a password to change at the next logon
- Read all user information
- Create, delete, and manage groups
- Modify the membership of a group
- Manage Group Policy links
- Generate **Resultant Set of Policy (Planning)**
- Generate **Resultant Set of Policy (Logging)**
- Create, delete, and manage **inetOrgPerson** accounts
- Reset **inetOrgPerson** passwords and force a password change at the next logon
- Read all the **inetOrgPerson** information

These also allow you to create a custom task to delegate permissions if it's not covered in the common tasks list.

Similar to ACLs, permissions can be applied at the following levels:

- **Site**: Delegated permissions will be valid for all the objects under the given AD site
- **Domain**: Delegated permissions will be valid for all the objects under the given AD domain
- **OU**: Delegated permissions will be valid for all the objects under the given AD OU

As an example, I have a security group called `Second Line Engineers`, and Scott is a member of it. I need to allow members of this group to reset passwords for objects in `OU=Users,OU=Europe,DC=rebeladmin,DC`.

To do that, we need to do the following:

1. Log in to the domain controller as a domain admin/enterprise admin.
2. Review the group membership using the following command:

```
Get-ADGroupMember "Second Line Engineers"
```

3. Go to ADUC, right-click on the `Europe` OU, and then from the list, click on the **Delegate Control...** option.
4. This opens a new wizard on the initial page; click on **Next** to proceed.
5. On the next page, click on the **Add…** button and add the `Second Line Engineers` group to it. Then, click on **Next** to proceed:

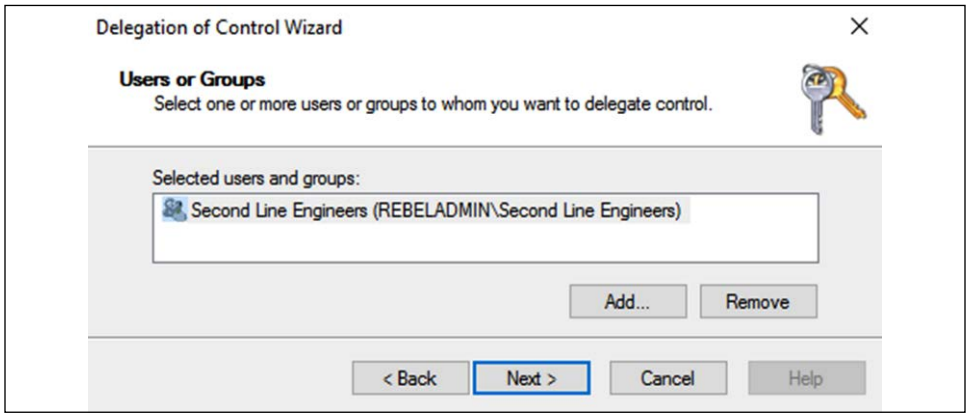


Figure 16.9: Delegating control for a group

6. From the **Tasks to Delegate** window, select the **Delegate the following common tasks** option, and from the list, select **Reset user passwords and force password change at next logon**. On this page, we can select multiple tasks. If none of those work, we can still select **Create a custom task to delegate**. Once you have completed the selection, click on **Next** to proceed:

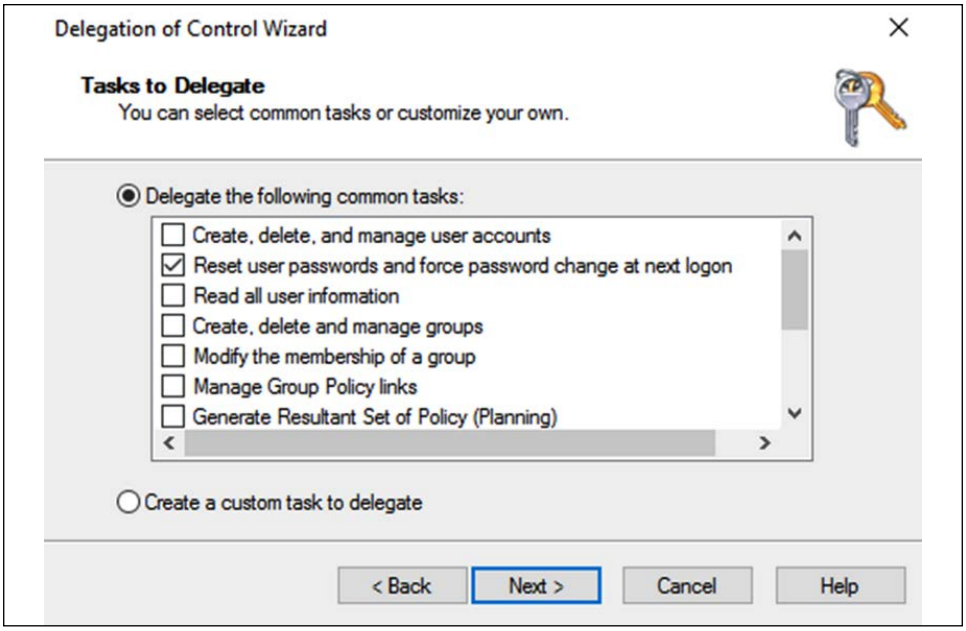


Figure 16.10: Delegate tasks

7. This completes the wizard. Click on **Finish** to complete.
8. Now, it's time for testing. I log in to a Windows 10 computer, as user Adam, who has RSAT tools (`https://bit.ly/3nKiZ3B`) installed.
9. According to the permissions, Adam should be able to reset the password of an object under `OU=Users,OU=Europe,DC=rebeladmin,DC`:

```
Set-ADAccountPassword -Identity difrancis
```

10. This allows you to change the password successfully.
11. However, it should not allow Adam to delete any objects. We can test it using the following command:

```
Remove-ADUser -Identity "CN=Dishan Francis,
OU=Users,OU=Europe,DC=rebeladmin,DC=com"
```

12. As expected, it returns an `Access is denied` error:

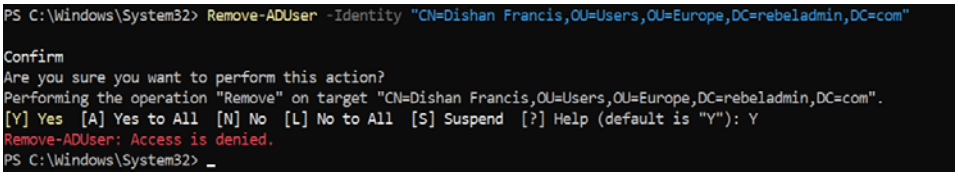


Figure 16.11: Can't remove user as no delegated permissions

As we can see, the Delegation of Control Wizard configures the correct permissions as expected.

Implementing fine-grained password policies

Complex passwords are a basic security setting that any administrator uses. In the AD environment, password complexity settings and account lockout settings can be configured by using GPO settings, which are located at **Computer Configuration | Policies | Windows Settings | Security Settings | Account Policies**. Before Windows Server 2008, there was only one password policy and account lockout policy setting that could be applied to users. With Windows Server 2008, Microsoft introduced fine-grained password policies, which allow administrators to create different password and account lockout policy settings for individual users or groups. This allows you to protect privileged accounts using stronger policies than regular user accounts. This feature continued with every AD DS version after 2008 and is available with AD DS 2022 as well.

Once, I was working on an AD audit for a hedge fund. As part of the report, I recommended that they use password policies with greater complexity, as they were not doing so. After I explained things, the IT manager there agreed and I configured the password policy and the account lockout policy. After a few days, I went to the same site, and the IT manager showed me that after forcing the policy, the end users started to write down their complex passwords on sticky notes and papers. So, even though it was a security setting, in the end, it led to bigger security issues as the users could see each other's passwords on sticky notes. However, by using fine-grained password policies, administrators can apply different settings based on the situation. As an example, while using 5-character complex passwords for sales department users, you can use 12-character complex passwords for domain admins.

Limitations

Fine-grained password policies have the following limitations:

- Fine-grained password policies can only be applied to users and global security groups. They can't be applied to OUs.
- By default, only domain admins/enterprise admins can set up/manage/delete fine-grained password policies. It is possible to delegate permission to other users if required.
- The minimum domain functional level is Windows Server 2008.

When you use fine-grained password policies, some objects may have multiple fine-grained password policies applied. However, only one password policy can be applied to an object at a given time. It is not possible to merge multiple policies either. So how do we know what is the winning policy? Or how can we enforce a policy? Let's find out the answers in the next section.

Resultant Set of Policy (RSoP)

RSoP uses the attribute value of `msDS-PasswordSettingsPrecedence`, which is associated with each password policy, to decide the winning policy. A precedence value is an integer value an administrator can define. A lower precedence value means higher priority. If multiple password policies are applied to the same object, the password policy with the lower precedence value wins.

The following list further explains how password policies work in an infrastructure:

- There are two ways to link an object to a password policy. The first method is via a directly linked policy. The second method is via group membership. If the policy targets a security group, its members will automatically link to the password policy. However, if a fine-grained password policy is linked to an object, it will be the winning policy.
- If there's no directly linked policy, the object will consider the lowest policy precedence. These policies are inherited from the security groups that they belong to.
- If neither of the settings is applicable, the default GPO password policy setting is applied.

Let's go ahead and look into the configuration of a fine-grained password policy.

Configuration

There are two ways to apply fine-grained password policies. The first option is to use **Active Directory Administrative Center** (**ADAC**), and the other options are to use PowerShell cmdlets or ADSI Edit.

In ADAC, go to **System | Password Settings Container**. Then, right-click and go to **New | Password Settings**. This will open up a window where we can define the policy settings:

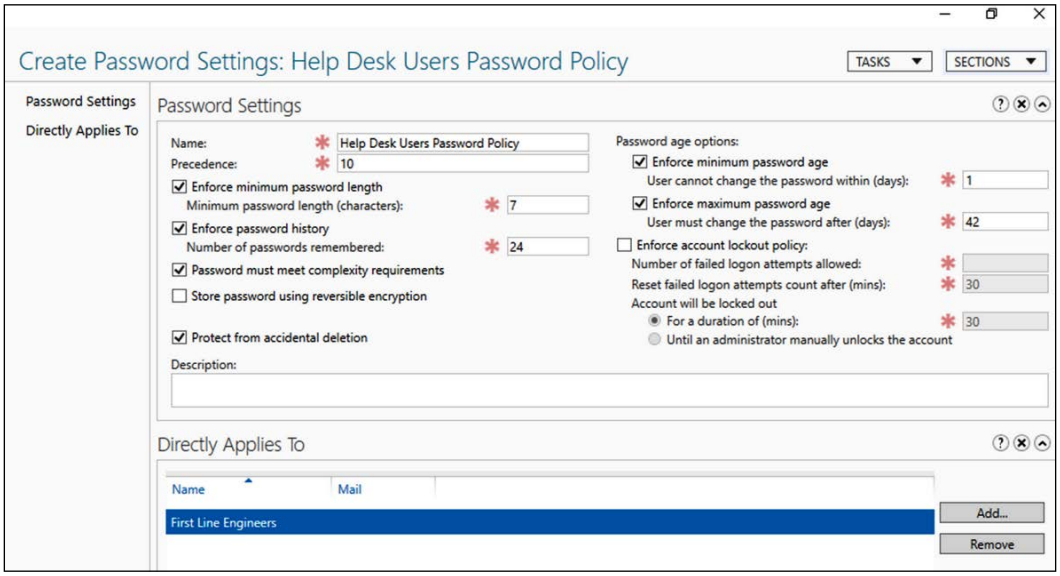


Figure 16.12: Password policy settings

In the policy window, we can define the policy name, precedence, and account lockout policy settings. **Directly Applies To** is the place where we can define the users or security groups this new policy should target. In the preceding example, I added the `First Line Engineers` security group as the target.

Fine-grained password policies can also be created using PowerShell:

```
New-ADFineGrainedPasswordPolicy -Name "Domain Admin Password Policy"
-Precedence 1 `
-MinPasswordLength 12 -MaxPasswordAge "30" -MinPasswordAge "7" `
-PasswordHistoryCount 50 -ComplexityEnabled:$true `
-LockoutDuration "8:00" `
-LockoutObservationWindow "8:00" -LockoutThreshold 3 `
-ReversibleEncryptionEnabled:$false
```

In the preceding command, `New-ADFineGrainedPasswordPolicy` is the cmdlet that is used to create a new policy. `-Precedence` defines the policy precedence. The `-LockoutDuration` and `-LockoutObservationWindow` values are defined in hours. The `-LockoutThreshold` value defines the number of login attempts allowed.

The policy settings can be viewed using ADAC or PowerShell:

```
Get-ADFineGrainedPasswordPolicy –Identity "Domain Admin Password
Policy"
```

The preceding command retrieves the following settings for the given password policy:

```
PowerShell 7.1.3
Copyright (c) Microsoft Corporation.

https://aka.ms/powershell
Type 'help' to get help.

PS C:\Windows\System32> Get-ADFineGrainedPasswordPolicy -Identity "Domain Admin Password Policy"

AppliesTo                   : {}
ComplexityEnabled           : True
DistinguishedName           : CN=Domain Admin Password Policy,CN=Password Settings
                              Container,CN=System,DC=rebeladmin,DC=com
LockoutDuration             : 08:00:00
LockoutObservationWindow    : 08:00:00
LockoutThreshold            : 3
MaxPasswordAge              : 30.00:00:00
MinPasswordAge              : 7.00:00:00
MinPasswordLength           : 12
Name                        : Domain Admin Password Policy
ObjectClass                 : msDS-PasswordSettings
ObjectGUID                  : bef98e70-ba1a-48d9-b762-221f22784c88
PasswordHistoryCount        : 50
Precedence                  : 1
ReversibleEncryptionEnabled : False


PS C:\Windows\System32>
```

Figure 16.13: Properties of the password policy

We now have a new policy, and the next step is to assign objects to it. We need to add the Domain Admins security group to it:

```
Add-ADFineGrainedPasswordPolicySubject -Identity "Domain Admin Password
Policy" -Subjects "Domain Admins"
```

The preceding command adds the Domain Admins group to `Domain Admin Password Policy`. This can be verified using the following command:

```
Get-ADFineGrainedPasswordPolicy -Identity "Domain Admin Password
Policy" | Format-Table AppliesTo –AutoSize
```

This confirms what the policy target is. This value is saved under the `AppliesTo` attribute.

The following command is also useful to list down filtered password policy details:

```
Get-ADFineGrainedPasswordPolicy -Filter * | Format-Table
Name,Precedence,AppliesTo –AutoSize
```

This will list all the fine-grained password policies along with `Name`, `Precedence`, and the targets.

Pass-the-hash attacks

If a client needs to authenticate into a server successfully, the client needs to prove their identity. This is done by using a username and password. The client needs to present its username and password to the authentication server, and it will verify the identity. There are legacy protocols and systems that send this information in cleartext, even in an open network. Telnet is a good example of this. If someone is listening to traffic (packet capturing) on a Telnet session, they can easily capture a password as it is transmitted in cleartext.

Modern authentication protocols are well aware of these types of threats and use different technologies to encrypt credentials or create cryptographic hashes for identity verification. **Cryptographic hash** means a password string is transformed into a fixed-length digest using an algorithm.

Earlier, in the *Authentication in an AD environment* section, we saw how Kerberos authentication works using hash values. When we use hash values, the authentication server compares the hash value submitted by the client with the hash value for the user password that is stored in its database. In the Windows environment, these password hashes are stored in three different places:

- The **Security Account Manager** (**SAM**) database
- **Local Security Authority Subsystem Service** (**LSASS**)
- The AD database

The SAM database stores usernames and **New Technology** (**NT**) hashes in a `%SystemRoot%/system32/config/SAM` file. This contains all the hash values for accounts that are local to the computer. The current version of SAM does not store **LAN Manager** (**LM**) hashes in its database file.

The very first password hash schema introduced by Microsoft was LM. It uses the **Data Encryption Standard** (**DES**) algorithm for hashing. This is a legacy-weak schema, and Microsoft highly recommends not using it.

It doesn't support passwords larger than 15 ASCII characters, or passwords that are not case-sensitive. However, any new operating system released after Windows Vista supports the **Advanced Encryption Standard** (**AES**) algorithm for hashing.

Compared to a cleartext password, for an attacker, it is almost impossible to figure out a password based on the hash. Even if they are able to do it, it will take a lot of computing power and time. But, if they can find the hash value instead of retrieving the password, the hash value can be used to initiate a connection with the server on behalf of the original owner of the hash. This sounds easy, but in practice, it is still very difficult as these authentication protocols have their own mechanisms to prevent attackers from using someone else's hash value. If you consider Kerberos, it uses timestamps along with requests and responses to verify the authenticity of hashes. NT, **New Technology LAN Manager** (**NTLM**) v1, and NTLM v2 also use a similar challenge-response mechanism to authenticate without revealing the password.

However, even hash values are not transmitted directly. LSASS stores credentials in memory on behalf of users with active sessions. LSASS can store credentials in multiple forms, such as an NT hash, an LM hash, and Kerberos tickets. This is required in order to maintain active sessions and perform future authentications faster. This will clear up during the reboot, but it can be enough for the attacker to retrieve a hash value and use it to compromise the entire identity infrastructure. Microsoft introduced many features and techniques to protect the AD environment from **Pass-the-Hash** (**PtH**) attacks, and in this section, we are going to look into them in detail.

The Protected Users security group

The Protected Users security group was introduced with Windows Server 2012 R2 and continued in Windows Server 2022. This group was developed to provide highly privileged accounts with better protection from credential theft attacks. Members of this group have non-configurable protection applied. To use the Protected Users group, the **Primary Domain Controller** (**PDC**) should be running with a minimum of Windows Server 2012 R2 and the client computers should be running with a minimum of Windows 8.1 or Windows 2012 R2.

If a member of this group logs in to Windows 8.1, Windows Server 2012 R2, Windows 10, Windows Server 2016, Windows Server 2019, or Windows Server 2022, then we can expect the following:

- Members of this group cannot use NTLM, digest authentication, or CredSSP for authentication. Plain-text passwords are not cached. So, any of the devices using these protocols will fail to authenticate to the domain.

- Kerberos's long-term keys are not cached. For accounts in this group, the Kerberos protocol verifies authentication at each request (the TGT acquired at logon).
- Sign-in is offline. A cached verifier is not created at sign-in.

For the Protected Users group feature, it is *not* a must to have a domain or forest functional level running on Windows Server 2012 R2 or higher (Windows Server 2008 is the minimum because Kerberos needs to use AES). The only requirement is to run the PDC emulator's **Flexible Single-Master Operations** (**FSMO**) role in the Windows Server 2012 R2 domain controller.

If required, after the Protected Users group object is replicated to all the domain controllers, the PDC emulator role can be transferred to a domain controller running a lower Windows Server version.

If the AD environment uses Windows Server 2012 R2 or Windows Server 2016 domain functional levels, it provides additional protections with Protected User groups, such as the following:

- No NTLM authentication
- No DES or RC4 encryption in Kerberos, pre-authentication
- No delegation using the unconstrained or constrained method
- No Kerberos TGT is valid for more than 4 hours

Service accounts and computers cannot be members of the Protected Users security group. These accounts can be protected using different features, such as policy silos, which we will discuss later in the Authentication policies and authentication policy silos section.

To start with, we can review the `Protected Users` security group using the following command:

```
Get-ADGroup -Identity "Protected Users"
```

The following screenshot shows the output for the preceding command:

```
PS C:\Windows\System32> Get-ADGroup -Identity "Protected Users"

DistinguishedName : CN=Protected Users,CN=Users,DC=rebeladmin,DC=com
GroupCategory     : Security
GroupScope        : Global
Name              : Protected Users
ObjectClass       : group
ObjectGUID        : 685c8266-fdf3-4092-8207-c3a57a16f0fa
SamAccountName    : Protected Users
SID               : S-1-5-21-1495263175-677223849-3620536339-525
```

Figure 16.14: Properties of the Protected Users group

We can add users to the Protected Users group using ADAC, ADUC MMC, and PowerShell. This group is located in the default `Users` container in AD.

Here, we are going to add the user account of `Adam` to the `Protected Users` group using the following command:

```
Get-ADGroup -Identity "Protected Users" | Add-ADGroupMember –Members
"CN=Adam,CN=Users,DC=rebeladmin,DC=com"
```

The first part of the command retrieves the group and the second part adds the `Adam` user account to it.

After the user is added to the group, we can verify their group membership by using the following command:

```
Get-ADGroupMember -Identity "Protected Users"
```

To test this, we are going to use a tool called **Mimikatz** (`https://bit.ly/3oYOYfS`), which can be used to perform experiments with Windows security.

I logged in to a computer as the user `liam`, and he is not part of the `Protected Users` group. When I list keys from LSASS for users, I can see Liam's NTLM hash clearly:

```
Authentication Id : 0 ; 3059384 (00000000:002eaeb8)
Session           : Interactive from 3
User Name         : liam
Domain            : REBELADMIN
Logon Server      : REBEL-PDC-01
Logon Time        : 15/04/2017 08:35:20
SID               : S-1-5-21-4041220333-1835452706-552999228-1230
        msv :
         [00010000] CredentialKeys
         * NTLM     : 947e1646ca81470d18fdb6d976ba8d6a
         * SHA1     :aabc44618a0645c/ddd29ca5/f95bacc318/1b6
         [00000003] Primary
         * Username : liam
         * Domain   : REBELADMIN
         * NTLM     : 947e1646ca81470d18fdb6d976ba8d6a
         * SHA1     : aabc44618a0645c7ddd29ca57f95bacc3f1871b6
        tspkg :
        wdigest :
         * Username : liam
         * Domain   : REBELADMIN
         * Password : (null)
        kerberos :
         * Username : liam
         * Domain   : REBELADMIN.COM
         * Password : (null)
        ssp :
        credman :
```

Figure 16.15: Viewing a hash value using Mimikatz

When I do the same thing for the user `adam`, who is a member of the `Protected Users` group, I cannot see the NTLM hash stored in the LSASS memory because members who are in the protected group do not use NTLM and don't save any credentials in the cache:

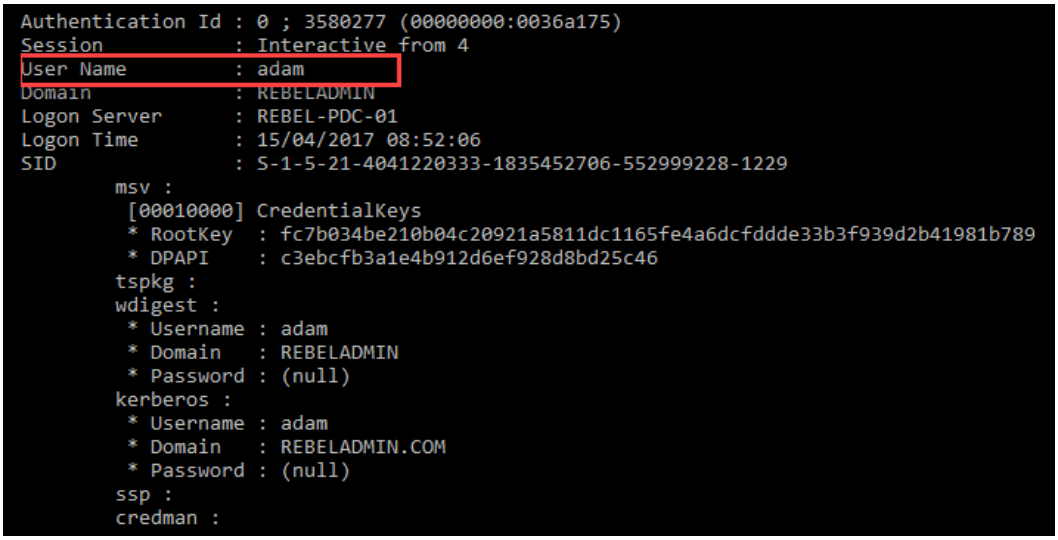


Figure 16.16: Viewing a hash value using Mimikatz of a user in the Protected Users group

The Protected Users security group was first introduced with Windows Server 2012 R2. It has built-in capabilities to protect privileged accounts from credential theft attacks. In this section, we learned about the technology behind the Protected Users security group. We also learned about the configuration of this security feature. In the next section, we are going to look into another built-in security feature that we can use to prevent the leakage of credentials over unsecured remote desktop connections.

Restricted admin mode for RDP

In a typical identity infrastructure attack, the first target is usually a regular user account or an endpoint. This is because highly privileged accounts and critical systems have advanced protection compared to end user devices (in most environments). A typical end user account does not have the privileges or capabilities to do much damage, but a privileged account does. Once an attacker completes an initial breach, the next thing they are looking to do is to get their hands on a privileged account.

If they start to mess around in an endpoint by doing things such as deleting files, increasing CPU/RAM usage, and damaging applications, then the end user will contact the IT department for help. IT department engineers are usually members of Enterprise Admins, Domain Admins, or at least a local administrator group of the endpoint. To log in and troubleshoot, engineers have to use their privileged accounts. If the attackers are running programs for password harvesting on the system, when engineers log in, they will be able to capture it.

In the previous section, we discussed how we can prevent credential hashes from being stored in LSASS. LSASS memory stores credentials when they do any of the following:

- Log in to a computer locally or using RDP
- Run an application or a task using the **Run As** option
- Run a Windows service on the computer with the service account
- Run a scheduled task or a batch job on the computer
- Run a task on the local computer using remote tools (system scans and installations)

RDP is the most commonly used method by engineers to access computers remotely. When a user connects to a system using RDP, it sends credentials to the remote computer in the form of cleartext. This is a security issue if the remote computer is already compromised. Microsoft introduced the *restricted admin mode for RDP* with Windows Server 2012 R2. When this mode is used for RDP, it will not send credentials to the remote computer. Once the user is logged in via the restricted RDP session, they cannot connect to other resources such as a shared network. Also, any users can't jump into other systems using RDP. This feature is also available in Windows Server 2022.

This feature can be used with Windows 7, Windows 8, Windows 8.1, Windows 10, Windows Server 2008 R2, Windows Server 2012, Windows Server 2012 R2, Windows Server 2016, Windows Server 2019, and Windows Server 2022.

By default, this mode is not enabled. Before you use it, it needs to be enabled in the target system. This can be done using the registry edit process:

1. Log in to the target computer or server as the administrator
2. Click on the Start menu, click on **Run**, type `regedit`, and then click on **OK**
3. In **Registry Editor**, browse to `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa`

4. Create the following registry key:

```
Name: DisableRestrictedAdmin
Type: REG_DWORD
Value: 0
```

The following screenshot illustrates the preceding step:

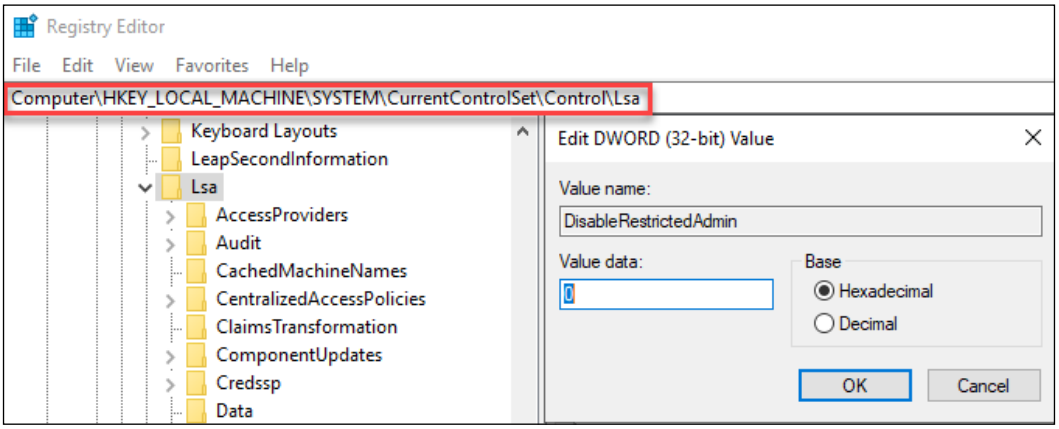


Figure 16.17: Registry key value

Once this is done, we can connect to the target computer using the restricted admin mode for RDP. To do that, the remote desktop client needs to run with the restricted mode. This can be done by running the `mstsc /restrictedadmin` command.

To test this feature, we will connect to a Windows 10 member PC using the restricted RDP mode. The user account belongs to Adam, and he is a member of the Domain Admins group:

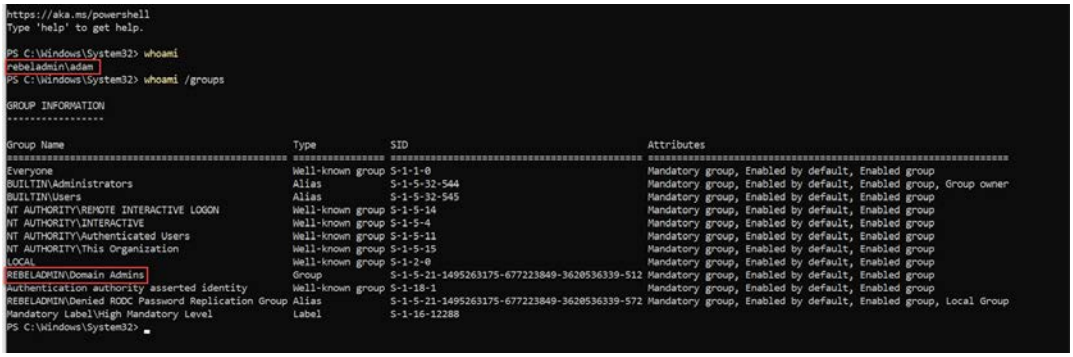


Figure 16.18: User's group memberships

Now, when we try to access another computer hard drive, it prompts with an **Access is denied** error. The user account is a domain admin, and it should not prevent access.

This is due to the restricted RDP mode, as it cannot be used to access other resources through the same session:

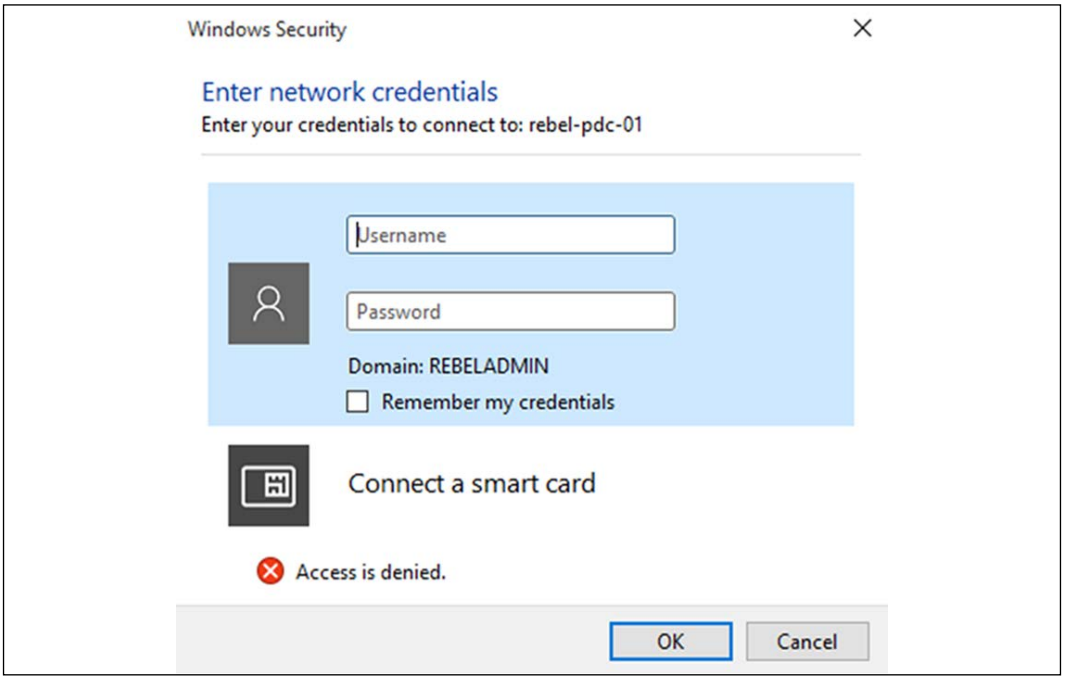


Figure 16.19: Login error

Similarly, I tried to add the domain controller to Server Manager on the remote computer. It issued an **Access denied** error as well:

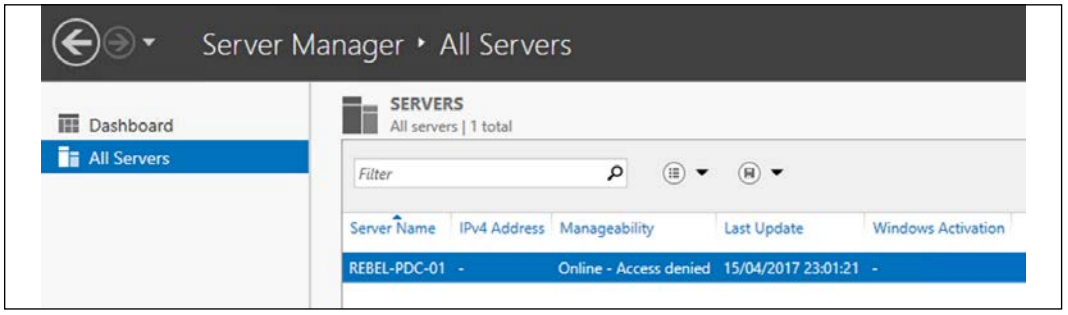


Figure 16.20: Server Manager access denied error

This feature minimizes the risk of highly privileged accounts being compromised.

The restricted RDP mode can be disabled by changing the value of the `DisableRestrictedAdmin` registry key to 1.

Authentication policies and authentication policy silos

The basic rule in PtH attack protection is to prevent trusted users from appearing on untrusted systems. Rebeladmin Corp. uses an MS SQL farm to run its databases. During the SQL Server setup, engineers use service accounts. It is obvious that these SQL service accounts should be used only with SQL Server. If the accounts appear on a receptionist's computer, something is definitely wrong. With Windows Server 2012 R2, Microsoft introduced authentication policies and policy silos that can be used to limit the use of highly privileged accounts to selected systems.

Authentication policies

Authentication policies can be used to specify the Kerberos protocol TGT validity period and access control conditions to restrict user sign-on.

Authentication policy silos

Authentication policy silos are similar to containers where we can assign user accounts, computer accounts, and service accounts. Then, these accounts can be managed by the authentication policies.

This feature requires the following prerequisites:

- All domain controllers in the domain must be based on Windows Server 2012 R2, Windows Server 2016, Windows Server 2019, or Windows Server 2022.
- The domain's functional level must be Windows Server 2012 R2 or higher
- Domain controllers must be configured to support **Dynamic Access Control** (**DAC**)
- Windows 8, Windows 8.1, Windows 10, Windows Server 2012, Windows Server 2012 R2, Windows Server 2016, Windows Server 2019, and Windows Server 2022 domain members must be configured to support DAC

Once all the prerequisites are in place, we can look into the configuration of authentication policies.

Creating authentication policies

Before we create policies, we need to enable DAC support for domain controllers and devices. DAC allows administrators to apply access control permissions and restrictions based on rules that can include the characteristics of the resources.

To enable DAC for domain controllers, perform the following steps:

1. Go to the Group Policy Management MMC.
2. Edit **Default Domain Controllers Policy**.
3. Go to **Computer Configuration** | **Policies** | **Administrative Templates** | **System** | **KDC**.
4. Click on **Enabled** to enable KDC support for claims, compound authentication, and Kerberos armoring.
5. Under **Options**, select **Always provide claims** and click on **OK**. This will ensure that it always returns claims for accounts and supports the RFC behavior to advertise **Flexible Authentication Secure Tunneling** (**FAST**):

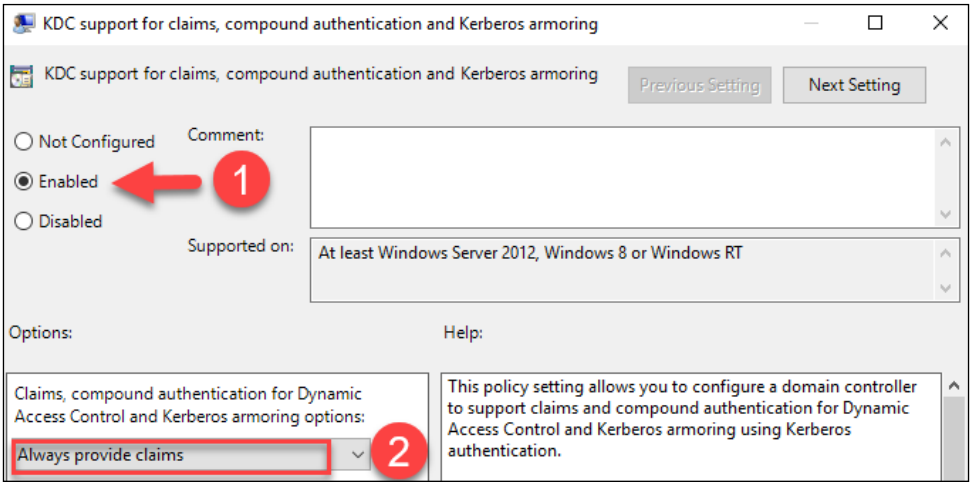


Figure 16.21: Enabling KDC support for claims, compound authentication, and Kerberos armoring policy setting

To enable DAC for computers, we need to do the following:

1. Go to the Group Policy Management MMC.
2. Edit **Default Domain Policy**.
3. Go to **Computer Configuration** | **Policies** | **Administrative Templates** | **System** | **Kerberos**.
4. Click on **Enabled** in Kerberos client support for claims, compound authentication, and Kerberos armoring.
5. Once this is done, we can create a new authentication policy using the `New-ADAuthenticationPolicy` cmdlet. This can also be created using ADAC:

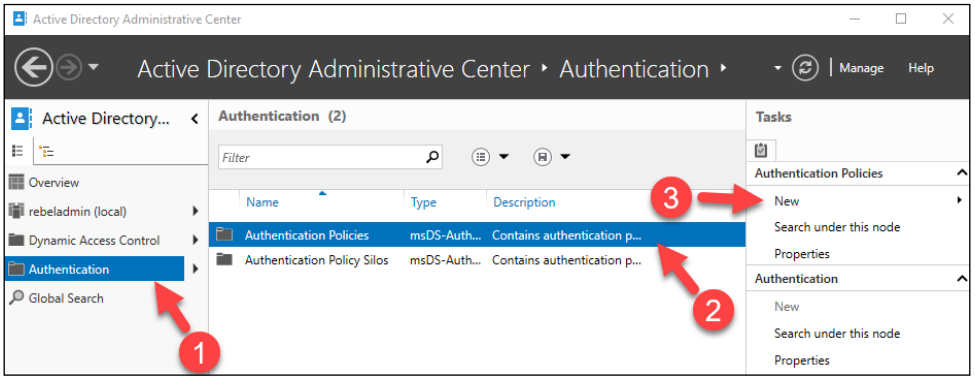


Figure 16.22: Creating a new authentication policy

As an example, let's create a new authentication policy called `AP_1hr_TGT` with a TGT lifetime of 60 minutes:

```
New-ADAuthenticationPolicy -Name "AP_1hr_TGT" -UserTGTLifetimeMins 60
-Enforce
```

In the preceding command, `-UserTGTLifetimeMins` defines the TGT lifetime for user accounts and the `-Enforce` parameter enforces policy restrictions.

Creating authentication policy silos

Now that we have created the authentication policy, the next step is to create a new authentication policy silo. My requirement is to create a policy silo to prevent the user account `Peter` from accessing `REBEL-PC01`.

Policy silos can be created using ADAC or the `New-ADAuthenticationPolicySilo` PowerShell cmdlet:

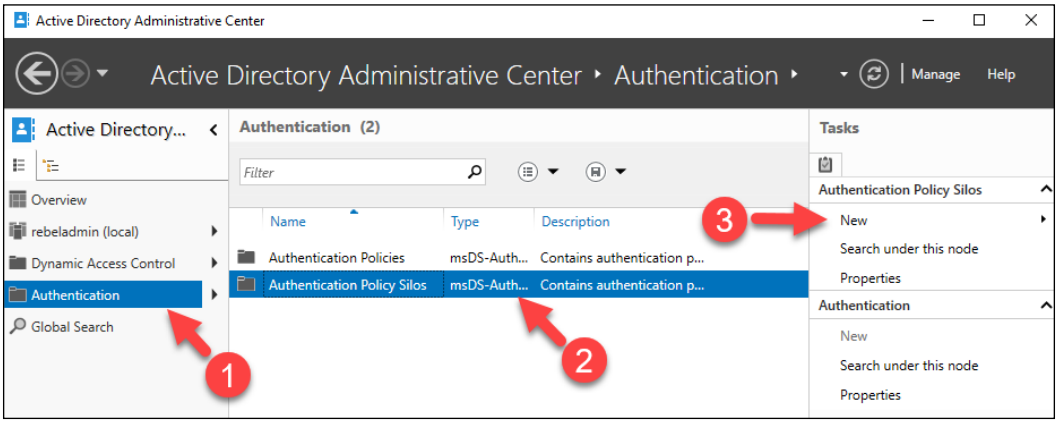


Figure 16.23: Creating new authentication policy silos

In this demo, let's create a new authentication policy silo called `Restricted_REBEL_PC01`:

```
New-ADAuthenticationPolicySilo -Name Restricted_REBEL_PC01
-UserAuthenticationPolicy AP_1hr_TGT -ComputerAuthenticationPolicy
AP_1hr_TGT -ServiceAuthenticationPolicy AP_1hr_TGT -Enforce
```

In the preceding command, `-UserAuthenticationPolicy`, `-ComputerAuthenticationPolicy`, and `-ServiceAuthenticationPolicy` refer to the authentication policies that will be attached to the policy silo. Here, we are only using one policy, but if needed, the policy silo can be attached to multiple authentication policies that cover the user, computer, and service classes.

The next step is to add the related objects to the policy silo as permitted accounts. In my demo, this is the user account of `Peter` and the computer called `REBEL-PC01`.

We can add these objects to the policy silos using the `Grant-ADAuthenticationPolicySiloAccess` PowerShell cmdlet:

```
Grant-ADAuthenticationPolicySiloAccess -Identity Restricted_REBEL_PC01
-Account Peter
```

The preceding command adds the user account `Peter` to the `Restricted_REBEL_PC01` policy silo as a permitted account.

We also can combine it with a filter and then add the result to the policy silo:

```
Get-ADComputer -Filter 'Name -like "REBEL-PC01"' | Grant-
ADAuthenticationPolicySiloAccess -Identity Restricted_REBEL_PC01
```

In the preceding command, we search for the computer object and then pass the result to the policy silo.

Once this is completed, we need to assign policy silos and the authentication policy to `Peter` and `REBEL-PC01`. This can be done using `Set-ADAccountAuthenticationPolicySilo`:

```
Set-ADAccountAuthenticationPolicySilo -Identity Peter
-AuthenticationPolicySilo Restricted_REBEL_PC01 -AuthenticationPolicy
AP_1hr_TGT
```

The preceding command assigns the `Restricted_REBEL_PC01` policy silo and the `AP_1hr_TGT` authentication policy to the user account `Peter`.

These commands can also be attached to filters:

```
Get-ADComputer -Filter 'Name -like "REBEL-PC01"' | Set-ADAccountAuth
enticationPolicySilo -AuthenticationPolicySilo Restricted_REBEL_PC01
-AuthenticationPolicy AP_1hr_TGT
```

The preceding command filters for the `REBEL-PC01` AD computer object and then assigns both the authentication policy and the policy silo.

The last step of the configuration is to define the access control condition for the `AP_1hr_TGT` authentication policy. This defines the condition of the device or host from which users log in. The condition for my demo will use the user's policy silo value:

```
Set-ADAuthenticationPolicy -Identity AP_1hr_TGT
-UserAllowedToAuthenticateFrom "O:SYG:SYD:(XA;OICI;CR;;;WD;(@USER.ad://
ext/AuthenticationSilo == `"Restricted_REBEL_PC01`"))"
```

In the preceding command, the condition is passed as a **Security Descriptor Definition Language** (**SDDL**) string. You can find more information about this SDDL string at `https://bit.ly/3FJCY92`.

This can also be modified by using the authentication policy properties window in ADAC:

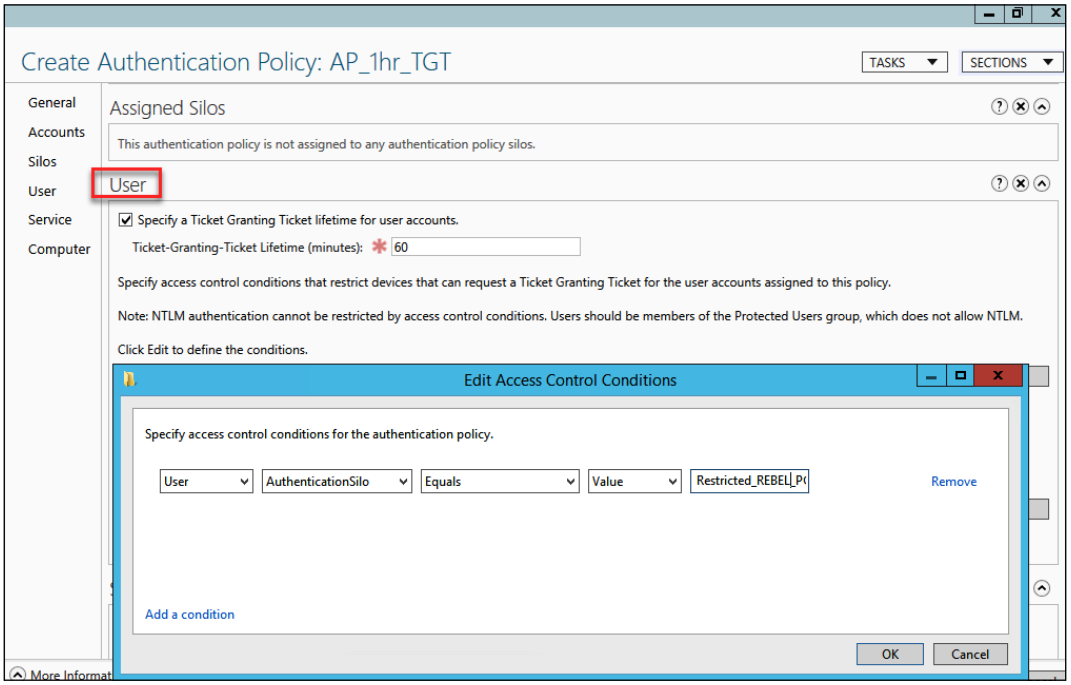


Figure 16.24: Edit Access Control Conditions

This finishes the configuration of the authentication policy silo and the authentication policy. Authentication policies and authentication policy silos provide greater flexibility in protecting privileged accounts on critical systems.

Secure LDAP

In an on-prem AD environment, there can be applications or services that require integration with AD. An AD-integrated application or service can query for AD users, authenticate, modify objects, and so on. This integration process is usually done using the **Lightweight Directory Access Protocol** (**LDAP**). By default, this LDAP connection between the client (application, service) and server (domain controller) is not encrypted. With this default configuration, a man-in-the-middle attacker can capture packets between the LDAP client and server, modify them, and then send the modified packets back to the server.

The LDAP server will not see the difference and reply to these forged requests with a decision. Microsoft is well aware of this vulnerability.

On August 13, 2019, Microsoft released a security advisory report, recommending to enable LDAP channel binding and LDAP signing to the LDAP client and server. For more information, refer to `https://bit.ly/3CRmr0B`.

By enabling secure LDAP (aka LDAPS, LDAP over SSL), we can encrypt LDAP data in transit and reject LDAP simple binds that are operating on a cleartext connection.

What are the characteristics of secure LDAP?

- Requires a valid certificate installed in domain controllers to enable secure LDAP. This can be a certificate issued from a private CA or public CA.
- After enabling secure LDAP, clients need to use port 636 to communicate with the server.
- If there are multiple certificates in the domain controller local computer store, the system will automatically select the latest certificate (the expiration date that is furthest in the future) for secure LDAP.
- The applications or services that are using a legacy LDAP connection method should also be compatible with secure LDAP. Once secure LDAP configuration is done, these application LDAP connection strings need to be updated to support secure LDAP connections.

Let's go ahead and see how we can enable secure LDAP.

Enable secure LDAP

In my demo environment, I have a Windows Server 2022 domain controller (DC01) and Windows Server 2022 member server. First, I am going to check LDAPS binding by using the LDP.exe tool. To use this tool, the member server should have RSAT AD tools installed. Otherwise, we can install it by using:

```
Install-WindowsFeature RSAT-AD-Tools -IncludeAllSubFeature
-IncludeManagementTools
```

To test the LDAPS connection:

1. Launch `ldp.exe`.
2. Go to **Connections** | **Connect**.

3. In the **Server** field, I typed the domain controller name, **DC01.rebeladmin.com**. I changed the port to **636** and selected **SSL** for a secure LDAP connection. Finally, I clicked on **OK**:

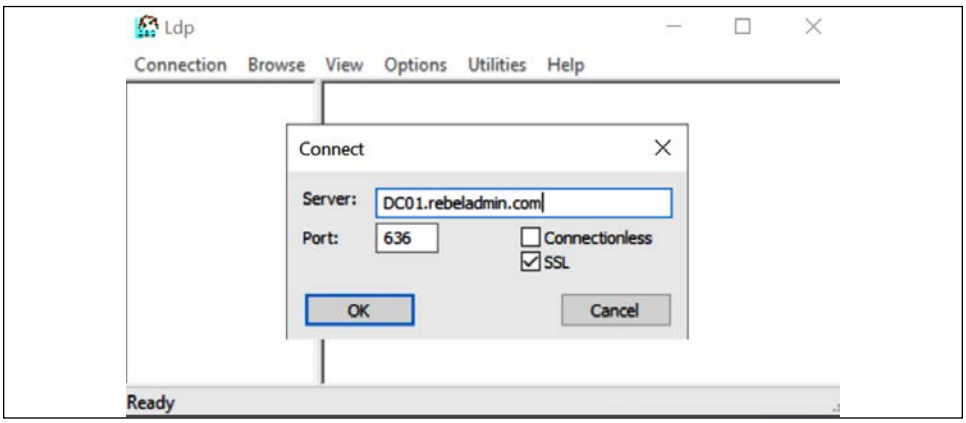


Figure 16.25: Testing a secure LDAP connection

4. Here, as expected, the connection has failed:

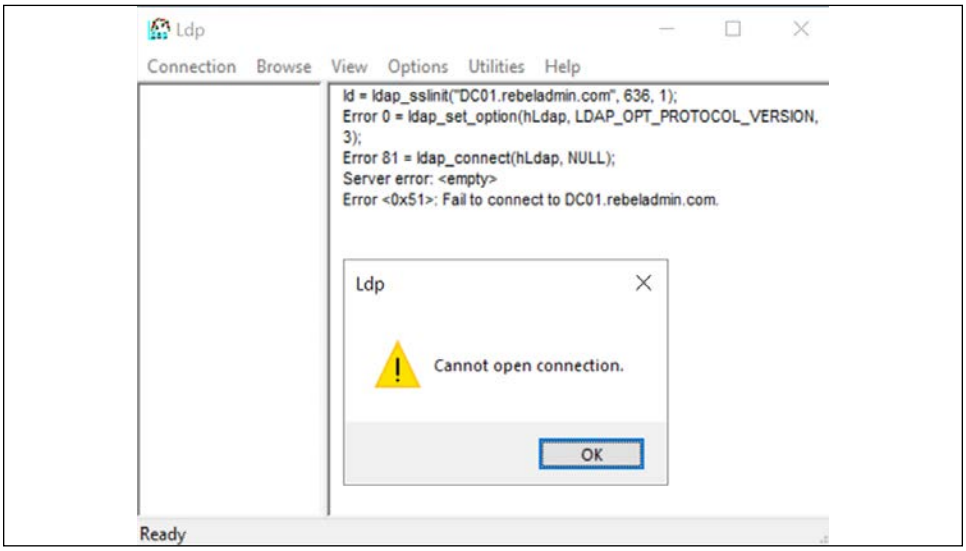


Figure 16.26: Secure LDAP connection fails

5. Then I went ahead and installed a wildcard certificate I had for `*.rebeladmin.com` in the `DC01` domain controller (under the local computer store):

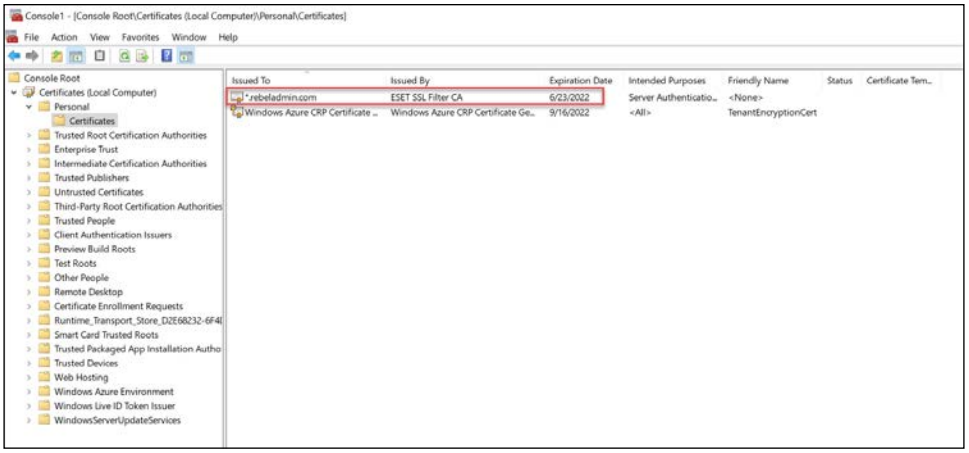


Figure 16.27: Installing a certificate in the domain controller

6. Then I went back to **SRV01** and tried the `ldp.exe` connection again with the same configuration. As expected, now I can connect to the server using secure LDAP:

Figure 16.28: Successful connection with secure LDAP

It is not required to restart the domain controller after installing the certificate (unless you are renewing a certificate).

This completes the secure LDAP enable process. But it will still allow users to use a simple bind using 389 ports. To disable this, we need to enable LDAP signing. More information about configuration is available at `https://bit.ly/2Zl8g6w`.

Microsoft Local Administrator Password Solution (LAPS)

In a business, when setting up new servers or computers, most of the time administrators are using one common password for the local administrator account. This account is usually used as a backdoor by administrators for software installation/uninstallation, to log in when domain authentication is not working, for operating system troubleshooting, and so on. Most of the time, this password is a non-complex one as well. I have seen some people use well-known passwords like "`Pa$$w0rd`" for local administrator accounts. When someone leaves the company, we usually change their domain password or disable their accounts. But these local administrator accounts remain the same, as changing passwords on local accounts is a time-consuming, complex process.

However, in a typical identity attack, a compromised local administrator account allows attackers to perform PtH attacks and laterally move within the organization by compromising more systems easily. Microsoft LAPS fixes this issue by setting a unique complex password for the local administrator account in all domain-joined devices. This local administrator account password set by Microsoft LAPS will automatically change according to the password policy. The new passwords will be saved in AD and authorized engineers can retrieve passwords from the AD server when required.

We do not need additional licenses or additional servers to implement this solution. This is a free tool. Microsoft LAPS needs a specific Group Policy **Client-Side Extension** (**CSE**) installed in each computer to do all management tasks.

Once LAPS is in place, the **Group Policy CSE** installed in each computer will update the local administrator password in the following order:

1. Generate a new password for the local administrator account.
2. Validate the new password with the password policy settings.
3. Save the password under the AD computer object's attribute **ms-Mcs-AdmPwd**. This attribute is added to the schema as part of the LAPS installation process.
4. Save the next expiry date of the password under the **ms-Mcs-AdmPwdExpirationTime** attribute. This attribute is also added to the schema as part of the LAPS installation process.
5. Change the administrator password.

In the next steps, I am going to demonstrate how we can implement Microsoft LAPS. To simplify the implementation process, I have categorized the tasks into the following steps:

1. Review the prerequisites
2. Install Microsoft LAPS
3. Update the AD schema
4. Change the computer object permissions
5. Assign permissions to the group for password access
6. Install CSE on computers
7. Create a GPO for LAPS settings
8. Testing

Before we start with the configuration, we need to verify if the current environment supports it.

Review prerequisites

We need to confirm the following before we go ahead with Microsoft LAPS configuration:

1. **Client operating systems –** Windows Server 2022, Windows Server 2019, Windows Server 2016, Windows 10, Windows Server 2012 R2, Windows Server 2012, Windows Server 2008 R2, Windows Server 2008, Windows Server 2003, Windows 7, Windows 8, Windows Vista, Windows 8.1
2. **AD –** Windows Server 2003 SP1 or later
3. **Management Tools –** PowerShell 2.0 or later, .NET Framework 4.0 or later

After verifying the prerequisites, the next step of the configuration is to install Microsoft LAPS.

Install Microsoft LAPS

Microsoft LAPS installation is a very straightforward process. To proceed:

1. Download the Microsoft LAPS package from `https://bit.ly/3xgJFfJ`.
2. This link has multiple `.msi` files. You need to download the `.msi` file that matches your setup. In my demo environment, I am going to use `LAPS.x64.msi`.

3. Double-click on the `LAPS.x64.msi` file. (You need to run this as an administrator.)
4. It will open a new wizard. On the initial screen, click **Next** to continue:

Figure 16.29: Microsoft LAPS installation wizard

5. Then in the next window, accept the license agreement and click on **Next** to proceed.
6. On the features window, deselect the default **AdmPwd GPO Extension** and select **Management Tools**. If you are also managing a local administrator account of the management server, you also need to install **AdmPwd GPO Extension**.

In my demo setup, I am installing it in a domain controller so I do not need it:

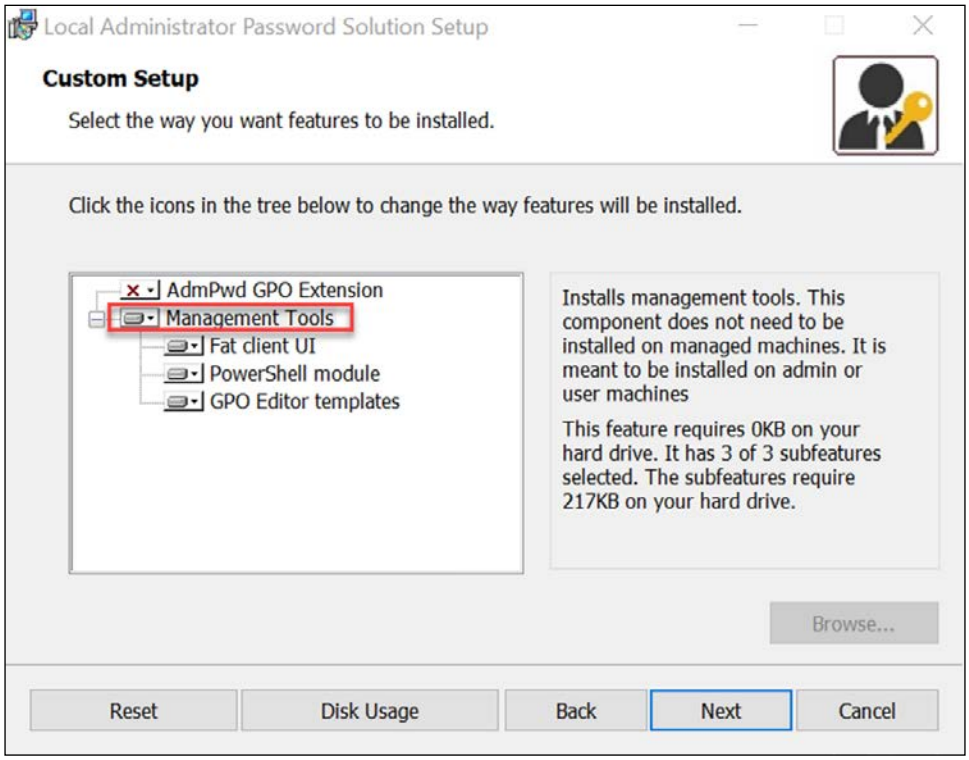


Figure 16.30: Installing Management Tools

7. On the next page, click on **Install** to begin the installation process.
8. Once installation is complete, click on **Finish**.

Once this is done, we need to update the AD schema to support Microsoft LAPS.

Update the AD schema

Microsoft LAPS uses two new attributes in computer objects:

1. **ms-Mcs-AdmPwd** – Save the administrator password in cleartext
2. **ms-Mcs-AdmPwdExpirationTime** – Save the timestamp of password expiration

To extend the AD schema:

1. Launch PowerShell as an AD schema administrator (I am using PowerShell 7.3)
2. Then, import the PowerShell module using

```
Import-module AdmPwd.PS
```

3. Once the module is imported successfully, run `Update-AdmPwdADSchema` to update the schema:

```
PS C:\Users\dfrancis> Update-AdmPwdADSchema

Operation           DistinguishedName                                                  Status
---------           -----------------                                                  ------
AddSchemaAttribute  cn=ms-Mcs-AdmPwdExpirationTime,CN=Schema,CN=Configuration,DC=reb… Success
AddSchemaAttribute  cn=ms-Mcs-AdmPwd,CN=Schema,CN=Configuration,DC=rebeladmin,DC=com  Success
ModifySchemaClass   cn=computer,CN=Schema,CN=Configuration,DC=rebeladmin,DC=com       Success

PS C:\Users\dfrancis>
```

Figure 16.31: Updating the AD schema

4. After the schema update, we can see these two new attributes in the computer object:

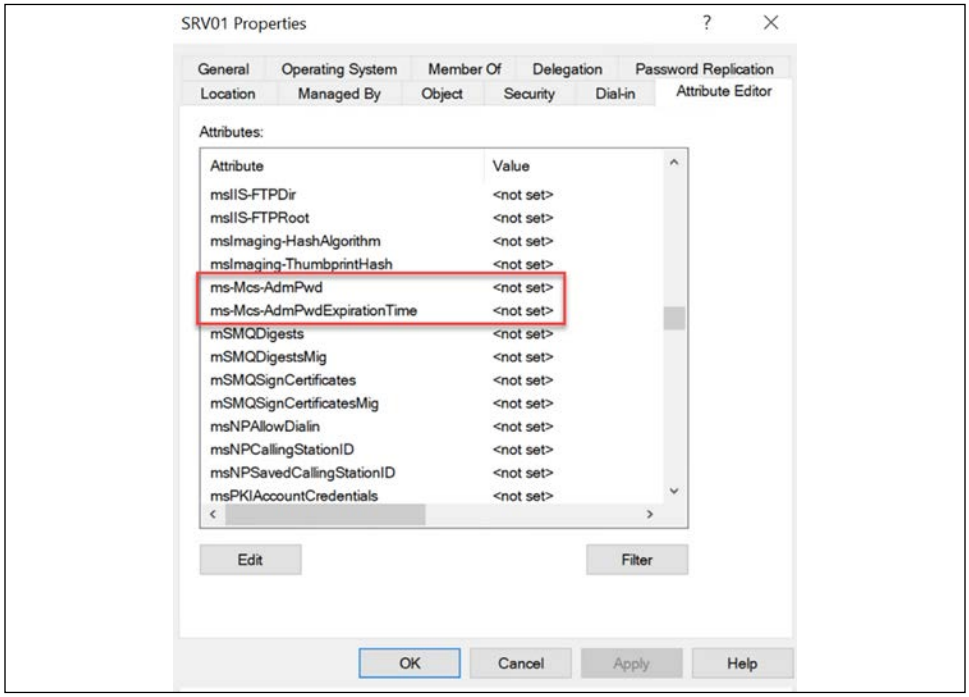


Figure 16.32: New attributes under the computer object

These values will be updated once we finish the rest of the configuration.

Change computer object permissions

During the password update process, the computer object itself should have permission to write values to `ms-Mcs-AdmPwd` and `ms-Mcs-AdmPwdExpirationTime` objects. To do that, we need to grant permissions to the SELF built-in account.

To do that:

1. Launch PowerShell as a domain administrator
2. Run the command `Set-AdmPwdComputerSelfPermission -OrgUnit RAServers`:

```
PS C:\Users\dfrancis> Set-AdmPwdComputerSelfPermission -OrgUnit RAServers

Name                DistinguishedName                                  Status
----                -----------------                                  ------
RAServers           OU=RAServers,DC=rebeladmin,DC=com                  Delegated

PS C:\Users\dfrancis>
```

Figure 16.33: Changing computer object permissions

In the above sample, RAServers is the OU I created for all the machine objects.

Assign permissions to groups for password access

In my demo environment, I have a security group called ITAdmins. I need users in this group to view the passwords for local administrators. Before we assign permissions, let's see who has privileges to view the passwords by default.

To do that:

1. Launch PowerShell as a domain administrator
2. Then import the PowerShell module using

```
Import-module AdmPwd.PS
```

3. After, to view the users/groups with extended rights, we need to run the following command:

```
Find-AdmPwdExtendedRights -Identity "RAServers"
```

```
PS C:\Windows\System32> Find-AdmPwdExtendedRights -Identity "RAServers"

ObjectDN                                    ExtendedRightHolders
--------                                    --------------------
OU=RAServers,DC=rebeladmin,DC=com           {NT AUTHORITY\SYSTEM, REBELADMIN\Domain Admins}

PS C:\Windows\System32>
```

Figure 16.34: Verifying extended rights

As we can see in the above figure, extended permissions are only applied to the Domain Admins group. It means a local administrator password for a computer object in the **RAServers** OU can only be accessed by a domain admin account. We need to grant the same permissions to the **ITAdmins** security group. To do that we can run:

```
Set-AdmPwdReadPasswordPermission -Identity "RAServers"
-AllowedPrincipals "ITAdmins"
```

4. The above command will add extended permissions to the **ITAdmins** security group. We can verify it by running `Find-AdmPwdExtendedRights -Identity "RAServers" | fl`:

```
PS C:\Windows\System32> Find-AdmPwdExtendedRights -Identity "RAServers" | fl

RunspaceId           : e7e145eb-a254-457a-a987-b93d3dfe28a8
ObjectDN             : OU=RAServers,DC=rebeladmin,DC=com
ExtendedRightHolders : {NT AUTHORITY\SYSTEM, REBELADMIN\Domain Admins, REBELADMIN\ITAdmins}


PS C:\Windows\System32>
```

Figure 16.35: Updated extended rights

Microsoft LAPS required a client-side extension installed on each device. As the next step, let's see how we can do that.

Install CSE in Computers

There are many different methods we can use to install an agent on a computer. But in this demo, I am going to use a GPO to publish and install the agent on computers:

1. Log in to the domain controller and launch the **GPMC** (**Group Policy Management Console**).
2. Create a new group policy under the **RAServers** OU.
3. Then, right-click on the group policy and click on **Edit**.
4. After that, go to **Computer Configuration** | **Policies** | **Software Settings** | **Software Installations**.
5. Right-click on it and select **New** | **Package**.
6. It will open up the explorer window. Then, browse to a network share that has the LAPS `.msi` file. In this demo, I am using the path `\\dc01\LAPS\ LAPS.x64.msi` and this share has read permissions to everyone.
7. Then, in the next window, select the deployment method as **Assigned**:

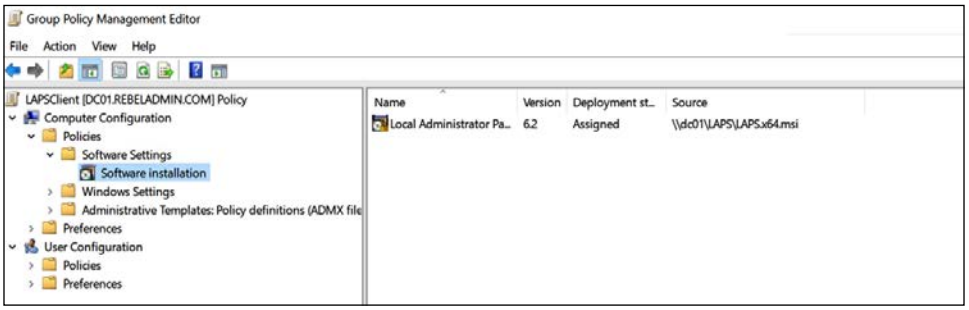


Figure 16.36: Installing an agent using a GPO

This will push agents to the computers under the **RAServers** OU. The installation requires rebooting on the client computer to complete the installation.

Create a GPO for LAPS settings

Now we have everything ready for Microsoft LAPS. The only thing left is to set up a new GPO with LAPS settings:

1. Log in to the domain controller and launch GPMC.
2. Create a new group policy under the **RAServers** OU.
3. Then, right-click on the group policy and click on **Edit**.

4. In a new window, go to **Computer Configuration** | **Administrative Templates** | **LAPS**. In there we can see four settings:

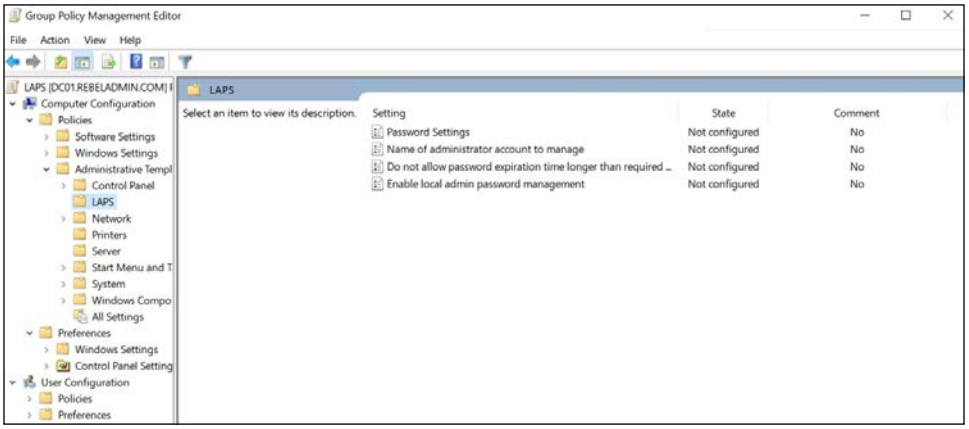


Figure 16.37: Microsoft LAPS GPO settings

5. Double-click on **Enable local admin password management**. Then click on **Enable** and click **OK** to apply settings. This will enable the password management feature.
6. After that, double-click on **Password Settings**. There we can define password complexity settings and password age. After all the settings are in place, click on **OK**:

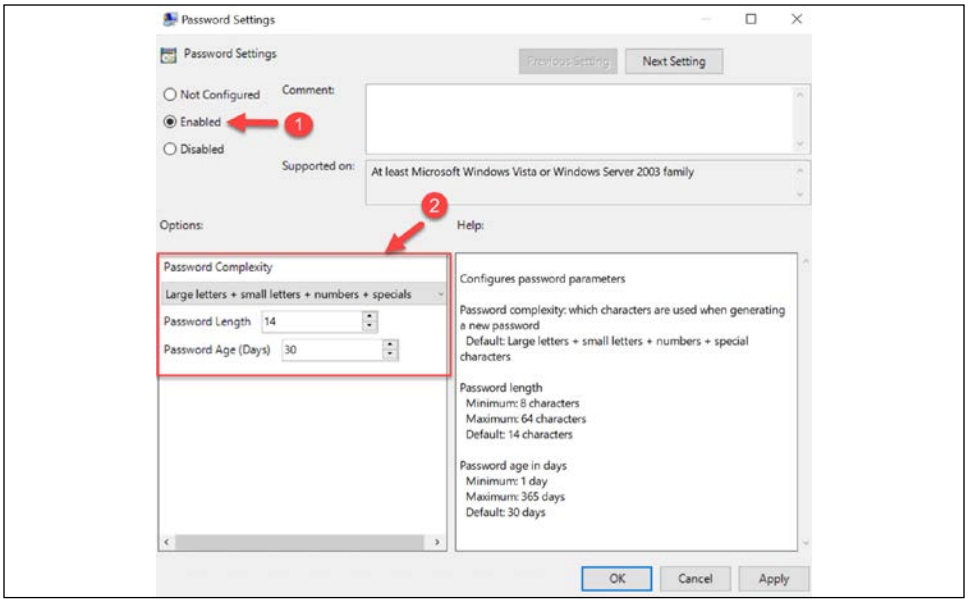


Figure 16.38: Microsoft LAPS GPO settings (password)

7. In my demo environment, the local administrator account is always `rebeladmin`. I need this account's password to be managed by LAPS. To configure that, click on **Name of administrator account to manage**. Then enable the setting and define the administrator account name:

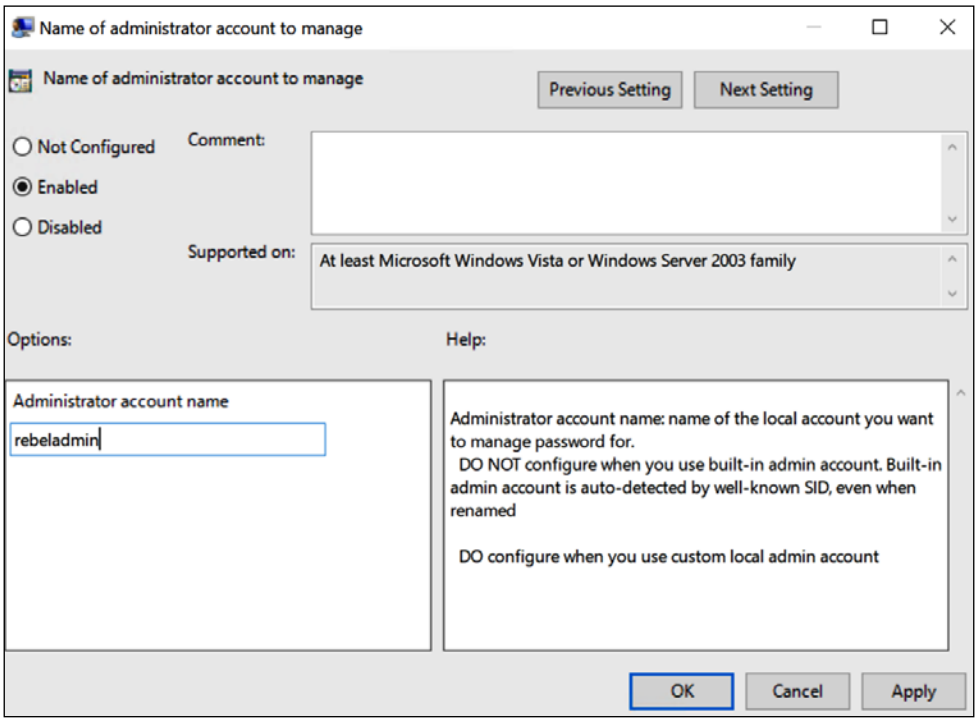


Figure 16.39: Microsoft LAPS GPO settings (Name of administrator account to manage)

8. Also, I want to make sure the local administrator password expiry times are no longer than what is defined by the policy. To enforce this, I double-click on **Do not allow password expiration time longer than required by policy** and enable the policy setting.

This completes the configuration process of Microsoft LAPS. After this policy is applied to endpoints, we can start testing.

Testing

Once the policy is applied, there are ways in which to see the local administrator password:

1. Log in to the LAPS management server as a member of the **ITAdmins** group
2. Launch **LAPS UI** from the programs

3. Then, type a test computer name and click on **Search**:

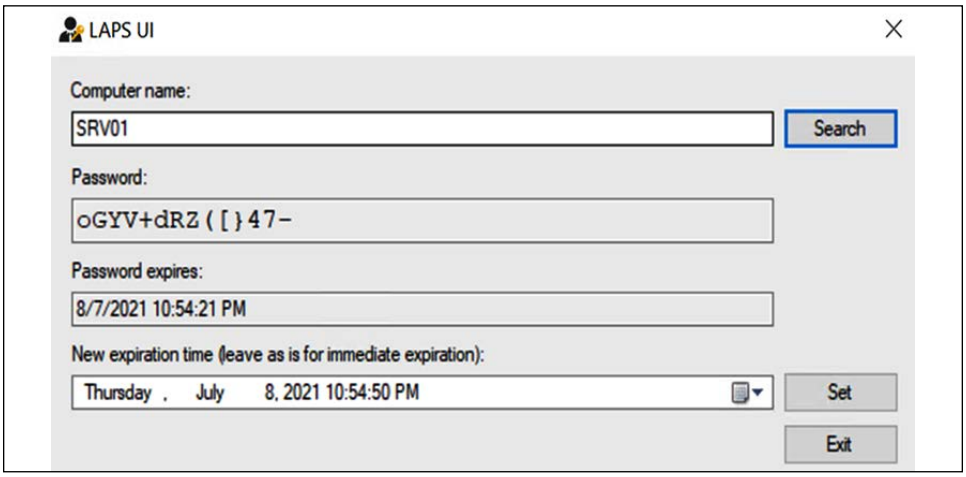


Figure 16.40: Checking the local administrator password using LAPS UI

As we can see, LAPS changed the password of the local administrator account.

4. We also can retrieve the password using the following PowerShell command, `Get-AdmPwdPassword -ComputerName SRV01`.
5. Note: Before running the command, make sure to import the `AdmPwd.PS` PowerShell module:

```
PS C:\Windows\System32> Get-AdmPwdPassword -ComputerName SRV01

ComputerName        DistinguishedName                          Password        ExpirationTimestamp
------------        -----------------                          --------        -------------------
SRV01               CN=SRV01,OU=RAServers,DC=rebeladmin,DC=com oGYV+dRZ([}47-    8/7/2021 10:54:21 PM
```

Figure 16.41: Checking the local administrator password using PowerShell

As we can see, Microsoft LAPS is automatically changing the local administrator password and recording it in AD. This helps to prevent lateral movement during an identity attack.

On-prem Azure AD Password Protection

In the very first chapter of this book, I mentioned the weaknesses of passwords and why we should consider passwordless authentication.

Passwords are guessable. As engineers, even if we enforce strong password policies, users always use passwords they can easily remember. Most of the time these passwords contain the company name, department name, product name, sports teams, current year, and so on. This makes it easier for attackers to guess the passwords based on dictionary attacks. Hackers don't break in, they log in, and unfortunately, dictionary attacks, particularly password spray attacks, are still one of the most commonly used techniques to find out the login details of an account.

Azure AD Password Protection can ban a list of common passwords globally to protect user accounts from dictionary attacks. However, this is not going to eliminate the threat of dictionary attacks because this all depends on the content of the password list and the list will be manually updated. When a policy is defined in Azure AD, it will apply to cloud users as well as on-prem AD users.

To enable this feature for an AD domain, we need to install and configure two components in an on-prem environment:

1. **Azure AD Password Protection proxy** (`AzureADPasswordProtectionProxySetup.exe`)
2. **Azure AD Password Protection DC agent** (`AzureADPasswordProtectionDCAgentSetup.msi`)

Apart from that, we also need an `Azure AD P1` licence or higher to enable this feature.

Azure AD Password Protection proxy

The role of the Azure AD Password Protection proxy is to forward a password policy download request from domain controllers to Azure AD and once it receives an update from Azure AD, return the response to the domain controllers. This can be run from any domain-joined computer that has access to the following websites:

1. `https://bit.ly/3rd4APw`
2. `https://enterpriseregistration.windows.net`

Domain controllers do not need to have internet access to enable this feature but domain controllers should be able to reach a proxy server on TCP port **135** (RPC).

The proxy server is also required to comply with the following:

1. Must run Windows Server 2012 R2 or later.
2. Should be running .NET Framework 4.7.2. This can be downloaded from `https://bit.ly/32vfN3R`.

3. Must allow outbound TLS 1.2 HTTP traffic.
4. It is recommended to run two proxy servers for high availability.

There is no minimum domain or forest functional level required to enable this feature.

Azure AD Password Protection DC agent

The role of the Azure AD Password Protection DC agent is to process password validate requests from clients. When a user is trying to change the password, the "password filter DLL" of the DC agent receives a password validation request from the operating system. Then filter DLL forward the request to DC agent service. After, the DC agent service validates the request using the current password policy, which is stored in the Sysvol folder, and responds back with a "pass" or "fail" result.

This agent is needed to be installed in domain controllers. If you have multiple domain controllers, it is recommended to get this agent installed in all the writable domain controllers because clients will be connected to any of the available domains. Read-only domain controllers do not process a password change or set events. Therefore, it is not required (not supported anyway) to install a DC agent in read-only domain controllers.

DC agents must use DFSR for sysvol replication. If you are running FRS replication, you must migrate to DFSR before enabling this feature. FRS is deprecated in newer domain controllers (2016, 2019, 2022) anyway.

How does Azure AD Password Protection work with AD?

Let's see how Azure AD Password Protection works when a user is trying to change their password:

1. When an Azure AD Password Protection proxy is installed, it will inform AD about it by creating a **serviceConnectionPoint** object. Azure AD Password Protection DC agents will query for this record to find proxy servers.
2. Let's assume a user is trying to change the password. This password validation only works when a user is trying to change their password or set a password. If there are accounts with the **password never expires** option, those will not validate. When the user tries to set a new password, a validation request will be forwarded to the DC agent.
3. The password filter DLL of the DC agent receives this request and forwards it to the DC agent service.

4. The DC agent validates the password with the password policy in the sysvol folder and replies with a "pass" or "fail" value. This policy updates every 1 hour. If the policy in sysvol is older than 1 hour, the DC agent queries for the **serviceConnectionPoint** object to find the proxy server in the environment. When the DC agent finds a proxy server, it sends a request to download the password policy.
5. Then the proxy server will go ahead and request the policy from Azure AD. Once it's received a response, the proxy server will go back to the DC agent with the new policy.
6. The DC agent will store it in the sysvol folder and other domain controllers will get a copy of it via sysvol replication.

Now we know how Azure AD Password Protection works and the next step is to configure this feature and test it.

Configuration

We are going to start the configuration process by installing an Azure AD Password Protection proxy. This component can be installed in any domain-joined computer that is running Windows Server 2012 R2 or later:

1. Download software using `https://bit.ly/3r6iMdq`.
2. Launch PowerShell as an administrator.
3. Navigate to the folder where `AzureADPasswordProtectionProxySetup.msi` is downloaded and run `msiexec.exe /i AzureADPasswordProtectionProxySetup.msi /quiet` to install the proxy.
4. After that, we need to register the proxy with Azure AD. To do that, we need to use an Azure global administrator account:

```
Import-Module AzureADPasswordProtection
Register-AzureADPasswordProtectionProxy -AccountUpn 'admin@
rebeladm.onmicrosoft.com'
```

   This registration process is one-time.

5. Then we also need to register the AD forest. This is also a one-time process. To do that we need an Azure AD global administrator or security administrator account:

```
Import-Module AzureADPasswordProtection
Register-AzureADPasswordProtectionForest -AccountUpn 'admin@
rebeladm.onmicrosoft.com'
```

After successful registration, the next task is to install DC agents. To do that:

1. Log in to the domain controller as a domain administrator
2. Download the software using `https://bit.ly/3xibfZM`
3. Launch PowerShell as an administrator
4. Navigate to the folder where `AzureADPasswordProtectionDCAgentSetup.msi` is downloaded and run `msiexec.exe /i AzureADPasswordProtectionDCAgentSetup.msi /quiet` to install the DC agent

It is required to restart to complete the installation. After the reboot, we can move to the next step of the configuration. In this step, we are going to configure Azure AD with a banned password list. To do that:

1. Launch the Azure portal, `https://bit.ly/3xg5o7o`, and log in as a global administrator.
2. Go to **Azure Active Directory | Security | Authentication Methods | Password protection**.
3. In there, select **Yes** under the **Custom banned passwords Enforce custom list** option.
4. Then, under the **Custom banned password list** textbox, type the patterns you'd like to ban. In this demo, I am going to enforce users not to use `rebeladmin` in their passwords:

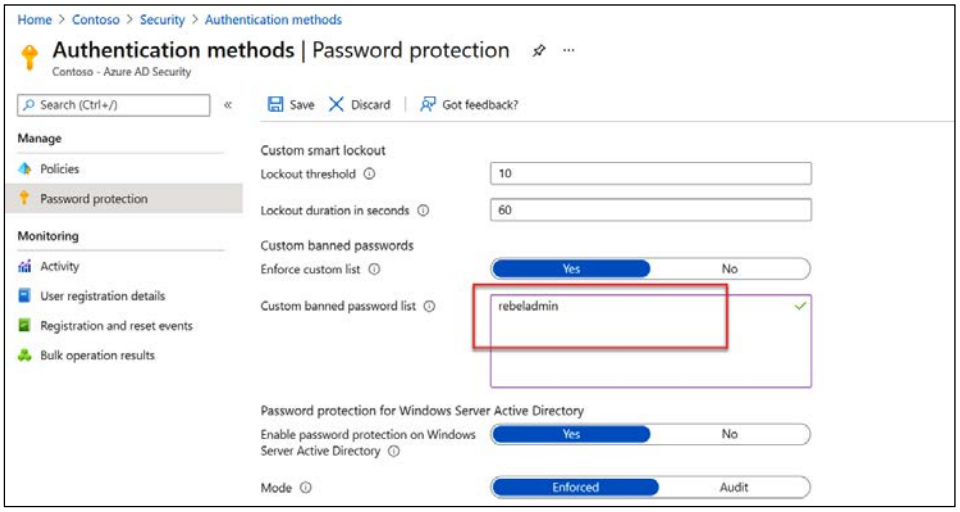


Figure 16.42: Enabling password protection on Windows AD

5. After the list updates, click on **Yes** under the **Enable password protection on Windows Server Active Directory** option. This will enable on-prem password protection.
6. Under the **Mode** option, select the mode you'd like to run. Here I am using enforced mode. For a more controlled approach, it is recommended to use audit mode as it will not prevent users from changing their passwords even if they breach the policy. Instead, an event log will be created, so we can see what sort of impact there is on a business that enables this policy.

After all the settings are in place, click on **Save** to update the settings.

This completes the configuration process and the next step is to do some testing.

Testing

I have logged in to another computer that is running Windows Server 2022, RSAT tools, and PowerShell 7.1. Then I launch PowerShell as an administrator and try to change the password for the user by using the following command:

```
Set-ADAccountPassword -Identity testuser -Reset -NewPassword
(ConvertTo-SecureString -AsPlainText "rebeladmin@A123" -Force)
```

In the preceding command, I am trying to change the password for the user "`testuser`" and set it to "`rebeladmin@A123`". But in my policy, I ban any password that has the text "rebeladmin" in it. When I run the command as expected, it said " Set-ADAccountPassword: The password does not meet the length, complexity, or history requirement of the domain."

Figure 16.43: Test user password change

As we can see here, the password protection policy is working. In this demo, the on-premises domain is not in the hybrid setup. So, a hybrid setup is not required to use this feature. You can implement this right away when you have relevant licenses and then think about a hybrid setup.

Summary

AD infrastructure security is a broad topic to cover in one chapter. AD security is not just dependent on AD DS; it is related to every layer of the OSI 7-layer model. At the beginning of the chapter, we learned about Kerberos authentication and what exactly happens behind the scenes when a user tries to access a resource in the AD environment. Then, we moved on to delegated permission control, where we learned about how we can delegate permissions to users, allowing them to only do specific administrative tasks. After that, we moved on to the *Pass-the-hash attacks* section, where we learned about PtH attacks.

Microsoft has introduced new tools and features that can be used to prevent PtH attacks. The Protected Users security group, restricted RDP mode, authentication policies, and authentication policy silos are some of them. In this chapter, we learned about how these tools work and how we can implement them in the AD environment. Then, we moved on to Microsoft LAPS. This solution can manage the local administrator password and save a copy of the password in AD.

After that, we learned about Azure AD Password Protection, which can be used to ban a list of common passwords globally.

In the next chapter, we will look into AD management with PowerShell.

17

Advanced AD Management with PowerShell

The very first **Active Directory** (**AD**) instance I set up was based on Windows Server 2003. It was a completely different approach from today's Active Directory installations. In Windows Server 2003, there were a lot of prerequisite tasks, such as installing a DNS role, setting up DNS zones, and adding the domain prefix. Even those tasks were directly related to **Active Directory Domain Services** (**AD DS**), and I had to configure them separately prior to running the `DCPORMO.exe` command. But today, the Active Directory role installation process is very straightforward. With basic knowledge and resources, anyone can get a domain controller up and running with a few clicks.

Microsoft has made server role installations and configurations easy over the years, not just AD DS. The main reason behind all these enhancements was to save time for engineers. Installations, configurations, and repetitive infrastructure tasks take up the majority of an engineer's time. Also with the pandemic, an engineer has to wear many hats as businesses shrink IT budgets. To save time on repetitive administrative tasks, we should be looking at automation technologies. In the early days, we used `DOS` commands, VBScript, and batch files to automate administrative tasks. But there were problems with that. Applications, server roles, and services had limitations on working with these automation technologies. Not every function available in the GUI supported the use of commands or scripts. This lack of support and lack of flexibility was holding engineers back from automating tasks.

To bring automation to the next level, Microsoft released a more flexible, more powerful, more integrated scripting language.

PowerShell 1.0 was the starting point and it was available to the public from November 2006. During the last decade, there have been a few versions released and it's now at version 5.1 (mainstream). Microsoft also released a separate version of PowerShell called PowerShell Core 6.0 (January 10, 2018). It was compatible with Linux and macOS as well. Now it has been replaced by PowerShell 7.0. In *Chapter 2*, *Active Directory Domain Services 2022*, I mentioned PowerShell 7 and throughout this book, I have used PowerShell 7 for configurations and the administration of Active Directory roles. In this chapter, I will explain how we can use PowerShell to further improve AD DS environment management.

We are also going to look at managing identities in a hybrid environment using Azure AD PowerShell and Microsoft Graph.

The cmdlets and scripts used in this chapter were written and tested in an environment that has the following:

- Windows Server 2022
- An AD domain and forest functional level set to Windows Server 2016
- PowerShell 7.1
- Azure AD Premium P2

In this chapter, we will cover the following topics:

- PowerShell scripts and commands that can be used to manage AD objects
- PowerShell scripts and commands that can be used to manage and troubleshoot AD replication
- PowerShell scripts and commands that can be used to manage identities in a hybrid environment using the Azure AD PowerShell module
- Microsoft Graph for Hybrid Identity management

Before we start using PowerShell for Active Directory management, we need to make sure relevant tools are installed and configured.

AD management with PowerShell – preparation

A PowerShell module includes assemblies, scripts, and functionalities. In order to use the functionalities, we need to import the module. After that, we can call for the contents of the module to manage relevant server roles, services, or features.

Before we start Active Directory management with PowerShell, first we need to import the `ActiveDirectory` module.

There are a few ways to do this. These include installing the AD DS server role or by installing **Remote Server Administration Tools** (**RSAT**):

- **AD DS server role**:

    1. If we install the AD DS server role using Server Manager, the Active Directory module for Windows PowerShell is installed as a feature:

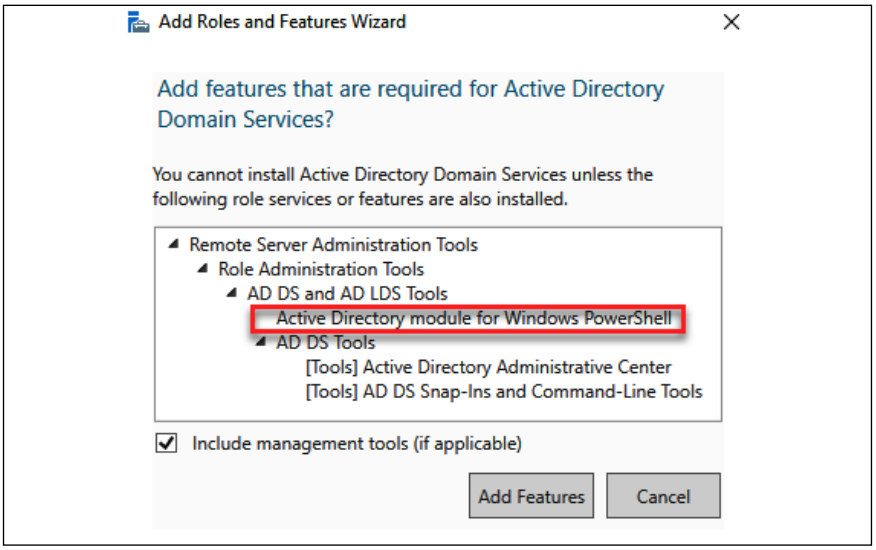


Figure 17.1: Active Directory module for Windows PowerShell feature

    2. If the AD DS role is installed using PowerShell, we need to include the management tools by using `-IncludeManagementTools`. Otherwise, by default, it will not install the module:

```
Install-WindowsFeature -Name AD-Domain-Services
-IncludeManagementTools
```

- **Remote Server Administration Tools**:

    1. Even if the server doesn't have the AD DS role installed, the existing domain environment can be managed using the AD DS PowerShell module. The AD PowerShell module is included with RSAT and can be installed using Server Manager or PowerShell.

2. On Server Manager, it can be found by navigating to **Features | Remote Server Administration Tools | Role Administration Tools | AD DS and AD LDS Tools | Active Directory module for PowerShell**, as shown in the following screenshot:

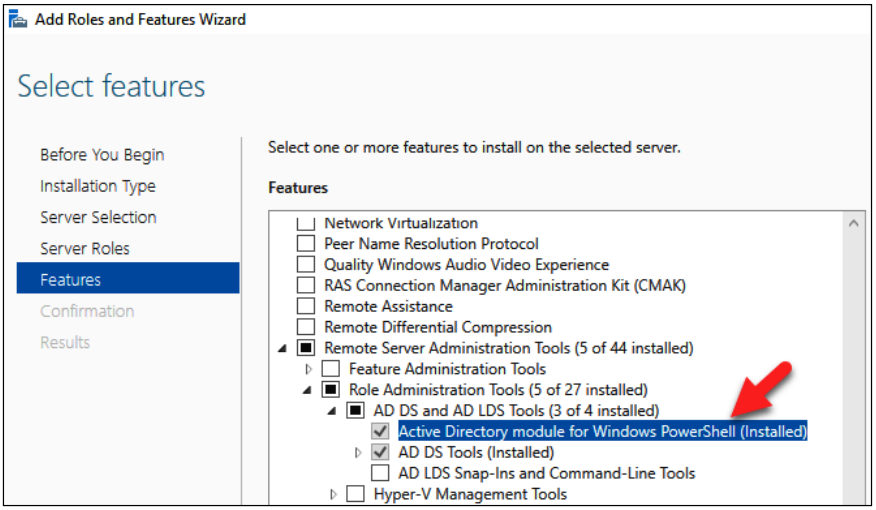


Figure 17.2: Active Directory module for Windows PowerShell feature under RSAT

3. It can also be installed using PowerShell:

```
Add-WindowsFeature RSAT-AD-PowerShell
```

It is also possible to install RSAT on the Windows desktop OS. As an example, RSAT for Windows 10 can be downloaded from `https://bit.ly/3HZ5k0W`.

PowerShell 7

In the preceding section, I explained how we can prepare native Windows PowerShell for Active Directory administration. Most of the scripts and commands used in this book also work with native Windows PowerShell. But PowerShell 7works in a different way. It doesn't come as part of Windows and needs to be installed separately as an application. To install PowerShell 7, please follow the following guide: `https://bit.ly/30Pu4HM`

After prerequisites are in place, we can list all the commands available under the module using the following command:

```
Get-Command -Module ActiveDirectory
```

There are about 147 commands under the module. The complete syntax for any command can be viewed using this command:

```
Get-Command commandname -Syntax
```

As an example, the following command will list the syntax for the `New-ADUser` command:

```
Get-Command New-ADUser -Syntax
```

The `Get-Help` command provides help for any command. As an example, the following command provides help for the `New-ADUser` command:

```
Get-Help New-ADUser
```

We also can view an example for the `New-ADUser` command using this:

```
Get-Help New-ADUser -Example
```

More information on the command can be viewed using this:

```
Get-Help New-ADUser -Detailed
```

Technical information on the command can be viewed using the following:

```
Get-Help New-ADUser -Full
```

Online information about the command can be viewed using this:

```
Get-Help New-ADUser -Online
```

In this section, we learned how to install the Active Directory module for PowerShell. We also learned about the basic functions of the module. Now, let's move on and further explore the Active Directory management capabilities of the module.

AD management commands and scripts

The module has 147 commands, and they can be used in countless different ways to manage the Active Directory environment. In this section, we will look at the capabilities of these commands and see how we can use them to improve Active Directory management.

I'd like to start this section by explaining how we can review the existing configuration of an Active Directory environment. The quick way to review the directory server configuration and capabilities is to use the following command:

```
Get-ADRootDSE
```

This command provides important information, such as forest and domain functional levels, the default naming context, the current time, and the currently logged-in domain controller.

The next step is to find the domain controllers in the domain. We can use the following to list the domain controller name, the IP address, the status of the global catalog server, and the **Flexible Single Master Operation** (**FSMO**) roles:

```
Get-ADDomainController -Filter * | Select-Object Name,IPv4Address,IsGlo
balCatalog,OperationMasterRoles
```

It is also important to know about the Active Directory site as it explains the physical topology of Active Directory:

```
Get-ADDomainController -Filter * | Select-Object Name,IPv4Address,Site
```

An Active Directory forest can have multiple domains. The following commands will list the forest names, the domain name, the domain controller, the IP address, and the Active Directory site:

```
$Forestwide = (Get-ADForest).Domains | %{ Get-ADDomainController
-Filter * -Server $_ }
write-output $Forestwide -Filter * | Select-Object Name,Forest,Domain,I
Pv4Address,Site
```

If we know the domain name, we can list the domain controllers and the **read-only domain controller** (**RODC**) using the following command:

```
$Domain = Read-Host 'What is your Domain Name ?'
Get-ADDomain -Identity $Domain | select ReplicaDirectoryServers,ReadOnl
yReplicaDirectoryServer
```

With this command, the system will ask the user to input the domain name. Once the user replies, it lists the domain controllers.

In the preceding command, `ReplicaDirectoryServers` represents the read and write domain controllers, and `ReadOnlyReplicaDirectoryServer` represents the RODCs.

Replication

Data replication is crucial for a healthy Active Directory environment. For a given domain controller, we can find its inbound replication partners using this:

```
Get-ADReplicationPartnerMetadata -Target REBEL-SRV01.rebeladmin.com
```

The preceding command provides a detailed description of the replication health of the given domain controller, including the last successful replication, replication partition, server, and so on.

We can list all the inbound replication partners for the given domain using the following command:

```
Get-ADReplicationPartnerMetadata -Target "rebeladmin.com" -Scope Domain
```

In the preceding command, the scope is defined as the domain. This can be changed to the forest to get a list of the inbound partners in the forest. The output is based on the default partition. If needed, the partition can be changed using `-Partition` to a configuration or schema partition. It will list the relevant inbound partners for the selected partition.

The associated replication failures for a site, forest, domain, and domain controller can be found using the `Get-ADReplicationFailure` cmdlet:

```
Get-ADReplicationFailure -Target REBEL-SRV01.rebeladmin.com
```

The preceding command will list the replication failures for the given domain controller.

Replication failures for the domain can be found using this:

```
Get-ADReplicationFailure -Target rebeladmin.com -Scope Domain
```

Replication failures for the forest can be found using the following command:

```
Get-ADReplicationFailure -Target rebeladmin.com -Scope Forest
```

Replication failures for the site can be found using the following command:

```
Get-ADReplicationFailure -Target LondonSite -Scope Site
```

In the preceding command, `LondonSite` can be replaced with a relevant site name.

Using both `Get-ADReplicationPartnerMetadata` and `Get-ADReplicationFailure`, I have created the following PowerShell script to generate a replication health report against a specific domain controller.

The first part of the script is used to define the objects that we'll use throughout the script:

```
## Active Directory Domain Controller Replication Status##
 $domaincontroller = Read-Host 'What is your Domain Controller?'
 ## Define Objects ##
 $report = New-Object PSObject -Property @{
 ReplicationPartners = $null
 LastReplication = $null
 FailureCount = $null
 FailureType = $null
 FirstFailure = $null
 }
```

In the preceding script, I have given an option for the engineer to specify the name of the domain controller:

```
$domaincontroller = Read-Host 'What is your Domain Controller?'
```

In the next part of the script, I am collecting the following data, which describes the replication connection status with the other domain controllers:

- Replication partner (`ReplicationPartners`)
- Last successful replication (`LastReplication`)

```
## Replication Partners ##
 $report.ReplicationPartners = (Get-ADReplicationPartnerMetadata
-Target $domaincontroller).Partner
 $report.LastReplication = (Get-ADReplicationPartnerMetadata -Target
$domaincontroller).LastReplicationSuccess
```

Then, I also gather the following data, which helps engineers to troubleshoot replication issues, if any exist:

- Active Directory replication failure count (`FailureCount`)
- Active Directory replication failure type (`FailureType`)
- Active Directory replication failure first recorded time (`FirstFailure`)

```
## Replication Failures ~##
 $report.FailureCount = (Get-ADReplicationFailure -Target
$domaincontroller).FailureCount
 $report.FailureType = (Get-ADReplicationFailure -Target
$domaincontroller).FailureType
 $report.FirstFailure = (Get-ADReplicationFailure -Target
$domaincontroller).FirstFailureTime
```

The last part of the script formats the output of the collected data:

```
## Format Output ##
 $report | select ReplicationPartners,LastReplication,FirstFailure,Fail
ureCount,FailureType | Out-GridView
```

The aforementioned script is displayed in an easy way for readers to understand. When it is used in PowerShell, make sure to prevent extra line spaces.

Further to Active Directory replication topologies, there are two types of replication:

- **Intra-site**: Replication between domain controllers in the same Active Directory site
- **Inter-site**: Replication between domain controllers in different Active Directory sites

We can review AD replication site objects using the `Get-ADReplicationSite` cmdlet. The following command returns all the Active Directory replication sites in the Active Directory forest:

```
Get-ADReplicationSite -Filter *
```

We can review Active Directory replication site links on the Active Directory forest using the following command:

```
Get-ADReplicationSiteLink -Filter *
```

In site links, the most important information is to know the site cost and the replication schedule. This allows us to understand the replication topology and expected delays in replication.

The following command lists all the replication site links, which includes the `CanadaSite` along with the site link name, link cost, and replication frequency:

```
Get-ADReplicationSiteLink -Filter {SitesIncluded -eq "CanadaSite"} |
Format-Table Name,Cost,ReplicationFrequencyInMinutes -AutoSize
```

A site link bridge can be used to bundle two or more site links and enable transitivity between site links.

Site link bridge information can be retrieved using the following command:

```
Get-ADReplicationSiteLinkBridge -Filter *
```

An AD site uses multiple IP subnets that are assigned to sites for its operations. It is important to associate these subnets with AD sites so that domain controllers know which computer is located at which site.

The following command will list all the subnets in the forest in a table with the subnet name and Active Directory site:

```
Get-ADReplicationSubnet -Filter * | Format-Table Name,Site -AutoSize
```

Bridgehead servers operate as the primary communication point to handle the replication data that comes in and goes out of the Active Directory site.

We can list all the preferred bridgehead servers in a domain:

```
$BHservers = ([adsi]"LDAP://CN=IP,CN=Inter-Site Transports,CN=Sites,CN=
Configuration,DC=rebeladmin,DC=com").bridgeheadServerListBL
$BHservers | Out-GridView
```

In the preceding command, the `bridgeheadServerListBL` attribute value is retrieved via the ADSI connection.

Information about the replication topology helps engineers in many ways, especially if engineers are troubleshooting Active Directory replication issues or performing an Active Directory audit. By using the preceding commands, I have created the following script to gather Active Directory replication topology data in one go.

As usual, the first part of the script is dedicated to defining objects:

```
## Script to gather information about Replication Topology ##
 ## Define Objects ##
 $replreport = New-Object PSObject -Property @{
 Domain = $null
 }
```

Before we move on to the replication, it is good to collect the Active Directory domain information. This is important if an organization is using multiple domains as we can easily separate the reports:

```
## Find Domain Information ##
 $replreport.Domain = (Get-ADDomain).DNSroot
```

I have used the next section of the script to list the Active Directory sites:

```
## List down the AD sites in the Domain ##
 $a = (Get-ADReplicationSite -Filter *)
 Write-Host "########" $replreport.Domain "Domain AD Sites" "#########"
 $a | Format-Table Description,Name -AutoSize
```

Then, I am going to collect data about the Active Directory replication site link and the Active Directory replication site link bridge by using the following:

```
## List down Replication Site link Information ##
 $b = (Get-ADReplicationSiteLink -Filter *)
 Write-Host "########" $replreport.Domain "Domain AD Replication
SiteLink Information" "########"
 $b | Format-Table Name,Cost,ReplicationFrequencyInMinutes -AutoSize
 ## List down SiteLink Bridge Information ##
 $c = (Get-ADReplicationSiteLinkBridge -Filter *)
 Write-Host "########" $replreport.Domain "Domain AD SiteLink Bridge
Information" "########"
 $c | select Name,SiteLinksIncluded | Format-List
```

In a computer network, there can be multiple IP subnets. These subnets need to be assigned correctly to Active Directory sites. This way, Active Directory Domain Controller computers know which site they belong to. This also has a direct impact on Active Directory replication. In the next section, we are going to collect Active Directory subnet information and the preferred bridgehead servers for the domain:

```
## List down Subnet Information ##
 $d = (Get-ADReplicationSubnet -Filter * | select Name,Site)
 Write-Host "########" $replreport.Domain "Domain Subnet Information"
"########"
 $d | Format-Table Name,Site -AutoSize
 ## List down Prefered BridgeHead Servers ##
 $e = ([adsi]"LDAP://CN=IP,CN=Inter-Site Transports,CN=Sites,CN=Configu
ration,DC=rebeladmin,DC=com").bridgeheadServerListBL
 Write-Host "########" $replreport.Domain "Domain Prefered BridgeHead
Servers" "########"
 $e
 ## End of the Script ##
```

The aforementioned script is displayed in a way that's easy for readers to understand. When it is used in PowerShell, make sure to prevent extra line spaces.

In the preceding script, we need to replace the ADSI connection with the relevant domain name:

```
$e = ([adsi]"LDAP://CN=IP,CN=Inter-Site Transports,CN=Sites,CN=Configur
ation,DC=rebeladmin,DC=com")
```

Healthy replication is critical for Active Directory Domain controllers. The time it takes to replicate a change to all the domain controllers depends on the number of domain controllers in place, geographical locations, replication schedules, etc. In some situations, we have to force the replication of objects and in the next section, we are going to look into it in detail.

Replicating a specific object

Once an object is added to a domain controller, it needs to be replicated to all other domain controllers. Otherwise, users will face issues during login using AD-integrated applications and services. The replication is dependent on many different factors, such as the replication schedule and intra-site connectivity. Sometimes, we need to force the replication between domain controllers:

```
## Replicate Object to From Domain Controller to Another ##
$myobject = Read-Host 'What is your AD Object Includes ?'
$sourcedc = Read-Host 'What is the Source DC ?'
$destinationdc = Read-Host 'What is the Destination DC ?'
$passobject = (Get-ADObject -Filter {Name -Like $myobject})
Sync-ADObject -object $passobject -source $sourcedc -destination
$destinationdc
Write-Host "Given Object Replicated to" $destinationdc
```

The preceding script will ask a few questions:

- **Name of object**: This need not be a **distinguished name** (**DN**). All that is needed is that text be included in the object name field.
- **Source DC**: The hostname of the source DC.
- **Destination DC**: The hostname of the destination DC.

Once the relevant information is provided, the object will be forcibly replicated:

```
PS C:\Windows\system32> ## Replicate Objects to Domain Controllers ##
$myobject = Read-Host 'What is your AD Object Includes ?'
$sourcedc = Read-Host 'What is the Soruce DC ?'
$destinationdc = Read-Host 'What is the Destination DC ?'
$passobject = (Get-ADObject -Filter {Name -Like $myobject})
Sync-ADObject -object $passobject -source $sourcedc -destination $destinationdc
Write-Host "Given Object Replicated to" $destinationdc
What is your AD Object Includes ?: Adam
What is the Soruce DC ?: REBEL-PDC-01
What is the Destination DC ?: REBEL-SRV01
Given Object Replicated to REBEL-SRV01

PS C:\Windows\system32>
```

Figure 17.3: Replicating a specific object

In this section of the chapter, we learned how the Active Directory module for PowerShell can be used to review the topology of an Active Directory environment. We also learned how we can audit, troubleshoot, and manage Active Directory replication using PowerShell. In the next section, we are going to look into Active Directory object management.

Users and groups

In this section, let's look at PowerShell commands and scripts that we can use to manage AD users and groups.

Last logon time

On certain occasions, we are required to find when a user successfully logs on to a domain. This can be for audit purposes or for troubleshooting purposes:

```
$username = Read-Host 'What is the User account you looking for ?'
   $dcs = Get-ADDomainController -Filter {Name -like "*"}
      foreach($dc in $dcs)
   {
     $hostname = $dc.HostName
     $user = Get-ADUser $userName -Server $hostname -Properties
lastLogon
     $lngexpires = $user.lastLogon
     if (-not ($lngexpires)) {$lngexpires = 0 }
     If (($lngexpires -eq 0) -or ($lngexpires -gt [DateTime]::MaxValue.
Ticks))
     {
       $LastLogon = "User Never Logged In"
     }
      Else
     {
       $Date = [DateTime]$lngexpires
       $LastLogon = $Date.AddYears(1600).ToLocalTime()
     }
  }
  Write-Host $username "last logged on at:" $LastLogon
```

The preceding script will ask for the username of the account and, once it is provided, the system will search for the `lastLogon` attribute value on all available domain controllers. If it cannot be found, it will return `User Never Logged In` or, if found, it will return the last logon timestamp.

Last login date report

Periodic housekeeping in AD is required for integrity. There may be user objects that have not been used for years. If we can create a report along with the last login dates, we can use it as a reference to clean up objects:

```
## Script For Filter user with Last logon Time ##
$htmlformat = "<style>BODY{background-color:LightBlue;}</style>"
Get-ADUser -Filter * -Properties "LastLogonDate" | sort-object
-property lastlogondate -descending | Select-Object Name,LastLogonDate
| ConvertTo-HTML -head $htmlformat -body "<H2>AD Accounts Last Login
Date</H2>"| Out-File C:\lastlogon.html
Invoke-Expression C:\lastlogon.html
```

This script creates an HTML report that includes all the user accounts with their last login date timestamps:

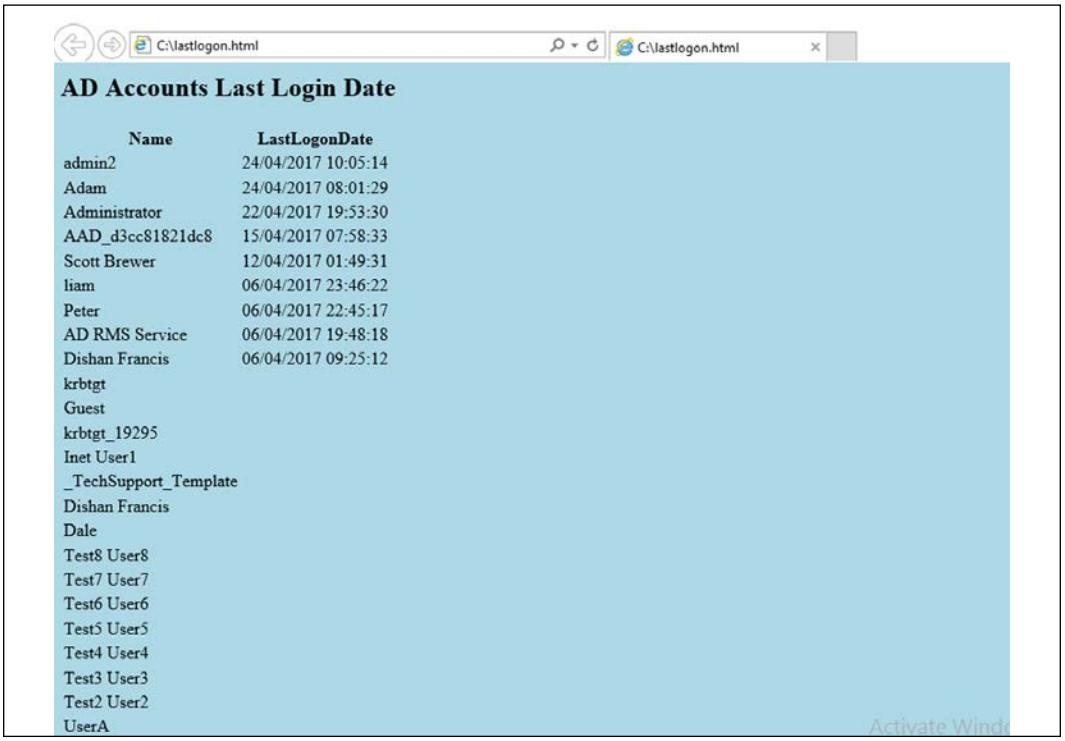


Figure 17.4: Last login date HTML report

Some of the accounts in the above reports don't show the last login date value. It means no one has logged into those accounts yet.

Login failures report

It is important to know about failed attempts to log in to the DC, not just the successful attempts. These can be a result of potentially malicious activity.

The following script will create a report to indicate the login failures on a given domain controller:

```
## Report for DC login Failures ##
$failedevent = $null
$Date= Get-date
$dc = Read-Host 'What is the Domain Controller ?'
$Report= "C:\auditreport.html"
$HTML=@"
<title>Failed Login Report for $dc</title>
<style>
BODY{background-color :LightBlue}
</style>
"@
 $failedevent = Get-Eventlog security -Computer $dc -InstanceId 4625
-After (Get-Date).AddDays(-7) |
 Select TimeGenerated,ReplacementStrings |
 % {
 New-Object PSObject -Property @{
 SourceComputer = $_.ReplacementStrings[13]
 UserName = $_.ReplacementStrings[5]
 SourceIPAddress = $_.ReplacementStrings[19]
 Date = $_.TimeGenerated
 }
 }
 $failedevent | ConvertTo-Html -Property SourceComputer,UserName,Sour
ceIPAddress,Date -head $HTML -body "<H2>Failed Login Report for $dc</
H2>"|
 Out-File $Report
 Invoke-Expression C:\auditreport.html
```

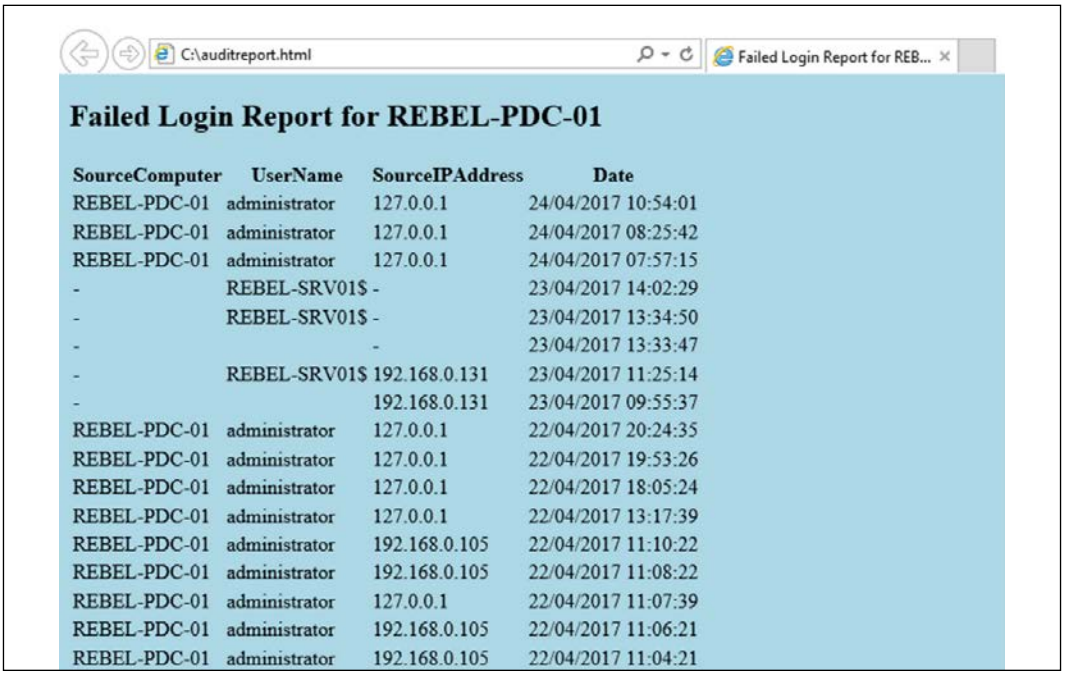
The aforementioned script is displayed in a way that's easy for readers to understand. When it is used in PowerShell, make sure to prevent extra line spaces.

When you run the preceding script, it will ask for the name of the domain controller that you wish to run this report against.

Then, in the background, it will search for event `4625` in the event viewer and then list the following data in a report:

- The source computer
- The username
- The source IP address
- The event time

The following screenshot shows the failed login report for REBEL-PDC-01:



Failed Login Report for REBEL-PDC-01

SourceComputer	UserName	SourceIPAddress	Date
REBEL-PDC-01	administrator	127.0.0.1	24/04/2017 10:54:01
REBEL-PDC-01	administrator	127.0.0.1	24/04/2017 08:25:42
REBEL-PDC-01	administrator	127.0.0.1	24/04/2017 07:57:15
-	REBEL-SRV01$	-	23/04/2017 14:02:29
-	REBEL-SRV01$	-	23/04/2017 13:34:50
-		-	23/04/2017 13:33:47
-	REBEL-SRV01$	192.168.0.131	23/04/2017 11:25:14
-		192.168.0.131	23/04/2017 09:55:37
REBEL-PDC-01	administrator	127.0.0.1	22/04/2017 20:24:35
REBEL-PDC-01	administrator	127.0.0.1	22/04/2017 19:53:26
REBEL-PDC-01	administrator	127.0.0.1	22/04/2017 18:05:24
REBEL-PDC-01	administrator	127.0.0.1	22/04/2017 13:17:39
REBEL-PDC-01	administrator	192.168.0.105	22/04/2017 11:10:22
REBEL-PDC-01	administrator	192.168.0.105	22/04/2017 11:08:22
REBEL-PDC-01	administrator	127.0.0.1	22/04/2017 11:07:39
REBEL-PDC-01	administrator	192.168.0.105	22/04/2017 11:06:21
REBEL-PDC-01	administrator	192.168.0.105	22/04/2017 11:04:21

Figure 17.5: Login failures report

The login failures records are different from one server to another. So when it comes to troubleshooting, make sure you select the correct domain controller.

Finding the locked-out account

If password policies are defined, accounts with a large number of login failures will be locked out. Locked-out accounts in an AD environment can be found using the following command:

```
Search-ADAccount -Lockedout | Select name,samAccountName,Lockedout
```

If any of those in the list need to be unlocked, we can use the `Unlock-ADAccount` cmdlet to unlock an account.

For an individual account, perform the following command:

```
Unlock-ADAccount tuser4
```

For all the accounts on the list, perform the following command:

```
Search-ADAccount -Lockedout | Unlock-ADAccount
```

It is not a good practice to unlock all the accounts unless there is a specific reason.

Password expire report

Issues due to expired passwords are a common support call type for helpdesks. The following script can generate a report about expiring passwords:

```
## Password Expire Report ##
$passwordreport = $null
$dc = (Get-ADDomain | Select DNSRoot).DNSRoot
$Report= "C:\passwordreport.html"
$HTML=@"
<title>Password Expire Report For $dc</title>
<style>
BODY{background-color :LightBlue}
</style>
"@
$passwordreport = Get-ADUser -filter * –Properties "SamAccountName"
,"pwdLastSet","msDS-UserPasswordExpiryTimeComputed" | Select-Object
-Property "SamAccountName",@{Name="Last Password Change";Expression={[d
atetime]::FromFileTime($_."pwdLastSet")}},@{Name="Next Password Change"
;Expression={[datetime]::FromFileTime($_."msDS-UserPasswordExpiryTimeCo
mputed")}}
$passwordreport | ConvertTo-Html -Property "SamAccountName","Last
Password Change","Next Password Change"-head $HTML -body "<H2>Password
Expire Report For $dc</H2>"|
Out-File $Report
Invoke-Expression C:\passwordreport.html
```

The aforementioned script is displayed in a way that's easy for readers to understand. When it is used in PowerShell, make sure to prevent extra line spaces.

This script will search for the attribute values for `SamAccountName`, `pwdLastSet`, and `msDS-UserPasswordExpiryTimeComputed` in every user object.

Then, they will be presented in an HTML report:

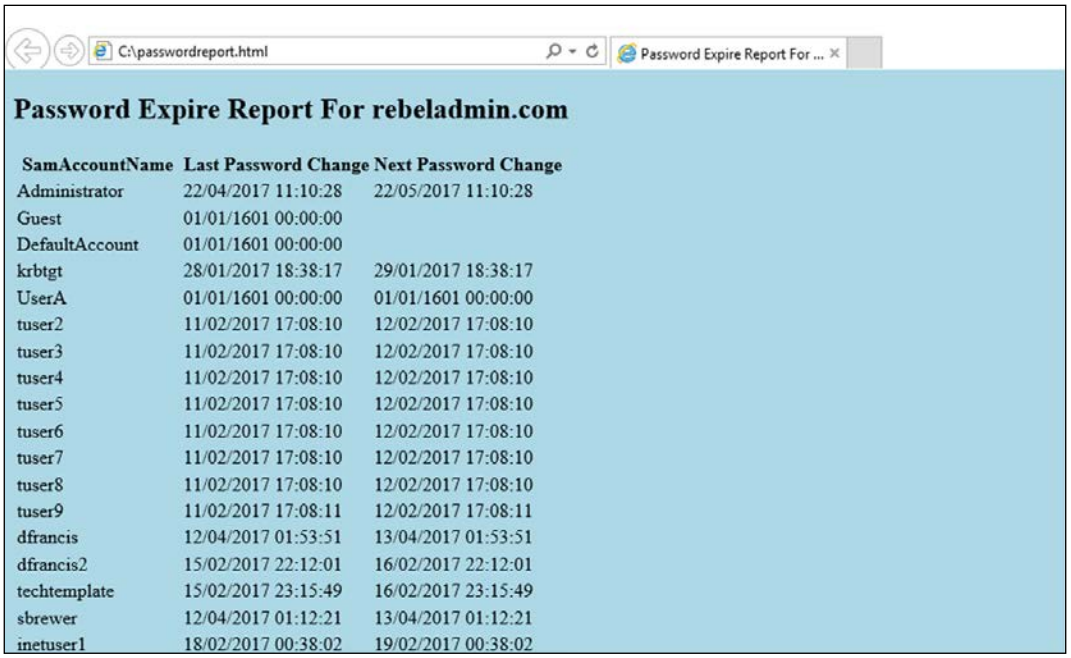


Figure 17.6: Password expire report

All these reports can run as scheduled jobs and were developed to be sent over as an email every week or month. This saves administrators time and also prevents mistakes that can occur with manual tasks.

Review the membership of the high-level administrative groups

As a security best practice, it is important to limit the number of privileged accounts used in an Active Directory environment. Sometimes we add users to privileged groups temporarily to do certain tasks and then forget to remove the permissions later on. Therefore, it is important to review members of the sensitive groups periodically and update those as required. In Active Directory, the following security groups are identified as sensitive groups:

- Enterprise Admins
- Schema Admins
- Domain Admins

- Account Operators (if present)
- Server Operators (if present)
- Print Operators (if present)
- DHCP Administrators
- DNSAdmins

To review the membership of a sensitive group we can use the following command:

```
Get-ADGroupMember -Identity "Domain Admins"
```

In the preceding command, `Domain Admins` is the group name and it can be replaced with any other sensitive group name. Once we have the list of members, the next step is to find out if those accounts are actively used. To do that we can use the value of the `LastLogonDate` user attribute. By considering all of the above requirements, I created the following scripts to list down all the sensitive groups with their members and `LastLogonDate` values:

```
## Sensitive Group Report ##
$HTML=@"
<title>Sensitive Groups Membership Report</title>
<style>
BODY{background-color :LightBlue}
</style>
"@
$enterpiseadmins = Get-ADGroupMember -Identity "Enterprise Admins"
| where {$_.objectclass -eq 'user'} | Get-ADUser -Properties
LastLogonDate | select Name,LastLogonDate | ConvertTo-Html -Property
"Name","LastLogonDate" -Fragment -PreContent "<h2>Enterprise Admins</
h2>"
$schemaadmins = Get-ADGroupMember -Identity "Schema Admins"
| where {$_.objectclass -eq 'user'} | Get-ADUser -Properties
LastLogonDate | select Name,LastLogonDate | ConvertTo-Html -Property
"Name","LastLogonDate" -Fragment -PreContent "<h2>Schema Admins</h2>"
$domainadmins = Get-ADGroupMember -Identity "Domain Admins"
| where {$_.objectclass -eq 'user'} | Get-ADUser -Properties
LastLogonDate | select Name,LastLogonDate | ConvertTo-Html -Property
"Name","LastLogonDate" -Fragment -PreContent "<h2>Domain Admins</h2>"
$accountoperators = Get-ADGroupMember -Identity "Account Operators"
| where {$_.objectclass -eq 'user'} | Get-ADUser -Properties
LastLogonDate | select Name,LastLogonDate | ConvertTo-Html -Property
"Name","LastLogonDate" -Fragment -PreContent "<h2>Account Operators</
h2>"
$serveroperators = Get-ADGroupMember -Identity "Server Operators"
| where {$_.objectclass -eq 'user'} | Get-ADUser -Properties
LastLogonDate | select
```

```
Name,LastLogonDate | ConvertTo-Html -Property "Name","LastLogonDate"
-Fragment -PreContent "<h2>Server Operators</h2>"
$printoperators = Get-ADGroupMember -Identity "Print Operators"
| where {$_.objectclass -eq 'user'} | Get-ADUser -Properties
LastLogonDate | select Name,LastLogonDate | ConvertTo-Html -Property
"Name","LastLogonDate" -Fragment -PreContent "<h2>Print Operators</h2>"
$dnsadmins = Get-ADGroupMember -Identity "DnsAdmins" | where {$_.
objectclass -eq 'user'} | Get-ADUser -Properties LastLogonDate | select
Name,LastLogonDate | ConvertTo-Html -Property "Name","LastLogonDate"
-Fragment -PreContent "<h2>DNS Admins</h2>"
$Reportvalues = ConvertTo-HTML -Body "$enterpiseadmins $schemaadmins
$domainadmins $accountoperators $serveroperators $printoperators
$dnsadmins" -Head $HTML
$Reportvalues | Out-File "C:\sensativegroupreport.html"
```

The preceding script will produce an HTML report similar to below.

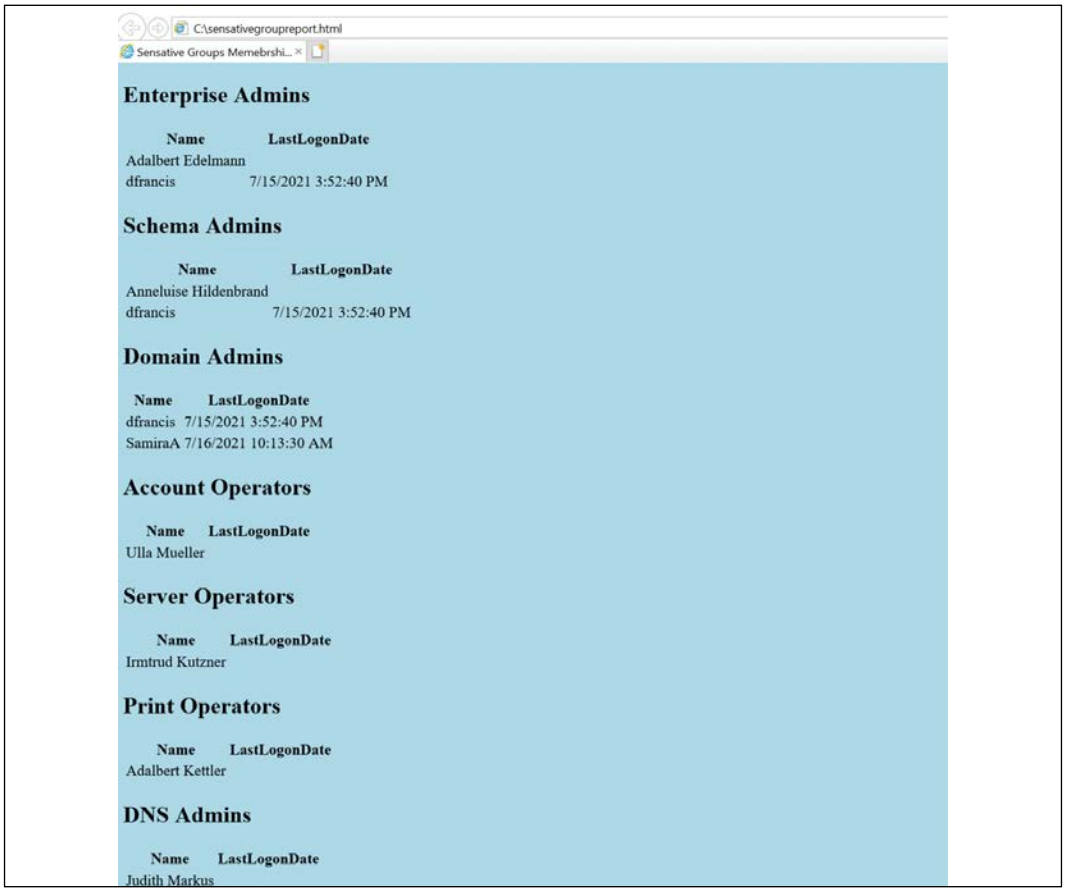


Figure 17.7: Sensitive group memberships

In this report, the `LastLogonDate` value can be used as a reference and if the account is not used often, we can go ahead and remove it from the sensitive groups.

In the script, I didn't mention `DHCP Administrators` as it is a local group.

In the above report, we are listing all the users in sensitive groups. Then it is up to engineers to tidy up the group membership. However, we can further develop this script and list down users who haven't logged in for a certain number of days. Then engineers can use it as a starting point to update the group memberships.

```
## Sensitive Group Members Inactive for 30 days ##
$30Days = (get-date).adddays(-30)
$HTML=@"
<title>Sensitive Groups Memebrship Report : USers Inactive for 30
days</title>
<style>
BODY{background-color :LightBlue}
</style>
"@
$enterpiseadmins = Get-ADGroupMember -Identity "Enterprise Admins"
| where {$_.objectclass -eq 'user'} | Get-ADUser -Properties
LastLogonDate |Where {$_.LastLogonDate -le $30Days}| select
Name,LastLogonDate | ConvertTo-Html -Property "Name","LastLogonDate"
-Fragment -PreContent "<h2>Enterprise Admins</h2>"
$schemaadmins = Get-ADGroupMember -Identity "Schema Admins" | where
{$_.objectclass -eq 'user'} | Get-ADUser -Properties LastLogonDate
| Where {$_.LastLogonDate -le $30Days}| select Name,LastLogonDate |
ConvertTo-Html -Property "Name","LastLogonDate" -Fragment -PreContent
"<h2>Schema Admins</h2>"
$domainadmins = Get-ADGroupMember -Identity "Domain Admins" | where
{$_.objectclass -eq 'user'} | Get-ADUser -Properties LastLogonDate
| Where {$_.LastLogonDate -le $30Days}| select Name,LastLogonDate |
ConvertTo-Html -Property "Name","LastLogonDate" -Fragment -PreContent
"<h2>Domain Admins</h2>"
$accountoperators = Get-ADGroupMember -Identity "Account Operators"
| where {$_.objectclass -eq 'user'} | Get-ADUser -Properties
LastLogonDate | Where {$_.LastLogonDate -le $30Days}| select
Name,LastLogonDate | ConvertTo-Html -Property "Name","LastLogonDate"
-Fragment -PreContent "<h2>Account Operators</h2>"
$serveroperators = Get-ADGroupMember -Identity "Server Operators"
| where {$_.objectclass -eq 'user'} | Get-ADUser -Properties
LastLogonDate | Where {$_.LastLogonDate -le $30Days}| select
Name,LastLogonDate | ConvertTo-Html -Property "Name","LastLogonDate"
```

```
-Fragment -PreContent "<h2>Server Operators</h2>"
$printoperators = Get-ADGroupMember -Identity "Print Operators" | where
{$_.objectclass -eq 'user'} | Get-ADUser -Properties LastLogonDate
| Where {$_.LastLogonDate -le $30Days}| select Name,LastLogonDate |
ConvertTo-Html -Property "Name","LastLogonDate" -Fragment -PreContent
"<h2>Print Operators</h2>"
$dnsadmins = Get-ADGroupMember -Identity "DnsAdmins" | where {$_.
objectclass -eq 'user'} | Get-ADUser -Properties LastLogonDate | Where
{$_.LastLogonDate -le $30Days}| select Name,LastLogonDate | ConvertTo-
Html -Property "Name","LastLogonDate" -Fragment -PreContent "<h2>DNS
Admins</h2>"
$Reportvalues = ConvertTo-HTML -Body "$enterpiseadmins $schemaadmins
$domainadmins $accountoperators $serveroperators $printoperators
$dnsadmins" -Head $HTML
$Reportvalues | Out-File "C:\inactiveusers.html"
```

In the above, I am creating an HTML report that lists down the members of sensitive groups who haven't logged in for the last 30 days.

Dormant accounts

In Active Directory, at least 10% of user accounts are dormant (inactive) accounts. These accounts can represent:

- Test accounts
- Contractors
- Former employees
- Disabled accounts

It is important to review these dormant accounts periodically and remove all unnecessary accounts from Active Directory as they are a possible security threat. If it is not possible to remove some of these accounts, at least remove them from sensitive groups and disable the accounts.

We can find these accounts in Active Directory by looking at the `LastLogonDate` attribute value and the account status. By considering these requirements, I created the following script to find dormant accounts:

```
## Dormant Accounts ##
$InactiveDate = (Get-Date).Adddays(-30)
$HTML=@"
<title>Dormant Accounts Report</title>
<style>
BODY{background-color :LightBlue}
```

```
</style>
"@
$disabledaccounts = Get-ADUser -Filter {Enabled -eq $false} | select sa
mAccountName,GivenName,Surname | ConvertTo-Html -Property "samAccountNa
me","GivenName","Surname" -Fragment -PreContent "<h2>Disabled Account</
h2>"
$inactiveaccounts = Get-ADUser -Filter {LastLogonDate -lt $InactiveDate
-and Enabled -eq $true} -Properties LastLogonDate | select samAccountN
ame,GivenName,Surname,LastLogonDate | ConvertTo-Html -Property "samAcc
ountName","GivenName","Surname","LastLogonDate" -Fragment -PreContent
"<h2>Inactive Accounts</h2>"
$Reportvalues = ConvertTo-HTML -Body "$disabledaccounts
$inactiveaccounts" -Head $HTML
$Reportvalues | Out-File "C:\dormantusers.html"
```

In the above, I am looking for disabled accounts by using `Get-ADUser -Filter {Enabled -eq $false} | select samAccountName,GivenName,Surname`. Then, after that, I am searching for users who have been inactive for the last 30 days by using `Get-ADUser -Filter {LastLogonDate -lt $InactiveDate -and Enabled -eq $true} -Properties LastLogonDate | select samAccountName,GivenName,Surname,LastLogonDate`. Here, I am ignoring disabled accounts as they are already recorded as separate entities. At the end, I am passing the findings to an HTML report called `dormantusers.html`.

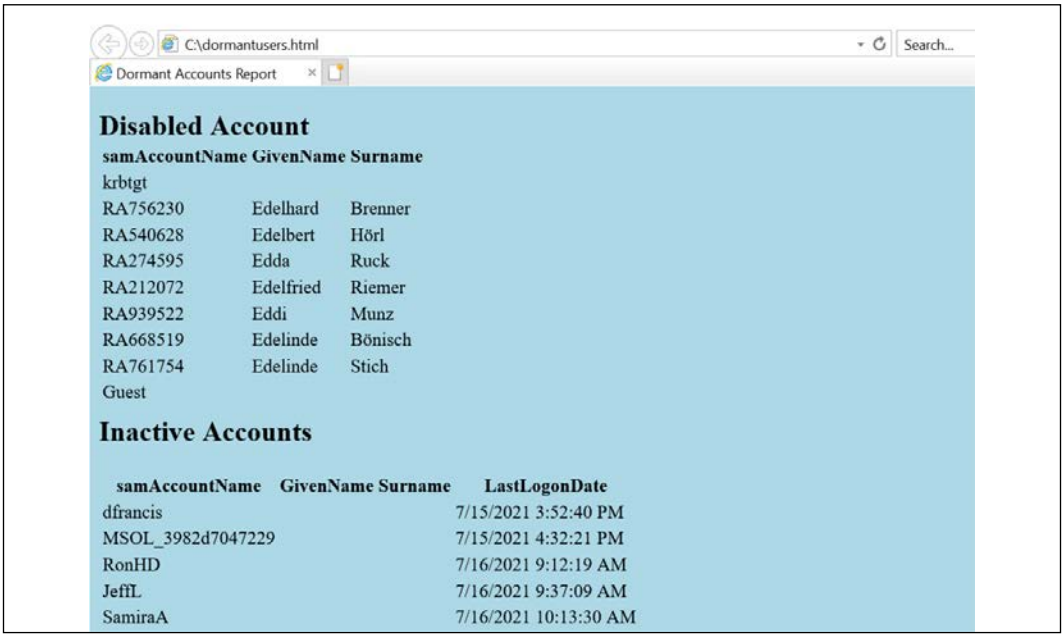


Figure 17.8: Dormant accounts

If required, this data can also be exported into a CSV file. But in these examples, I used the HTML format to show the results in a user-friendly way.

Users with the Password Never Expires setting

In an Active Directory environment, we use password policies to enforce users to follow complexity standards and other best practices related to passwords. Users should use complex passwords and should update their passwords at regular intervals. This is one of the basic requirements of identity protection. However, if the user account has the Password Never Expires setting enabled, the user will not be forced to update the passwords according to the password policy.

We can find Active Directory user accounts that have the Password Never Expires setting enabled by using the following PowerShell commands:

```
Get-ADUser -Filter  {passwordNeverExpires -eq $true -and Enabled -eq
$true } -Properties * | Select samAccountName,GivenName,Surname
```

In the preceding command, I am looking for the `passwordNeverExpires` attribute value and if it's set to true, it means the setting is enabled. At the same time, I also checked if the user account is active.

Figure 17.9: Users with the Password Never Expires setting

In the next section of this chapter, we are going to look into managing Active Directory objects in a hybrid environment.

Azure Active Directory PowerShell

Similar to on-prem Active Directory, we also can use PowerShell to manage Azure Active Directory.

Let's see why we should use PowerShell to manage Azure Active Directory:

- **Early bird access to features**: Microsoft keeps releasing new features, bug fixes, updates, and feature enhancements more frequently to Azure AD services than on-prem Active Directory.

  Microsoft releases new features to the public in two stages. In the first stage, they are released as a preview version. This is not recommended for use in production, but IT professionals can use them for testing and provide feedback to Microsoft. At this stage, the features can have many updates and, most of the time, it will take some time to update the GUI accordingly. Some of these changes will not be available on the GUI until general release. But if we are using PowerShell, we do not have to wait. We can have early access to features as soon as they are released.

- **Faster response**: The Azure Active Directory portal has many different windows, wizards, and forms to configure and manage users, groups, roles, and associated features. The GUI makes it easy to do things, but it takes time. As an example, if you add a user account using the Azure AD portal, you have to go to four sub-windows at least. But PowerShell allows us to do it using one window and a few lines of commands.
- **Granular control**: The Azure AD portal visualizes the data and configuration of the service using different windows. However, it may not always show what we want. As an example, let's assume we are looking for a specific value in two user accounts. If we use the GUI, we need to go to a few different windows to gather this information. But using a PowerShell command or script, we will be able to gather the same information in one window. This is really helpful when troubleshooting.
- **Microsoft Graph integration**: Microsoft Graph provides a unified programmability model to access a vast amount of data in Microsoft 365, Azure Active Directory, Enterprise Mobility Suite, Windows 10, and so on. As part of it, the Azure AD PowerShell for Graph module allows you to retrieve data, update directory configurations, add/update/remove objects, and configure features via Microsoft Graph.

In this chapter, I will be using the Azure Active Directory PowerShell for Graph module to manage an Azure AD hybrid environment.

Installation

The Azure Active Directory PowerShell for Graph module comes in two versions. The public preview version is the most recent, but it is not recommended for use in production.

The installation steps for this version can be found at `https://bit.ly/3HR3EpU`.

The general availability version is the stable, recommended version for production environments. It can be installed on any computer that runs Windows Server 2008 R2 or above with the latest updates. Microsoft .NET Framework 4.5 or above is also required.

Once the prerequisites are in place, perform the following steps:

1. Log in to the computer you have selected for the Azure Active Directory PowerShell for Graph module.
2. Launch the PowerShell console as an administrator.
3. Run the `Install-Module -Name AzureAD` command. Answer `Yes` if it is a required repository update:

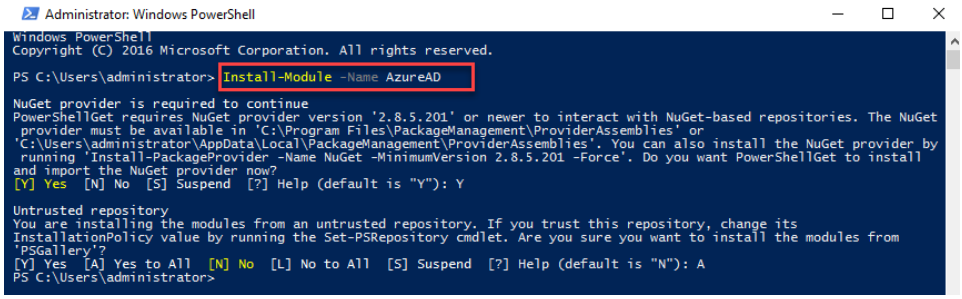


Figure 17.10: Install AzureAD PowerShell module

4. After installation, we can verify the module installation using `Get-Module AzureAD`.
5. After successfully installing the module, run `Connect-AzureAD` to initiate a connection to the Azure AD tenant.
6. Then, it will prompt you with a login window. Use Azure AD global administrator account details to connect.

Now we have the Azure Active Directory PowerShell for Graph module installed. Let's see how we can manage an Azure AD hybrid environment using this module.

Azure AD and MSOL modules are not supported in PowerShell 7.x. Therefore, here, I am using default Windows PowerShell running on Windows 10.

General commands

We can start by listing all the available commands under the Azure AD module, which can be done by using the following:

```
Get-Command -module AzureAD
```

We can view the full syntax for a command by using the `Get-Help` command. As an example, we can view the full syntax for the `Get-AzureADUser` command using the following:

```
Get-Help Get-AzureADUser
```

We can verify the status of Azure AD domains using the following command:

```
Get-AzureADDomain | fl
```

The preceding command helps to identify the domain verification status by referring to the value of the `IsVerified` attribute.

If you are using a custom domain in Azure AD, we need to verify ownership of the domain using DNS records. If it is not verified, we can retrieve the required DNS records by using the following command:

```
Get-AzureADDomainVerificationDnsRecord -Name M365x562652.onmicrosoft.
com | fl
```

In the preceding example, `M365x562652.onmicrosoft.com` represents the domain name:

```
PS C:\Users\administrator> Get-AzureADDomainVerificationDnsRecord -Name M365x562652.onmicrosoft.com | fl


DnsRecordId       : aceff52c-06a5-447f-ac5f-256ad243cc5c
IsOptional        : False
Label             : M365x562652.onmicrosoft.com
RecordType        : Txt
SupportedService  : Email
Ttl               : 3600
Text              : MS=

DnsRecordId       : 5fbde38c-0865-497f-82b1-126f596bcee9
IsOptional        : False
Label             : M365x562652.onmicrosoft.com
RecordType        : Mx
SupportedService  : Email
Ttl               : 3600
MailExchange      : .msv1.invalid
Preference        : 32767
```

Figure 17.11: Required DNS records for domain verifications

We can view the details of the Azure AD tenant by using the following:

```
Get-AzureADTenantDetail | fl
```

In a hybrid environment, the health of the on-prem AD sync is crucial. We can view the time of the last directory sync by using the following command:

```
Get-AzureADTenantDetail | select CompanyLastDirSyncTime
```

Managing users

We can view the user account details for a known account using the following:

```
Get-AzureADUser -ObjectId AdeleV@M365x562652.OnMicrosoft.com | fl
```

In the preceding command, `AdeleV@M365x562652.OnMicrosoft.com` represents the UPN of the user.

We also can use user attributes to find user account details:

```
Get-AzureADUser -Filter "startswith(GivenName,'Adele')"
```

The preceding command will filter Azure AD users with `GivenName` as `Adele`.

We can also filter users based on a specific attribute value:

```
Get-AzureADUser -Filter "GivenName eq 'Adele'"
```

The preceding command will search for the exact user with the given name value `Adele`.

In my demo environment, I'd like to see a list of disabled accounts. I can do this using the following command:

```
Get-AzureADUser -All $true -Filter 'accountEnabled eq false'
```

We can modify the output of the filtered data further:

```
Get-AzureADUser -All $true -Filter 'accountEnabled eq false' | select
DisplayName,UserPrincipalName,Department
```

The preceding command will display the value of the `DisplayName`, `UserPrincipalName`, and `Department` attributes of the filtered accounts.

In a hybrid environment, we can filter accounts that are synced from on-prem AD by using the following:

```
Get-AzureADUser -All $true -Filter 'DirSyncEnabled eq true'
```

In the preceding command, the value of the `DirSyncEnabled` attribute defines whether it's a cloud-only account or a synced account.

We also can check the last sync value for the synced accounts:

```
Get-AzureADUser  -All $true -Filter 'DirSyncEnabled eq true' | select
DisplayName,UserPrincipalName,LastDirSyncTime
```

In the preceding command, the `LastDirSyncTime` value defines the last sync time of the object.

We can also export the output to a CSV file using the `Export-CSV` command:

```
Get-AzureADUser  -All $true -Filter 'DirSyncEnabled eq true' | select
DisplayName,UserPrincipalName,LastDirSyncTime | Export-CSV -Path .\
syncaccount.csv
```

The `ImmutableID` value of a user account is used to map an Azure AD user object to an on-prem user object. `ImmutableID` does have a relationship with on-prem user accounts' `ObjectGUID`. We can use this to identify cloud-only users. If it is a cloud-only user, the `ImmutableID` value should be `null`:

```
Get-AzureADUser -All $true | where-Object {$_.ImmutableId -eq $null}
```

The preceding command returns a list of all the cloud-only accounts. We can export the required attribute values to CSV by using the following:

```
Get-AzureADUser -All $true | where-Object {$_.ImmutableId -eq $null} |
select DisplayName,UserPrincipalName | Export-CSV -Path .\cloudaccount.
csv
```

Another important thing related to accounts is licences. If we are going to use Azure AD's premium features, we need to have relevant licenses assigned. By default, a user only has Azure AD free version features.

To view licenses associated with a user account, we can use the following command:

```
Get-AzureADUserLicenseDetail -ObjectId MeganB@M365x562652.OnMicrosoft.
com | fl
```

The preceding command will return the licenses associated with the user `MeganB@M365x562652.OnMicrosoft.com`.

We also can view the subscribed SKUs using the following command:

```
Get-AzureADSubscribedSku | fl
```

The preceding command lists all the details about licenses that are associated with the tenant. However, we only need to know how many licenses have been used and how many licenses are available. We can do this using the following command:

```
Get-AzureADSubscribedSku | select SkuPartNumber,ConsumedUnits
-ExpandProperty PrepaidUnits
```

In the preceding example, the `SkuPartNumber` value represents the license part number. The value of the enabled field represents the number of purchased licenses. `ConsumedUnits` represents the number of consumed licenses.

Let's move on and see how we can assign a new license to a user.

In my environment, I have a user who synced from on-prem Azure AD who doesn't have a license assigned:

```
Get-AzureADUserLicenseDetail -ObjectId ADJellison@M365x562652.
onmicrosoft.com | fl
```

The following screenshot displays the output of the preceding command:

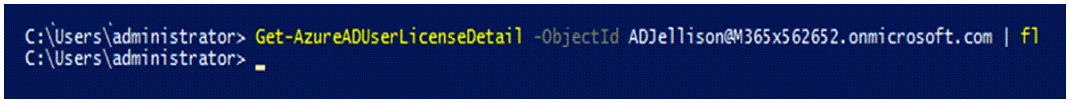


Figure 17.12: Check Azure AD license assignment

As a first step, let's create objects to use in the license assignment process:

```
$newlicence = New-Object -TypeName Microsoft.Open.AzureAD.Model.
AssignedLicense
$newlicenceadd = New-Object -TypeName Microsoft.Open.AzureAD.Model.
AssignedLicenses
```

Then, we need to find the `SkuId` of the licenses.

I am going to assign the `ENTERPRISEPREMIUM` license to the user:

```
$newlicence.SkuId = (Get-AzureADSubscribedSku | Where-Object -Property
SkuPartNumber -Value "ENTERPRISEPREMIUM" -EQ).SkuId
```

Then, we need to assign the licenses to the object:

```
$newlicenceadd.AddLicenses = $newlicence
```

Now, we can go ahead and assign the license to the user:

```
Set-AzureADUserLicense -ObjectId "ADJellison@M365x562652.onmicrosoft.
com" -AssignedLicenses $newlicenceadd
```

The preceding command assigns `ENTERPRISEPREMIUM` licenses to the user `ADJellison@M365x562652.onmicrosoft.com`:

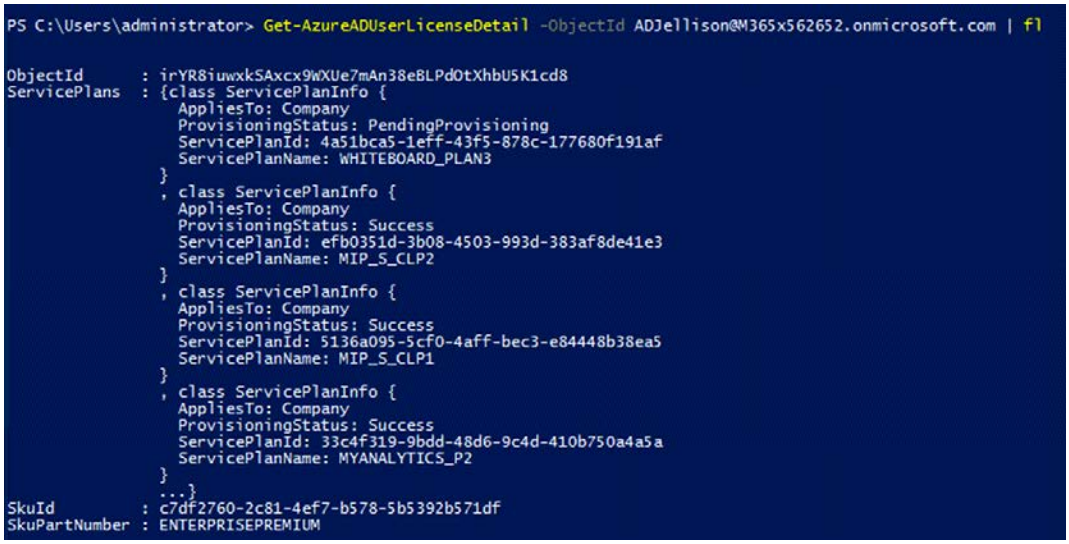


Figure 17.13: Assign license to user

It is a must to set the `UsageLocation` value for users who sync from on-prem AD before assigning licenses. We can do this using `Set-AzureADUser -ObjectId ADJellison@M365x562652.onmicrosoft.com -UsageLocation "US"`.

We can remove the licenses assigned using the following command:

```
$licenseB = New-Object -TypeName Microsoft.Open.AzureAD.Model.
AssignedLicenses
$licenseB.RemoveLicenses =  (Get-AzureADSubscribedSku | Where-Object
{$_.SkuPartNumber -eq 'ENTERPRISEPREMIUM'}).SkuId
Set-AzureADUserLicense -ObjectId "ADJellison@M365x562652.onmicrosoft.
com" -AssignedLicenses $licenseB
```

Using the preceding commands, I have created a script to do the following:

- Search for users who synced from on-prem AD
- Of those users, select the users who don't have Azure AD licenses assigned
- Set the `UsageLocation` value for selected users
- Assign Azure AD licenses to selected users

```
#######Script to Assign Licences to Synced Users from On-Permises
AD#############
Import-Module AzureAD
Connect-AzureAD
###Filter Synced Users who doesn't have licence assigned#######
$ADusers = Get-AzureADUser -All $true -Filter 'DirSyncEnabled eq true'
$notlicenced = Get-AzureADUser -All $true | Where-Object {$ADusers.
AssignedLicenses -ne $null} | select ObjectId | Out-File -FilePath C:\
users.txt
#####Set UsageLocation value to sync users#########
(Get-Content "C:\users.txt" | select-object -skip 3) | ForEach { Set-
AzureADUser -ObjectId $_ -UsageLocation "US" }
#####Set User Licecnes############
$newlicence = New-Object -TypeName Microsoft.Open.AzureAD.Model.
AssignedLicense
$newlicenceadd = New-Object -TypeName Microsoft.Open.AzureAD.Model.
AssignedLicenses
$newlicence.SkuId = (Get-
AzureADSubscribedSku | Where-Object -Property SkuPartNumber -Value
"ENTERPRISEPREMIUM" -EQ).SkuId
$newlicenceadd.AddLicenses = $newlicence
(Get-Content "C:\users.txt" | select-object -skip 3) | ForEach { Set-
AzureADUserLicense -ObjectId $_ -AssignedLicenses $newlicenceadd }
```

In a hybrid environment, users are mainly created through on-prem Active Directory, but there are occasions when we need to add cloud-only accounts. This is mainly for cloud management tasks.

We can create a new user by using the following command:

```
$Userpassword = New-Object -TypeName Microsoft.Open.AzureAD.Model.
PasswordProfile
$Userpassword.Password = "London@1234"
New-AzureADUser -DisplayName "Andrew Xavier" -PasswordProfile
$Userpassword -UserPrincipalName "Andrew.Xavier@M365x562652.
onmicrosoft.com" -AccountEnabled $true -MailNickName "AndrewXavier"
```

In the preceding command, `-PasswordProfile` is used to define the password profile for the new user account. `-MailNickName` defines the value for the user's mail nickname. In the preceding example, add a new user account, `Andrew.Xavier@M365x562652.onmicrosoft.com`, with the password `London@1234`.

We also can create multiple user accounts using CSV files. In the following example, I am using a CSV file to create users. The CSV file contains the following:

```
UserPrincipalName, DisplayName,MailNickName
DishanM@M365x562652.onmicrosoft.com, Dishan Melroy,DishanMel
JackM@M365x562652.onmicrosoft.com,Jack May,JackMay
RicahrdP@M365x562652.onmicrosoft.com,Richard Parker,RichardPar
```

Then, I can create these new users using the following:

```
$Userpassword = New-Object -TypeName Microsoft.Open.AzureAD.Model.
PasswordProfile
$Userpassword.Password = "London@1234"
Import-Csv -Path C:\newuser.csv | foreach {New-AzureADUser
-UserPrincipalName $_.UserPrincipalName -DisplayName $_.DisplayName
-MailNickName $_.MailNickName -PasswordProfile $Userpassword
-AccountEnabled $true}
```

By using the preceding commands, I have created a script to do the following:

- Create new user accounts using a CSV file
- Set `UsageLocation` for new user accounts
- Assign `ENTERPRISEPREMIUM` licenses to users

```
########A Script to create new users and assign Azure AD
licences#######
Import-Module AzureAD
Connect-AzureAD
###########Create New Users using CSV ##################
$Userpassword = New-Object -TypeName Microsoft.Open.AzureAD.Model.
PasswordProfile
$Userpassword.Password = "London@1234"
Import-Csv -Path C:\newuser.csv | foreach {New-AzureADUser
-UserPrincipalName $_.UserPrincipalName -DisplayName $_.DisplayName
-MailNickName $_.MailNickName -PasswordProfile $Userpassword
-UsageLocation "US" -AccountEnabled $true} | select ObjectId | Out-File
-FilePath C:\users.txt
###########Assign Licences#################
$newlicence = New-Object -TypeName Microsoft.Open.AzureAD.Model.
AssignedLicense
$newlicenceadd = New-Object -TypeName Microsoft.Open.AzureAD.Model.
AssignedLicenses
$newlicence.SkuId = (Get-AzureADSubscribedSku | Where-Object -Property
```

```
SkuPartNumber -Value "ENTERPRISEPREMIUM" -EQ).SkuId
$newlicenceadd.AddLicenses = $newlicence
(Get-Content "C:\users.txt" | select-object -skip 3) | ForEach { Set-
AzureADUserLicense -ObjectId $_ -AssignedLicenses $newlicenceadd }
```

To remove an Azure AD user, we can use the following:

```
Remove-AzureADUser -ObjectId "JDAllen@M365x562652.onmicrosoft.com"
```

We can combine it with a user search using the following command:

```
Get-AzureADUser -Filter "startswith(DisplayName,'Dishan')" | Remove-
AzureADUser
```

The preceding command will search for user accounts that have a `DisplayName` that starts with `Dishan`. If there are any, the second part of the command will remove them.

Managing groups

Azure AD groups also work similarly to on-prem AD groups. They can be used to manage permissions in an effective manner. In a hybrid environment, there will be cloud-only groups as well as synced groups from the on-prem AD environment. In this section, we are going to look into group management using the Azure Active Directory PowerShell for Graph module.

Let's start with listing groups. We can search for a group using the following command:

```
Get-AzureADGroup -SearchString "sg"
```

In the preceding command, `SearchString` is used to define the search criteria. The preceding example will list any groups containing `sg` in the `DisplayName` field:

```
PS C:\Users\administrator> Get-AzureADGroup -SearchString "sg"

ObjectId                             DisplayName            Description
--------                             -----------            -----------
93291438-be19-472e-a1d6-9b178b7ac619 sg-Engineering         All engineering personnel
2a11d5ee-8383-44d1-9fbd-85cb4dcc2d5a sg-Executive           All executives
38bd48e7-e37c-48c9-bae4-7f00949529a0 sg-Finance             All finance personnel
c93737dd-0e77-4325-8d5e-6524d8baedff sg-HR                  All HR personnel
c00fc3df-a395-48fb-a620-05d64384fa3e sg-IT                  All IT personnel
f7ca523a-3a17-4755-aea9-5c723e8d9c6a sg-Legal               All legal executives
73115583-7ce1-4890-8b96-40bc257e0186 sg-Operations          All operations personnel
0636227c-0a03-4584-a72d-52e5479c2aad sg-Retail              All retail Users
6a232560-7a59-413c-8f2b-60ecda3b21cb sg-Sales and Marketing All marketing personnel
```

Figure 17.14: Search for groups

In the search result, we can see the `ObjectId` for the group. Once we know the `ObjectId`, we can see the details of the group using the following command:

```
Get-AzureADGroup -ObjectId 93291438-be19-472e-a1d6-9b178b7ac619 | fl
```

In a hybrid environment, there will be security groups that have synced from the on-premises Active Directory. We can filter these groups using the following:

```
Get-AzureADGroup -Filter 'DirSyncEnabled eq true' | select ObjectId,Dis
playName,LastDirSyncTime
```

In the preceding example, the `LastDirSyncTime` column displays the last successful sync time of the group.

We can filter cloud-only groups using the following command:

```
Get-AzureADGroup -All $true | where-Object {$_.
OnPremisesSecurityIdentifier -eq $null}
```

In the preceding command, we are using the `OnPremisesSecurityIdentifier` attribute to filter the groups. This attribute only has value if it is synced from on-premises AD.

We can view group memberships by using the following:

```
Get-AzureADGroupMember -ObjectId 2a11d5ee-8383-44d1-9fbd-85cb4dcc2d5a
```

In the preceding command, we are using `ObjectId` to uniquely identify the group.

We can add members to the group using the `Add-AzureADGroupMember` cmdlet:

```
Add-AzureADGroupMember -ObjectId 2a11d5ee-8383-44d1-9fbd-85cb4dcc2d5a
-RefObjectId a6aeced9-909e-4684-8712-d0f242451338
```

In the preceding command, the `ObjectId` value represents the group, and the `RefObjectId` value represents the user.

We can remove a member from the group by using the following command:

```
Remove-AzureADGroupMember -ObjectId 2a11d5ee-8383-44d1-9fbd-
85cb4dcc2d5a -MemberId a6aeced9-909e-4684-8712-d0f242451338
```

In the preceding command, the `ObjectId` value represents the group, and the `MemberId` value represents the user's `ObjectId`.

We can also combine the `Add-AzureADGroupMember` cmdlet with the `Get-AzureADUser` cmdlet to add bulk users to a group.

In the following script, I used the `Get-AzureADUser` cmdlet to search for users in `Marketing Department`, and then used `Add-AzureADGroupMember` to add those users to `Sales Group` as members:

```
#######Script to Add Multiple users to Security Group##############
Import-Module AzureAD
Connect-AzureAD
##### Search for users in Marketing Department ##########
Get-AzureADUser -All $true -Filter "Department eq 'Marketing'" | select
ObjectId | Out-File -FilePath C:\salesusers.txt
#####Add Users to Sales Group#########
(Get-Content "C:\salesusers.txt" | select-object -skip 3) | ForEach {
Add-AzureADGroupMember -ObjectId f9f51d29-e093-4e57-ad79-2fc5ae3517db
-RefObjectId $_ }
```

In a hybrid environment, security groups are mainly synced from on-prem AD. But there can be requirements for cloud-only groups as well. We can create a cloud-only group by using the following:

```
New-AzureADGroup -DisplayName "REBELADMIN Sales Team" -MailEnabled
$false -MailNickName "salesteam" -SecurityEnabled $true
```

The following screenshot displays the output of the preceding command:

```
PS C:\Users\administrator> New-AzureADGroup -DisplayName "REBELADMIN Sales Team" -MailEnabled $false -MailNickName
steam" -SecurityEnabled $true

ObjectId                             DisplayName          Description
--------                             -----------          -----------
7592b555-343d-4f73-a6f1-2270d7cf014f REBELADMIN Sales Team
```

Figure 17.15: Create a cloud-only group

The preceding command creates a security group called `REBELADMIN Sales Team`. This group is not a mail-enabled group.

We can remove an Azure AD group using the following command:

```
Remove-AzureADGroup -ObjectId 7592b555-343d-4f73-a6f1-2270d7cf014f
```

In the preceding command, the `ObjectId` value defines the group.

Apart from security groups, Azure AD also has predefined administrative roles, which can be used to assign access permissions to Azure AD and other cloud services. There are more than 35 predefined administrative roles. Each role has its own set of permissions. More details about these roles can be found at `https://bit.ly/3r5FMcs`.

We can list all the administrative roles using the following:

```
Get-AzureADDirectoryRoleTemplate
```

By default, only a few administrative roles are enabled. We can list these roles using the following:

```
Get-AzureADDirectoryRole
```

Here, the company administrator directory role represents the Azure AD global administrators.

We can enable the administrative role using the following:

```
Enable-AzureADDirectoryRole -RoleTemplateId e6d1a23a-da11-4be4-9570-
befc86d067a7
```

In the preceding command, the `RoleTemplateId` value represents the administrative role.

We can assign the administrative role to a user by using the following command:

```
Add-AzureADDirectoryRoleMember -ObjectId b63c1671-625a-4a80-8bae-
6487423909ca -RefObjectId 581c7265-c8cc-493b-9686-771b2f10a77e
```

In the preceding command, the `ObjectId` value represents the administrative role. `RefObjectId` is the object ID value of the user.

We can list members of the administrative role using the following:

```
Get-AzureADDirectoryRoleMember -ObjectId 36b9ac02-9dfc-402a-8d44-
ba2d8995dc06
```

In the preceding command, `ObjectId` represents the administrative role.

We can remove a member from the role using the following command:

```
Remove-AzureADDirectoryRoleMember -ObjectId 36b9ac02-9dfc-402a-8d44-
ba2d8995dc06 -MemberId 165ebcb7-f07d-42d2-a52e-90f44e71e4a1
```

In the preceding command, `MemberId` is equal to the user's object ID value.

This marks the end of this section. There are lots of cmdlets that can still be used to manage Azure AD, but here I explained the cmdlets that will be required for day-to-day operations.

Microsoft Graph

Microsoft Graph is like a gateway that allows users to access enormous amounts of data and collect information from:

1. Microsoft 365 core services (for example, Office 365, Microsoft Search, OneDrive, SharePoint)
2. Identity and Security Services (for example, Azure AD, Defender 365, Endpoint Manager)
3. Windows 10 services

Microsoft Graph connects to the above services by using REST APIs and client libraries to retrieve required data.

We can use three methods to interact with Microsoft Graph data:

1. The Microsoft Graph API endpoint (`https://bit.ly/3DSbZHF`) can be used to access data and information collected from various Microsoft services. This data can be processed and present in the way an organization/individual requires. Also, this data can be used to develop rich applications/services.
2. Microsoft Graph connectors help to bring third-party application/service data to Microsoft Search so all company data can be searched from one location. Some of these connectors are built by Microsoft and others are built by partners. You can access the connectors gallery by using `https://bit.ly/3FMBaft`.
3. Microsoft Graph Data Connect allows us to access Microsoft Graph data at scale. This gives engineers a granular level of control over Microsoft Graph data. Microsoft Graph Data Connect also provides a unique set of tools that can be used to build intelligent applications.

Based on the requirements, we can use one or more of the above methods to access Microsoft Graph data.

In this section, I am going to demonstrate how we can use Microsoft Graph to access Azure AD data.

Microsoft Graph Explorer

Graph Explorer is a Microsoft-developed tool that allows you to make Microsoft Graph REST API requests. This tool can be accessed using `https://bit.ly/3l6EF8m`.

The first thing we need to do on the page is to log in. Then we need to grant permissions to Microsoft Graph to access data. To do that, go to the **Modify permissions** (**Preview**) tab and give consent to grant relevant permissions.

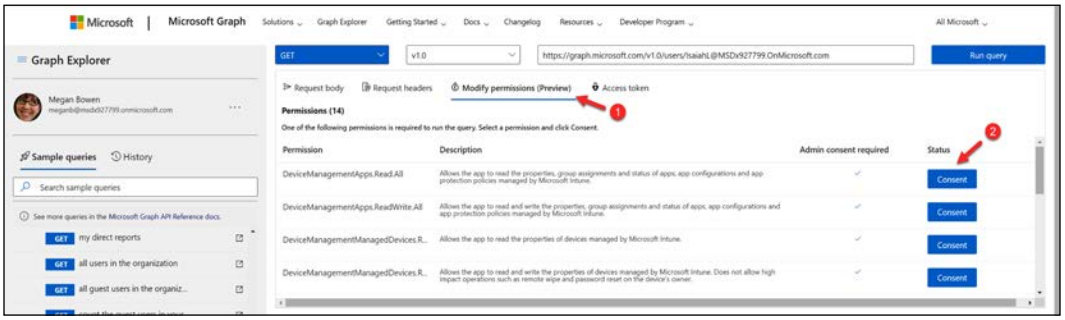


Figure 17.16: Grant permissions to user

After relevant permissions are in place, we can go ahead and query data using Microsoft Graph Explorer. Let's go ahead and start with a user query.

In this example, I would like to view the account details of the `IsaiahL@MSDx927799.OnMicrosoft.com` user. To do that, I am going to use the GET HTTP method and the `https://graph.microsoft.com/v1.0/users/IsaiahL@rebeladmin.OnMicrosoft.com` query.

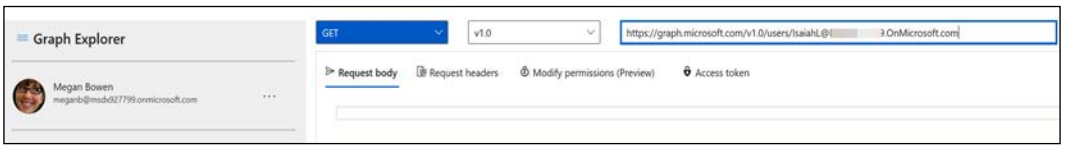


Figure 17.17: User query

Then Microsoft Graph responds to the query with the following data:


```
{
    "@odata.context": "https://graph.microsoft.com/v1.0/$metadata#users/$entity",
    "businessPhones": [
        "+1 918 555 0101"
    ],
    "displayName": "Isaiah Langer",
    "givenName": "Isaiah",
    "jobTitle": "Sales Rep",
    "mail": "IsaiahL@                .OnMicrosoft.com",
    "mobilePhone": null,
    "officeLocation": "20/1101",
    "preferredLanguage": null,
    "surname": "Langer",
    "userPrincipalName": "IsaiahL@                .OnMicrosoft.com",
    "id": "0bb2ceaa-244f-4ea9-aa71-                "
}
```


Figure 17.18: Output of user query

According to the above data, the user has the `jobTitle` attribute value as `Sales Rep`. This user has been promoted recently and I need to change the title to `Sales Manager`. To do that, we need to use the **PATCH** HTTP method:

`PATCH https://graph.microsoft.com/v1.0/users/IsaiahL@rebeladmin.OnMicrosoft.com`

Before I run the query under the request body, I define the new values I need to apply:

```
{
    "jobTitle": "Sales Manager"
}
```

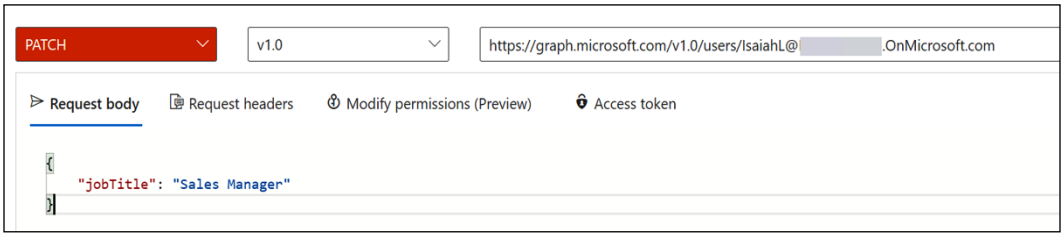


Figure 17.19: Update attribute value

After I run the query successfully, I can see the relevant attribute has a new value.

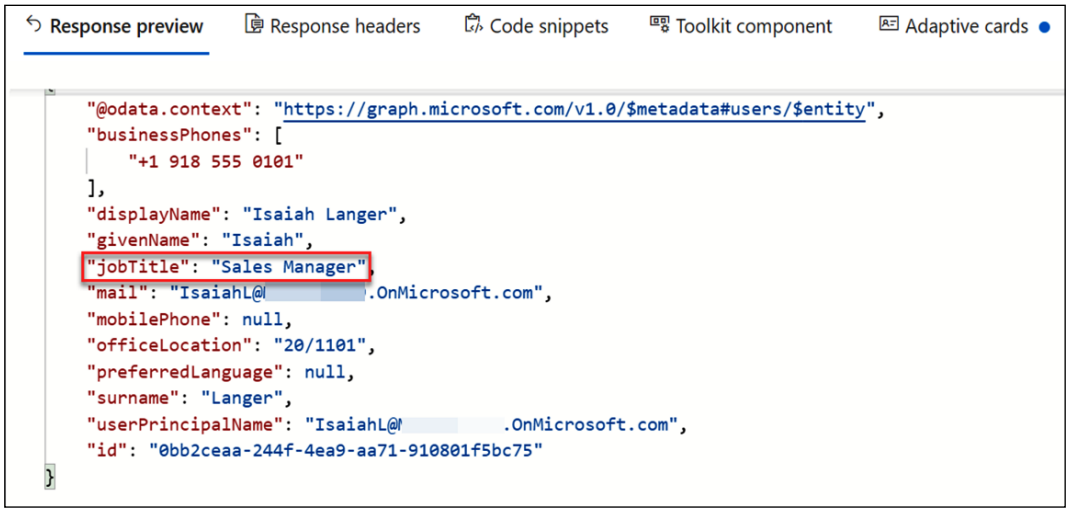


Figure 17.20: Confirm new attribute value

As the next step, I would like to list my domains under Azure AD. To do that, I can use:

```
GET https://graph.microsoft.com/v1.0/domains
```

It returns the details about all the domains under the current subscription. If we need to view data related to a particular domain, we can do it by using:

```
GET https://graph.microsoft.com/v1.0/domains/rebeladmin.OnMicrosoft.com
```

In the preceding command, `rebeladmin.OnMicrosoft.com` is the FQDN.

Figure 17.21: Domain information

As we can see, it provides rich data about the domain. Microsoft Graph Explorer can also be used to create, modify, and delete data by using the HTTP methods POST and DELETE. As an example, if the domain name is not verified, I can force the verification of the domain by using:

```
POST https://graph.microsoft.com/v1.0/domains/rebeladmin.onmicrosoft.
com/verify
```

Next, I would like to list all the groups in Azure AD. For that, I can use:

```
GET https://graph.microsoft.com/v1.0/groups
```

The above command lists down all the groups in the directory with attribute values, but I am more interested in finding out the `id` and `displayName` values of the groups.

So, I am going to modify the query and only select `id` and `displayName` values:

```
GET https://graph.microsoft.com/v1.0/groups?$select=id,displayName
```

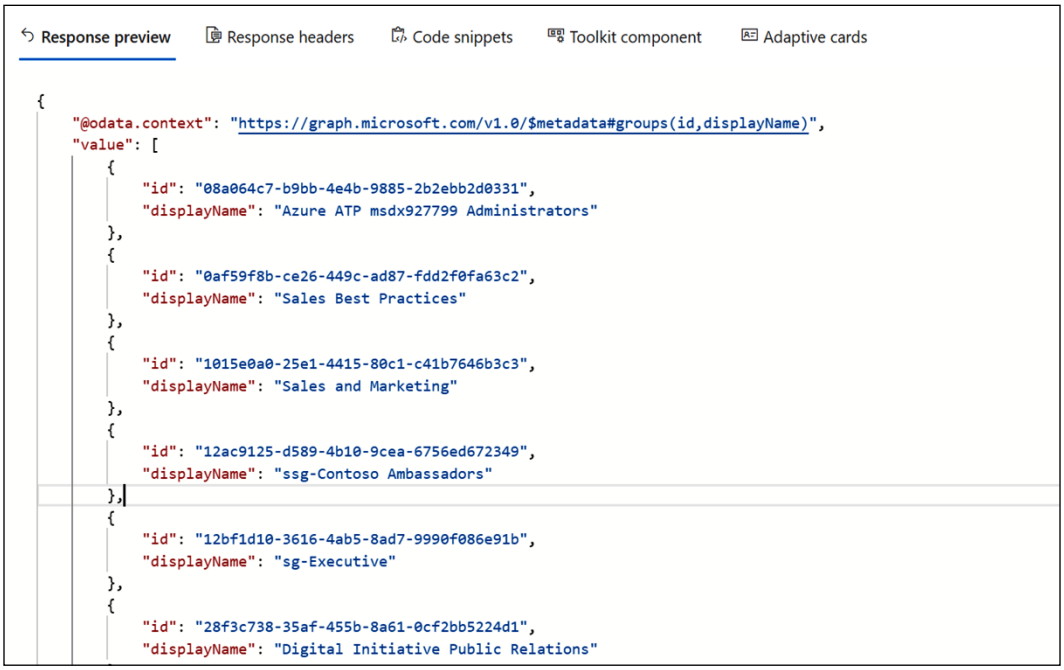


Figure 17.22: List groups

But this is still a long list. I want to search for groups that start with the letters sg. To do that, we can use:

```
GET https://graph.microsoft.com/v1.0/groups?$filter=startswith(
displayName, 'sg')&$select=id,displayName
```

From the group list, I am more interested in the `sg-Sales and Marketing`. Let's take note of the group `id` value for future use.

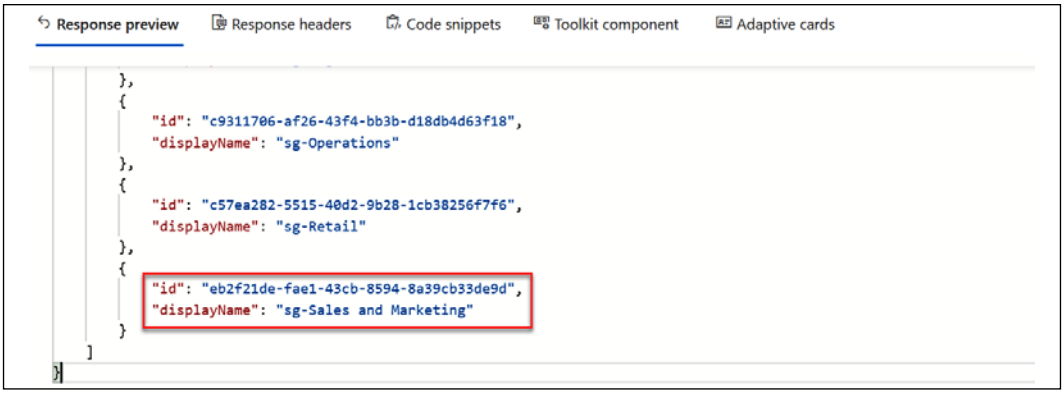


Figure 17.23: List of groups contain "sg"

As the next step, I would like to get a list of members of `sg-Sales and Marketing`. We can do this by using:

```
GET https://graph.microsoft.com/v1.0/groups/{eb2f21de-fae1-43cb-8594-
8a39cb33de9d}/members?$count=true
```

In the above command, `eb2f21de-fae1-43cb-8594-8a39cb33de9d` is the group ID of the `sg-Sales and Marketing` group.

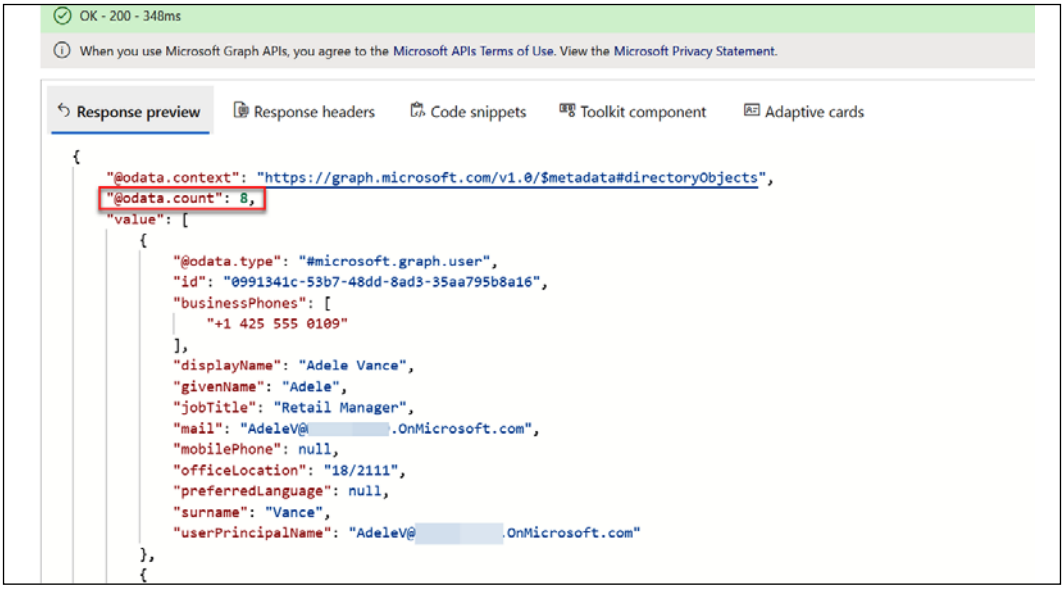


Figure 17.24: List members of a group

As we can see above, the query has a response with a list of all the users. In response, the `@odata.count` value represents the number of members in the group.

I have a new user called Megan with the object ID `086ba971-ecc2-49a5-a063-6e2d4fba9e9b`. I would like to add this user to the `sg-Sales and Marketing` group. This task includes two steps.

As the first step, we need to paste the following command in the query window:

```
POST https://graph.microsoft.com/v1.0/groups/{eb2f21de-fae1-43cb-8594-8a39cb33de9d}/members/$ref
```

In the above, `eb2f21de-fae1-43cb-8594-8a39cb33de9d` is the group ID value.

Then, under the `Request body` section, I use the following command:

```
{
    "@odata.id": "https://graph.microsoft.com/v1.0/directoryObjects/{086ba971-ecc2-49a5-a063-6e2d4fba9e9b}"
}
```

This is the reference for the new member object ID. In the above, `086ba971-ecc2-49a5-a063-6e2d4fba9e9b` is the object ID of the user Megan.

After that, we can run the query. If it is successful, we should get a `No Content - 204` response.

After that, I rerun the command to list down members of `sg-Sales and Marketing` and now I can see the new user as a member.

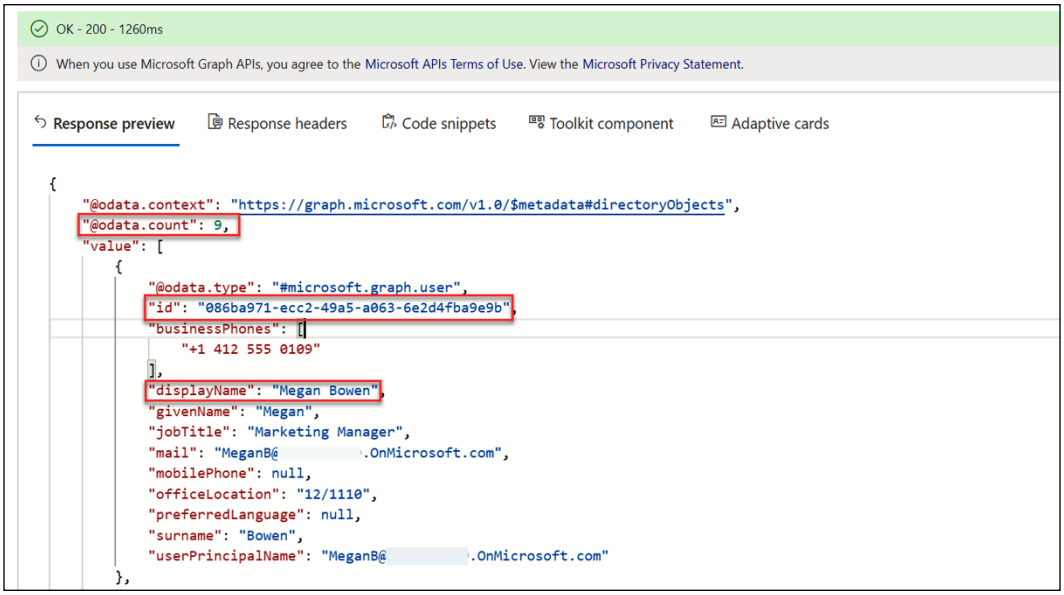


Figure 17.25: Add user to a group

We also can delete a member from a group by using the **DELETE** HTTP method. If we need to remove Megan from the group, we can use:

```
DELETE https://graph.microsoft.com/v1.0/groups/{eb2f21de-fae1-43cb-
8594-8a39cb33de9d}/members/{086ba971-ecc2-49a5-a063-6e2d4fba9e9b}/$ref
```

In the preceding command, `eb2f21de-fae1-43cb-8594-8a39cb33de9d` is the object ID of the group and `086ba971-ecc2-49a5-a063-6e2d4fba9e9b` is the object ID of the user Megan.

In this section of the chapter, I have used several examples to show how we can use Microsoft Graph with Azure AD. This is a vast topic but I believe, now, you have an idea of how Microsoft Graph works. For more information, please visit `https://bit.ly/3HS0MJq`.

Summary

PowerShell has become the most powerful script language for Windows systems. PowerShell is very useful for systems management but is also an incredibly powerful tool for managing Active Directory infrastructures. Throughout the book, I have used PowerShell for Active Directory configuration and management.

Furthermore, I have shared different commands and scripts that can be used to manage an Active Directory environment efficiently.

Toward the end of the chapter, you learned how to manage Azure AD using the Azure Active Directory PowerShell for Graph module. We also looked into Microsoft Graph and learned how to use it to manage Azure AD. In the next chapter, we will look at Azure AD closely and learn how to manage identities in a hybrid environment.

18

Hybrid Identity

Back in 2006, I was working with a large Canadian managed-hosting service provider. At that time, there was a huge demand for dedicated server hosting and colocation services. Hardware, bandwidth, and management all came at a high cost. However, things started to change with the rise of virtualization: it was able to bring the hosting costs down massively. I still remember that there were all sorts of discussions at the time about the pros and cons of virtualization. As with any technology, in the beginning, there were issues, but virtualization technologies developed rapidly and brought businesses to a point that they can't look away from.

For us, it was the same: business-wise, we were safe with dedicated server hosting. We were making good profits. But with virtualization, customers were able to bring racks of dedicated servers into a few hypervisor hosts. Then, businesses in the hosting field started to find new ways of making money with virtualized technologies. This was the beginning of the cloud era. However, what I want to emphasize is similar to the technological shift from dedicated servers to virtualization: the majority of today's infrastructures are going through a very interesting phase of moving workloads from on-prem infrastructure to the public cloud. When Microsoft Azure was released, the technology world was deluged with all sorts of discussions again. Most of the points were related to data security, compliance, reliability, and cost.

Over the past few years, Microsoft has been addressing all those concerns and challenges, and it came to the point where organizations could not stay away from it anymore for the following reasons:

- The cloud pricing model (only pay for the resources you use) and operational model can bring down long-term infrastructure operation and maintenance costs.

- Software vendors started replacing their products with cloud-based versions and discontinued support for on-prem versions.
- Microsoft products have equivalent cloud versions on-prem, and new features will only be available in the cloud versions. Also, the cloud versions have more frequent updates and bug fixes compared to on-prem versions.
- It removed dependencies (such as network connectivity, VPN, and firewall configuration) for mobile workers and provided seamless access to workloads from anywhere.
- The cloud adopts new technology changes more quickly compared to on-prem infrastructures.
- A robust cloud infrastructure setup provides **High Availability** (**HA**) and scalability for workloads, which may not be possible to achieve on-prem without large investments.

When an organization starts its cloud journey, it's not easy to bring each and every workload to the public cloud at once. There are limitations for applications that still require some workloads to run on-prem. Even though workloads operate from two technologies, the user identities for the organization would stay the same. Azure **Active Directory** (**AD**) helps to extend the on-prem identity infrastructure to the Azure cloud and use the same on-prem identities to authenticate with the application and services, regardless of where they are running from.

In this chapter, we will look at the following topics:

- How to integrate Azure AD with on-prem AD
- Password hash synchronization
- Azure AD Pass-through Authentication
- Azure AD Seamless **Single Sign-On** (**SSO**)
- Azure AD Domain Services
- Configure secure LDAP
- Create replica sets
- Create forest trust between Azure AD and an on-prem AD forest

We are going to start this chapter with the most important topic. We are going to look into extending our on-prem identities to the cloud.

Extending on-prem AD to Azure AD

In *Chapter 1*, *Active Directory Fundamentals*, I explained why it is important to consider extending out identities to the cloud. Throughout this book, we've learned about the features and management of Azure AD in a hybrid environment. Now, it is time to talk about the integration of Azure AD with on-prem AD.

Based on experience, I would like to propose the following steps to consider before proceeding with the implementation task.

1. Evaluating the present business requirements
2. Evaluating an organization's infrastructure road map
3. Evaluating the security requirements
4. Selecting the Azure AD version
5. Deciding on the sign-in method
6. Implementation

The actual configuration process of hybrid identity is quite straightforward. In less than an hour, we can complete the process. But that's not the point. There are some critical decisions businesses have to make before stepping into hybrid identity.

Evaluating the present business requirements

There can be one or many reasons why a business is looking to extend their on-prem AD to Azure AD. The most common reasons can be as follows:

- Use of Software-as-a-Service (SaaS) applications
- Cloud migration
- Features
- Security

Let's go ahead and explore each of the above in detail:

- **Use of Software-as-a-Service (SaaS) applications**: This is one of the most common reasons for an organization to start using the Azure AD hybrid model. With the maturity of the cloud, lots of software vendors started to move into the SaaS market. Most of these solutions are now available through the Microsoft Azure Marketplace.

Due to HA, less management, scalability, and cost, most organizations do not hesitate to move into SaaS applications. Every SaaS application requires authentication to handle access permissions. By extending on-prem AD to Azure AD, users are able to use their existing domain logins to authenticate SaaS applications. I think we all agree that Microsoft Office 365 is the most commonly used SaaS, and most businesses started their cloud journey with Office 365.

- **Cloud migration**: When an organization is moving their workloads to Azure, they will be using one or a few of the following methods:
    - **Rehost**: This is also called the lift-and-shift method. This method doesn't require code changes or architecture changes. It is the easiest method for moving workloads from on-prem to the cloud. As an example, let's assume an organization is using a web application that is hosted on a server running on-prem. If we need to move anything to the cloud, we can simply create a similar VM using Azure **Infrastructure as a Service** (**IaaS**), and migrate the application and its data across. However, if the organization is using on-prem AD, it is more likely that their workloads use domain authentication. Therefore, by introducing an Azure AD hybrid setup, we can add new servers to the same domain and use the same identities to authenticate into services.
    - **Refactor**: This method requires changes to the application design or architecture. With this migration method, applications and other services will get benefits from the cloud SaaS and **Platform-as-a-Service** (**PaaS**) offerings. The migration of on-prem Microsoft Exchange to Office 365 is a good example of this method. Instead of migrating Exchange applications, an organization can migrate into feature-rich Office 365 solutions and start using the cloud's benefits. If the organization needs to maintain the same domain authentication with these new solutions, they have to use Azure AD with on-prem AD integration.
    - **Re-architecture**: This process will provide a modern touch to applications and services. The whole point of this method is to optimize applications and services to support the cloud's scalability, resilience, and HA. As an example, RebelAdmin Corp. is running an online store. They are currently using a few hosted web servers and MS SQL database servers in order to provide HA. When it comes to the cloud, instead of using the same architecture, they can migrate the application to Azure containers or scale sets. MS SQL clusters can also be replaced by an Azure SQL database in order to provide HA and scalability to applications.

This will also give them the opportunity to upgrade the identity element of the application.

As an example, instead of using our AD username and password, by using Azure AD B2C, we can allow consumers to use their already existing social accounts such as Microsoft, Gmail, and Facebook to log in to applications. However, not every application or service will support the re-architecture method. Therefore, organizations will have to use this method along with a rehost or refactor migration method.

- **Rebuild**: This method requires a complete restructuring of applications and services. No applications will be migrated to the cloud from on-prem. This is mainly due to the limitations and drawbacks of applications that cannot be improved by cloud migration; therefore, the only option is to start over and build a solution with modern cloud-native technologies. This also applies to identity management. There are many differences between on-prem AD and Azure AD. Azure AD is not the cloud version of AD. Nothing prevents organizations from putting more pressure on Azure AD capabilities, applications, or other infrastructure components. It isn't easy to completely cut out on-prem AD and start over with Azure AD without having a major impact on the entire identity infrastructure; therefore, it is safer to start with hybrid mode and then plan the cutover.

  As we can see, every cloud migration method has a role to play with identities, and it is easy and convenient to start with Azure AD on-prem integration.

- **Features**: Azure AD is a managed service. It receives feature updates and bug fixes more frequently. Most of these features also support work in hybrid environments; therefore, if an organization is looking to use Azure AD features with minimum infrastructure changes, on-prem AD integration is the place to start.
- **Security**: In Chapter 1, I explained why we need more than Windows AD to protect identities against emerging threats. The Covid-19 pandemic completely changed the way we work and the traditional perimeter defense approach is no longer working. We need tools and services that can help us to embrace a zero-trust approach for security. When we start using Azure AD in a hybrid environment, we can use Microsoft threat intelligence to identify user and sign-in risks. Using these insights, we can use Conditional Access to enforce zero-trust principles. Not only that, but we can also use many different Azure services to improve identity and data protection further.

Most of these solutions require Azure AD to deliver to their full potential. Apart from analyzing the main requirements, we can use the following questions to evaluate the requirements further:

- What is the immediate business requirement for using Azure AD?
- Will this requirement affect everyone in the company? If not, which set of users will this apply to? (As an example, if a company has a group of businesses, which company or business unit will this requirement apply to?)
- How soon should the requirement be addressed?
- Is the business aware of the identity infrastructure changes that are required to achieve the business requirements?
- What is the available budget?
- Is the business aware of the other benefits of using Azure AD, apart from authentication (features such as identity protection, data protection, and auditing)?
- Are the business' legal and compliance requirements satisfied by the required changes?
- Does the business have the required resources (skills) to do the implementation and management? If not, how can the organization get it (training, outsourcing)?
- Was the risk analysis for Azure AD integration done?

The answers to the previous questions will help engineers understand the requirements from a business point of view, as well as deciding on the next course of action before going further with the planning and implementation processes.

Evaluating an organization's infrastructure road map

In the previous section, we talked about why an organization may use an Azure AD hybrid setup. Moving from an on-prem service to a cloud service is a big decision for an organization; therefore, if it is not planned properly from an early stage, it can be hard to change things after implementation. To provide any IT solution, as engineers, we should not only consider the current state of the requirement: we also need to know the upcoming changes or future plans for the company infrastructure as it may have an impact on the solution that we have in mind. This is important as it helps engineers to provide future-proof solutions.

RebelAdmin Corp. has an initial requirement to move from on-prem Exchange services to Office 365. However, RebelAdmin Corp. has already decided to move their application and services completely to the Azure cloud by 2023. So, if an engineer just considers the immediate requirements, they can simply finish the project by integrating the on-prem AD with the Azure AD free version that comes with Office 365; but by knowing the business' plans for cloud migration, they can also consider the following, apart from Office 365 migration:

- Make plans to implement a zero-trust approach to identity and data security
- Use a suitable Azure AD version to use advanced identity and data protection features
- Try to use cloud-only identities for new users
- Start provisioning Azure landing zones and use VMs and applications in the cloud for new business requirements
- Educate users on how to use Azure AD features effectively, such as Azure MFA, password-less authentication, and self-service password reset
- Make plans to get rid of legacy authentications
- Move to a cloud-based security posture management approach using tools such as Azure Security Center, Defender 365, Azure PIM, and Azure Sentinel

We can use open-ended questions similar to the following, to evaluate the future plans of an organization:

1. What is the company's cloud strategy?
2. Is there any other plan to use Azure services in the future? If so, which services are the company looking to use?
3. If the answer is yes to the previous question, what is the time frame for that?
4. Are there any potential technical challenges that can have an impact on the company's cloud journey?
5. Are there any risks that can have an impact on moving to cloud services?
6. Were any feasibility studies done in order to evaluate the use of cloud services?

The answers to the preceding questions will help you determine the company's future in the cloud. Most of the time, it is best to gather answers to these questions from a business' decision-makers. We can't make decisions that are purely based on a long-term plan for businesses, but it will help engineers to consider this plan when designing a solution and leave room for future changes. Evaluation of business requirements is also part of Microsoft's **Cloud Adoption Framework** (**CAF**). More info about it is available on `https://bit.ly/3nLQW3W`.

Evaluating the security requirements

Azure AD comes with a lot of features and services that can be used to protect identities in cloud-only or hybrid environments against modern threats. Since it is a managed service, these features and services have been continuously improved according to trends. Each and every security feature or service is not needed for every Azure AD environment. We can decide on the security requirements based on the following points:

- **Nature of the business**: Identity protection and data protection are important for every infrastructure; however, some businesses require advanced protection due to the sensitivity of the data they process, compliance requirements, and legal requirements. As an example, a financial institution will require advanced protection compared to a college due to differences in the sensitivity of data.
- **Skills**: Even if you have advanced security features and services, if you do not know how to use them appropriately, they will not provide the expected benefits. Also, the configuration and regular maintenance of some of these features and services require knowledge. As an example, Azure Information Protection requires skills to set up appropriate policies and labels in order to get the benefits of data classification. These policies and labels will be different from business to business. So "skills" is a key factor when preparing security plans and budgets. If an organization doesn't have relevant skills, then they need to consider what to do to acquire the required skills.
- **Cost**: The security features and services we can use will be impacted by which Azure AD version businesses can afford. There are five different Azure AD versions available to choose from. For more details, visit `https://bit.ly/3cGtNtm`.
- **Compatibility**: Some of the Azure AD security features require a compatible device or environment in order to work. As an example, Azure AD password-less authentication (`https://bit.ly/3HJ6g9z`) only works with Windows 10 Azure AD-joined devices. If the existing environment is not compatible with Azure AD security features, the organization will have to either invest in bringing it to a compatible level or hold the implementation until they get there.
- **Resources**: By simply applying a security feature or enabling a service, we can't expect the environment to be 100% protected. We are fighting against human adversaries who change their tactics all the time. To stay one step ahead, we need to regularly evaluate the service configuration in place, evaluate collected logs, and keep an eye on alerts. All of this requires manpower and skills. If the organization is struggling with resources, it will not provide the expected results.

Over the years, I have written many articles about Azure AD-related security features. I have listed some of those for reference:

- **Step-by-Step Guide to Azure AD Privileged Identity Management – Part 1**: `https://bit.ly/3rkJ1Np`
- **Step-by-Step Guide to Azure AD Privileged Identity Management – Part 2**: `https://bit.ly/3oY6QHW`
- **Step-by-Step guide to setup temporally privilege access using Azure AD Privileged Identity Management**: `https://bit.ly/3nKZgkm`
- **Step-by-Step guide: Privileged access management in office 365**: `https://bit.ly/2ZjMBf1`
- **Step-by-Step Guide: Protect confidential data using Azure information protection**: `https://bit.ly/30UIwy2`
- **Step-by-Step Guide: Automatic Data Classification via Azure Information Protection**: `https://bit.ly/3cHejVY`
- **Step-by-Step Guide: On-premise Data Protection via Azure Information Protection Scanner**: `https://bit.ly/3DNPO5f`
- **Step-by-Step Guide: How to protect confidential emails using Azure information protection?**: `https://bit.ly/3FH7sID`
- **Step-by-Step Guide: How to track shared documents using Azure information Protection?**: `https://bit.ly/3xq3MYP`
- **Step-by-Step Guide to Azure AD Password-less Authentication (public-preview)**: `https://bit.ly/3l7qiAV`
- **Step-by-Step Guide: Using Microsoft Authenticator app (Public preview) to reset Azure AD user password**: `https://bit.ly/3cO16KU`
- **Azure AD Self-Service password reset for Windows 7/8.1 Devices**: `https://bit.ly/32ukbjE`
- **Conditional Access Policies with Azure Active Directory**: `https://bit.ly/3l4Tb0v`
- **Conditional Access with Azure AD B2B**: `https://bit.ly/3l4AtpX`
- **Step-by-Step guide to control data access using Azure cloud app security (based on content type)**: `https://bit.ly/3FItlqX`
- **Step-by-Step guide to manage Impossible travel activity alert using Azure cloud app security**: `https://bit.ly/3r4p792`
- **Step-by-Step guide to block data download using Azure Cloud App security**: `https://bit.ly/3r2PCeX`

Implementation of the above solutions is completely dependent on the business requirements. Before deciding on tools and services it is important to evaluate the business value they can deliver. When it comes to security, we always need to be mindful and invest appropriately.

Selecting the Azure AD version

Azure AD has five different editions. Azure AD free edition is the default edition for any Azure and Office 365 subscription. Basic and Premium versions are available through the Microsoft Enterprise Agreement, the Open Volume License program, and the Cloud Solution Provider program. Microsoft 365 E3 and E5 licenses also come with Azure AD Premium editions. More information about these editions can be found at `https://bit.ly/3oYV5km`.

Deciding on a sign-in method

In this section, we are going to evaluate different Azure AD sign-in options for hybrid environments.

The sign-in options for a hybrid environment are managed only through the **Azure AD Connect** configuration. These sign-in methods can simply be grouped into two categories:

1. **Authentication takes place against Azure AD**: In this category, cloud users and synced on-prem users will be directly authenticated via Azure AD. No authentication request will pass to on-prem AD for processing when users access Azure services.
2. **Authentication takes place against on-prem AD**: Under this category, if synced on-prem users try to authenticate into Azure services, it will be processed by on-prem AD.

Let's go ahead and see how each of the sign-in methods supported by Azure AD Connect fit into the above categories.

Password hash synchronization

This is the most commonly used method to allow on-prem AD users to authenticate into Azure services via Azure AD using their existing on-prem AD passwords. It is easy to implement as it is an extension of the Azure AD Connect directory synchronization. It doesn't require any other additional components to be installed on-prem other than Azure AD Connect.

It is also recommended that you use this as a backup sign-in method, even if you decided to use a federation or pass-through sign-in method.

When I talk to customers or engineers, on many occasions, I find that people think password hash synchronization uses cleartext passwords; however, this is completely wrong. On-prem AD doesn't use cleartext passwords either; it stores passwords in the form of hash values. Hash values are generated by using a one-way mathematical function against an actual user password. Even if someone has access to a hash value, it isn't possible to create a cleartext password out of it. Hash values can't be used as passwords to authenticate in AD either. So, how can it be insecure?

After password hash synchronization is enabled, Azure AD Connect retrieves the password hash values that are stored in on-prem AD. Before hash values are synced over to Azure AD, Azure AD Connect will further encrypt them using an advanced hashing algorithm. Once this data is synced to Azure AD, it can't be presented back to on-prem AD and be used to perform a successful authentication.

Let's go ahead and see how password hash synchronization works in a hybrid environment:

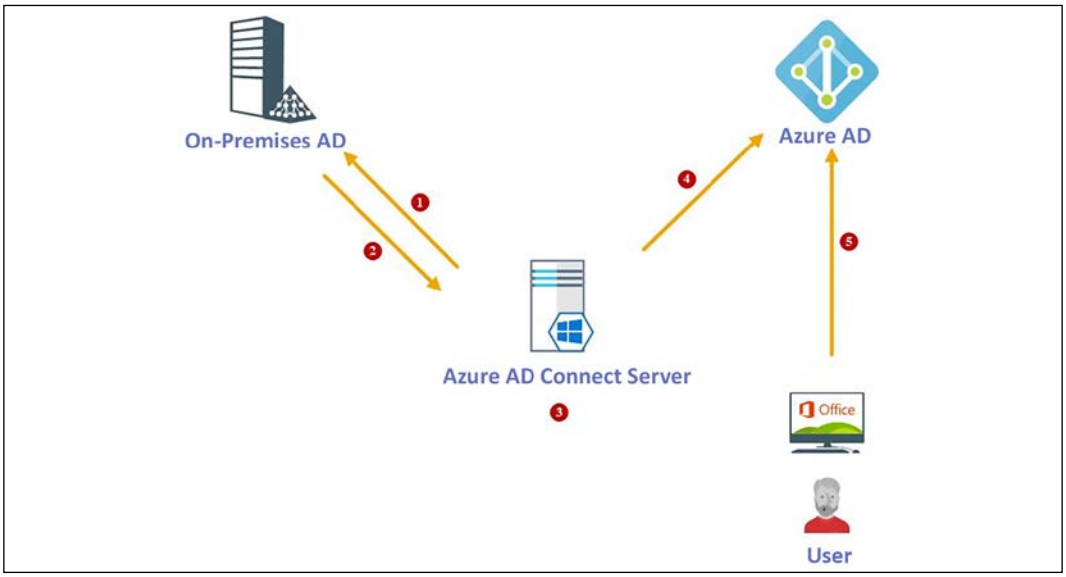


Figure 18.1: Password hash synchronization process

The following steps summarize the flow of password hash synchronization in a hybrid environment:

1. In this setup, **Azure AD** is configured to perform password hash synchronization.

First, **Azure AD Connect Server** is requesting password hash values stored in an **On-Prem AD** server. This is done by using the **Microsoft Directory Replication Service** (**MS-DRS**) remote replication protocol, which is standard for inter-**Domain Controller** (**DC**) replication. Azure AD Connect uses the service account that is used during the configuration to retrieve the password hashes. Hashes stored in AD are in the **Message Digest 4** (**MD4**) algorithm format (`https://bit.ly/3l4TvfJ`).

2. Then, the DC encrypts the MD4 hash values using a secure key. This key is the MD5 hash value of the **Remote Procedure Call** (**RPC**) session key and salt. Salt is providing additional security by adding random characters to the hash. Once the encryption is completed, the system sends the encrypted hash values to the synchronization agent via RPC. An on-prem DC also sends the salt to the synchronization agent via the inter-DC replication protocol.
3. Once the synchronization agent receives the encrypted hash, it uses the `MD5CryptoServiceProvider` class and salt to decrypt the hash envelope and retrieve the original hash values. Then, the synchronization agent converts a 16-byte hash into a 64-byte hash. After that, the synchronization agent adds a further 10-byte salt as extra protection for the original hash. This salt value is unique for each **User**. The synchronization agent then uses the original MD4 value and salt value in the **Password-Based Key Derivation Function 2** (**PBKDF2**) (`https://bit.ly/3cJ5JpW`). This process will generate a **Hash-Based Message Authentication Code** (**HMAC**) by using the **Secure Hash Algorithm 256-bit** (**SHA256**) hash function.
4. Here, the synchronization agent takes the 32-byte hash, per-user salt, and SHA256 iteration. It then transfers this to **Azure AD** over the **Secure Sockets Layer** (**SSL**).
5. When a **User** tries to authenticate into **Azure AD** with their on-prem password, the system will use the same method and generate a hash value for the provided password and compare it with the one saved in **Azure AD**. If both hashes match, **Azure AD** will accept user authentication.

With password hash synchronization, we also need to consider the following:

- Once Azure AD Connect is configured, password hash synchronization runs every two minutes. This schedule can't be changed.
- We can't enable/disable password hash synchronization for only certain users (unless it is controlled in the AD sync scope).
- Password changes will not affect already authenticated sessions.
- Password hash synchronization is not going to enable SSO. Users have to re-authenticate to Azure AD, and even log in using the domain computer.

- Password hash synchronization is only for AD user objects.
- Even if someone has access to the SHA256 hash of a user, it can't be used for on-prem AD authentication as the original MD4 hash was never transmitted to Azure AD.
- It is recommended to use password hash synchronization as a backup authentication method if you are using federation authentication or pass-through authentication.

Azure AD Connect password hash synchronization is the easiest way to enable hybrid identity for an organization. However, not every organization wants to use this method mainly due to compliance requirements.

Federation with Azure AD

Federation trusts between domains allow organizations to manage their own identities within their own environments. Azure AD also supports federation with on-prem AD. When a federation trust is in place, users can log in to Azure AD using the same on-prem AD passwords. With this method, on-prem users will always be authenticating via on-prem AD. We can use **AD Federation Services** (**AD FS**) or PingFederate to create federation trusts between Azure AD and on-prem AD.

In *Chapter 13*, I have explained how AD FS works with Azure AD and also demonstrated how to enable federation between Azure AD and AD FS. Please check *Chapter 13* if you need more details.

Pass-through authentication

To create federation trusts between Azure AD and on-prem AD, quite a bit of work is involved. We need additional servers, SSL certificates, licenses, a HA solution, firewall changes, and advanced configurations. But Azure AD Pass-through Authentication allows organizations to do on-prem-only user authentication with minimum changes being made to the environment. It uses an agent (`https://bit.ly/310Fxod`) that can be installed on any Windows server. By using pass-through authentication, users can do the following:

- Authenticate into Azure AD and on-prem AD using the same password.
- Authentication for on-prem users will always be processed through on-prem AD.
- The Pass-through Authentication feature comes as a part of Azure AD Connect. We don't need to pay anything to use this authentication method.

- Pass-through authentication agents are lightweight and you can install them with other applications. Multiple agents can be installed in multiple servers for HA purposes. No additional configuration is required to enable HA; it is just a matter of getting an agent installed.
- It is recommended to use at least three pass-through authentication agents in an environment.
- Pass-through authentication agents only make outbound connections to Azure AD via TCP port 443. Therefore, there is no requirement to place agents in a **Demilitarized Zone** (**DMZ**). Mostly, this will not require firewall changes at all as most environments allow outgoing TCP 443 traffic.
- Pass-through authentication agents and Azure AD use certificate-based authentication for communication. This certificate will be renewed automatically.
- It works well with Azure MFA and Conditional Access policies.
- It supports multi-forest environments as long as there is valid trust and name suffix routing in place.

Let's go ahead and see how pass-through authentication works:

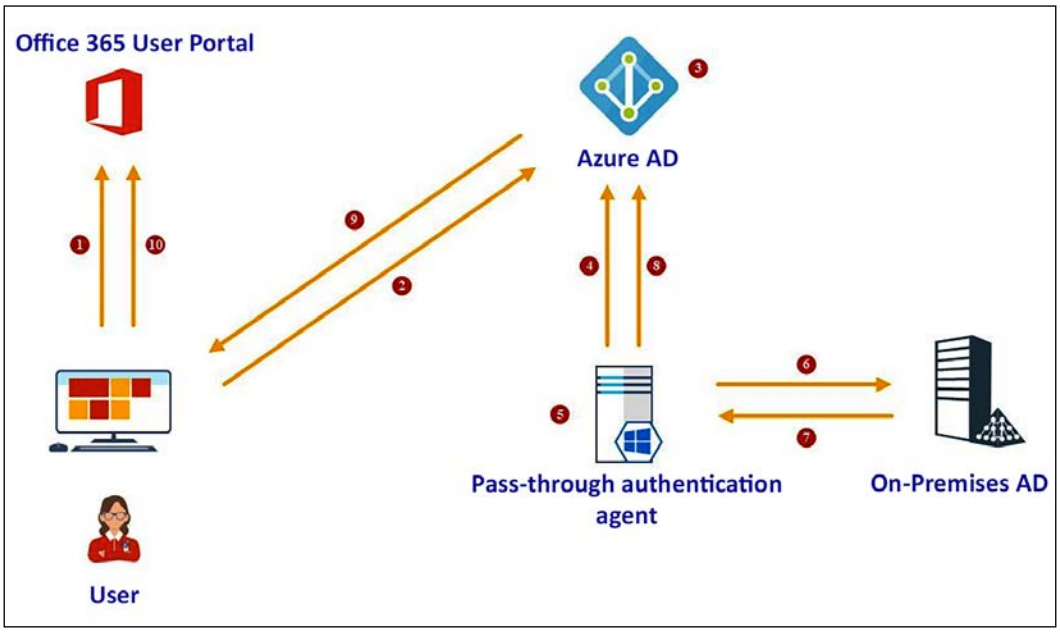


Figure 18.2: Pass-through authentication flow

In this example, **User** is trying to access **Office 365 User Portal**. They have pass-through authentication enabled:

1. **User** is accessing `https://www.office.com` using the browser installed on their computer.
2. In order to authenticate, **User** is directed to the Azure AD sign-in page. **User** then types the username and password and clicks on the **Sign in** button.
3. **Azure AD** receives the sign-in request and encrypts the password using the public key of the **Pass-through Authentication Agent**. This is retrieved from the Azure SQL database for the tenant. Azure AD then places the encrypted password in the service bus queue where the system will hold it until the **Pass-through Authentication Agent** retrieves it.
4. An on-prem **Pass-through Authentication Agent** retrieves the username and encrypted password from the service bus queue (using an outbound connection). This is done via a pre-established connection.
5. The **Pass-through Authentication Agent** decrypts the password using its private key.
6. The **Pass-through Authentication Agent** validates the username and password information with on-prem AD using the Win32 LogonUser API. This is the same API used by AD FS in the federated sign-in method.
7. The **On-Prem AD** evaluates the request and issues a response. It can be a success, failure, password expiry, or account lockout.
8. The Pass-through Authentication Agent passes the response back to Azure AD.
9. **Azure AD** evaluates the response and passes it back to the **User**.
10. If the response was successful, the **User** is allowed to access the application.

Later in this chapter, we will look into the configuration of the Pass-through Authentication feature.

Azure AD Seamless SSO

So far, we have learned about three different methods that we can use to integrate on-prem AD with Azure AD. This also allows on-prem users to use their existing domain usernames and passwords in order to authenticate into Azure AD integrated services. However, even though users can use the same usernames and passwords, when they access Azure AD integrated services from a corporate device (domain member), they still have to authenticate via the sign-in page. Azure AD Seamless SSO allows users to access Azure AD integrated services via corporate devices without re-authentication.

Azure AD Seamless SSO can be used with the password hash synchronization and pass-through authentication methods. However, its use is not supported by the federated authentication method:

- The Azure AD Seamless SSO feature can be enabled via Azure AD Connect, so it doesn't require any additional components in its environment.
- This is a free feature. We don't need to pay anything for it.
- Azure AD Seamless SSO only works in domain-joined devices (no need to use Azure AD join).
- If an SSO process has failed for any reason, the user can still authenticate using their username and password.
- It is supported to work with browser-based web applications and Office 365 clients with app versions 16.0.8730 and higher.
- Azure AD Seamless SSO accepts usernames as values associated with the **User Principal Name** (**UPN**) attribute or the Azure AD Connect **Alternate ID** attribute.
- Azure AD Seamless SSO uses the **Security Identifier** (**SID**) claim in the Kerberos ticket to find the relevant user object in Azure AD.
- Once Azure AD Seamless SSO is enabled, if an application can forward `domain_hint` (OpenID Connect) or the `whr` (SAML) parameter to identify the tenant, and the `login_hint` (OpenID Connect) parameter to identify the user, we can log in to Azure AD without typing usernames. This is also possible if the application uses a unique URL and is able to pass the domain info or tenant info.
- Once the user is logged in via SSO, if required, the user can still sign out and log in as a different user at any time.

Let's go ahead and see how Azure AD Seamless SSO works during the authentication process:

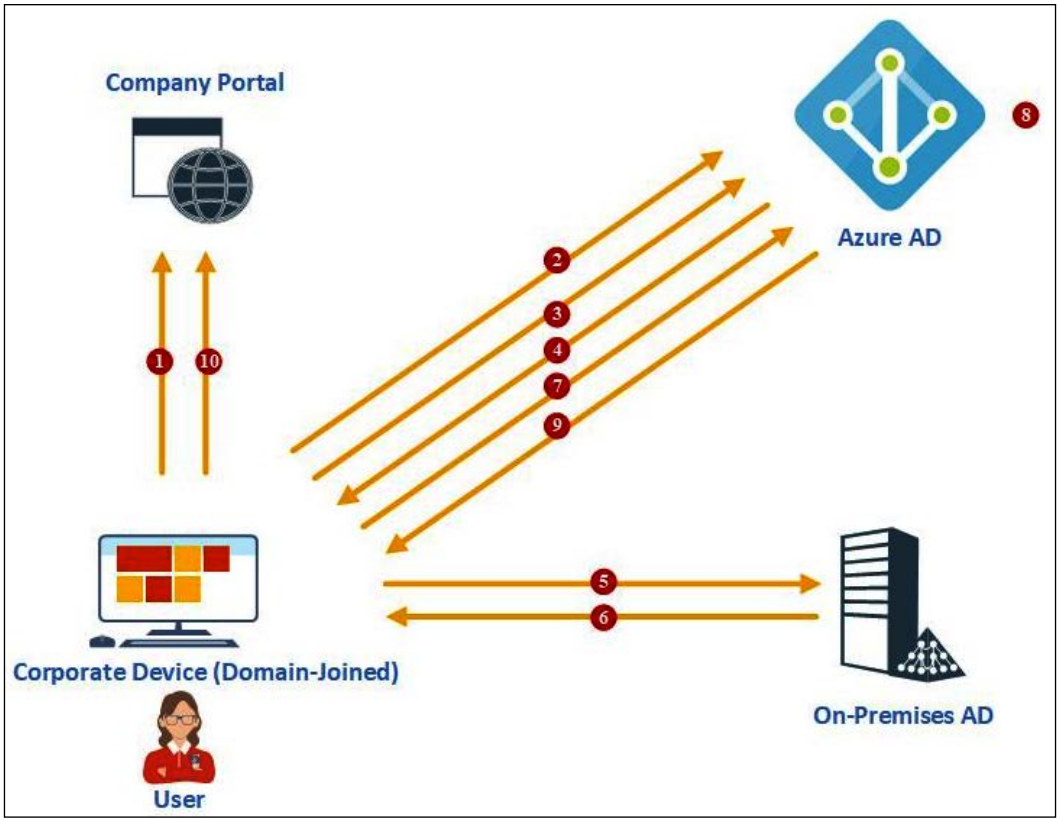


Figure 18.3: Seamless-SSO flow

In this example, **User** is trying to access their **Company Portal** using a **Domain-Joined** computer.

This is a hybrid setup, and the company already has Azure AD Seamless SSO enabled via Azure AD Connect:

1. **User** types `https://www.office.com` into the browser installed on the corporate device and presses *Enter*.
2. **User** is redirected to the **Azure AD** sign-in page.
3. The web application is not passing any domain info or tenant info, so **User** is typing their username into the **Azure AD** sign-in page.
4. **Azure AD** challenges the browser via a `401` unauthorized response to provide a valid Kerberos ticket.
5. The browser in the computer then requests a Kerberos ticket for the `AZUREADSSOACC` computer account. This account is created in **On-Prem AD** when Azure AD Connect is first configured for Azure AD Seamless SSO. This object represents the **Azure AD**.
6. **On-Prem AD** responds with the Kerberos ticket for the `AZUREADSSOACC` computer account. This is encrypted with the computer account's secret.
7. The browser responds back to **Azure AD** with the encrypted Kerberos ticket.
8. **Azure AD** decrypts the Kerberos key using its decryption key. This key was shared with **Azure AD** when Azure AD Seamless SSO was first enabled on Azure AD Connect.
9. If it is a valid ticket, **Azure AD** returns a token to the browser by accepting access.
10. **User** successfully logs into **Company Portal** without typing in the password again.

This feature allows us to improve the user experience greatly without any additional investment so why not?

Synchronization between on-prem AD and an Azure AD managed domain

An **Azure AD managed domain** is not a cloud version of on-prem AD, but it is supported by AD functions such as **Kerberos/NT LAN Manager** (**NTLM**) authentication, domain join, and **Lightweight Directory Access Protocol** (**LDAP**) queries. An Azure AD managed domain is not running any DCs that we can **Remote Desktop Protocol** (**RDP**) to or connect to using any other method. As tenants, we only have limited control over the Azure AD managed domain services. So far, we have learned how authentication works in a hybrid cloud environment.

In an Azure AD hybrid environment, two different identity platforms are going to work together and provide the same authentication experience to users, regardless of which environment they are located in. This is only possible through directory synchronization between two identity systems. Therefore, let's go ahead and see how synchronization works in a hybrid environment:

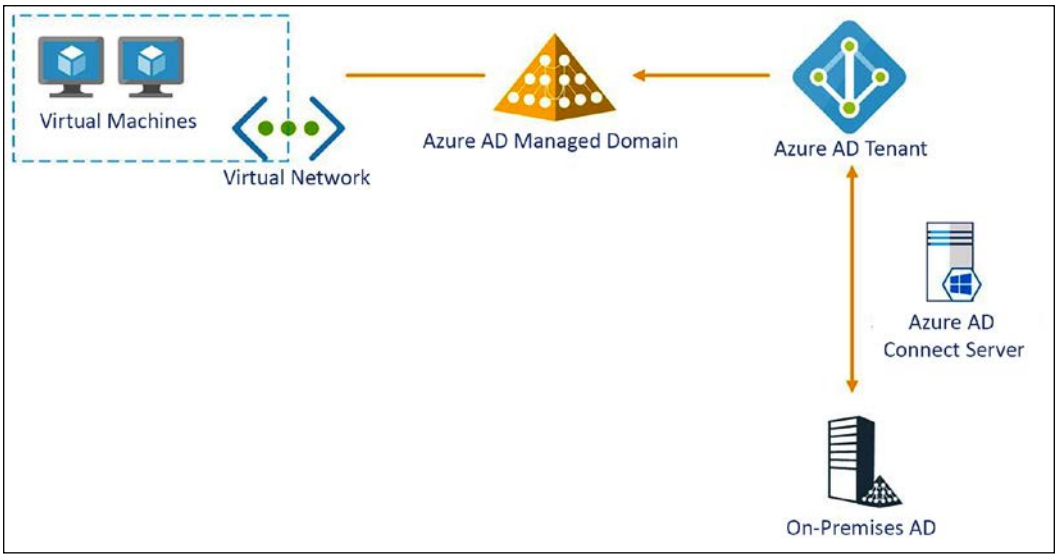


Figure 18.4: Azure AD hybrid topology with Azure AD managed domain

The preceding diagram demonstrates the high-level Azure AD hybrid topology.

1. In a hybrid environment, **Azure AD Connect Server** is responsible for syncing an on-prem user's UPN, SIDs, and group memberships to **Azure AD Tenant**.
2. **Azure AD Connect Server** can also be used to sync password hashes if required.
3. **Azure AD Connect Server** can also be used to define the user sign-in method to the directory (federated, pass-through authentication, and SSO).
4. Group policies, **System Volume** (**SYSVOL**) content, computer objects, OUs, and **SidHistory** attributes will not sync from **On-Prem AD** to the **Azure AD Managed Domain**.
5. Data synced from **On-Prem AD** to **Azure AD Tenant** will be synchronized to the **Azure AD Managed Domain**.
6. This synchronization between the **Azure AD Tenant** and the **Azure AD Managed Domain** will be automatic, and we can't change this schedule. Once a change is made in the **On-Prem AD**, it can take up to 30 minutes to sync back to the **Azure AD Managed Domain**.

7. The sync from the **Azure AD Tenant** to the **Azure AD Managed Domain** is one way.
8. The **Azure AD Managed Domain** will be attached to a virtual network. Any VM in this virtual network will be able to join the **Azure AD Managed Domain**.

So far in this chapter, on many occasions, I have mentioned Azure AD Connect. But what actually is Azure AD Connect? Let's find out more about Azure AD Connect in the next section.

Azure AD Connect

Azure AD Connect is the service that's responsible for integrating on-prem AD with an Azure AD tenant. Azure AD Connect replaces the previous versions of **Windows Azure AD Sync** (**DirSync**) and Azure AD sync components. Azure AD Connect has the following features:

- **Synchronization services**: This service checks whether Azure AD has the same identities as on-prem AD. If it doesn't, it will create the relevant objects in Azure AD.
- **Password hash synchronization**: Azure AD Connect can sync password hashes for on-prem AD users to the Azure AD tenant.
- **Federation service**: Azure AD Connect can be configured to authenticate via an on-prem AD FS or Ping Identity federation service. This is used by organizations that use domain-joined SSO, third-party MFA, smart cards, and so on.
- **Pass-through Authentication**: When this feature is enabled, on-prem users will always be authenticated via on-prem AD. This will be done via the authentication agent and it doesn't require additional servers and complex configurations such as AD FS farms.
- **Monitoring**: Azure AD Connect Health monitors the health of Azure AD Connect and its components. These stats can be viewed using the Azure portal.

The AD topology completely depends on the business requirements. Some businesses have a single domain and some may have structures with multiple domains and multiple forests. So, how does Azure AD Connect support these different topologies?

Azure AD Connect deployment topology

Azure AD Connect uses two different topologies to support on-prem AD deployments. However, there are certain limitations and unsupported configurations that we need to consider, which are as follows:

- **Single AD forest-single Azure AD**: This is the most commonly used deployment topology. When a user has a single AD forest, it can be synced to one Azure AD tenant. Even if it has multiple domains, it still can be used with one AD tenant. The Azure AD Connect express setup only supports this topology. However, at any given time, only one Azure AD Connect server can sync data to the Azure AD tenant. For HA, staging server support is available, which will be explained later in this section.
- **Multiple AD forests-single Azure AD**: Some organizations have multiple AD forests for various reasons. Azure AD supports syncing identities from all the forests into one Azure AD tenant. Each AD forest can have multiple domains as well. The AD Connect server should be able to reach all the forests, but this doesn't mean it needs to have AD trust between forests. The Azure AD Connect server can be placed in a perimeter network and then be allowed access to different forests from there. A rule of thumb in this model is to represent a user only once in Azure AD. If a user exists in multiple forests, it can be handled in two ways:
    - We can set the forest to match the user's identity using the mail attribute. If Microsoft Exchange is available in one or more forests, it may also have an on-prem **Global Address List Synchronization** (**GALSync**) solution. GALSync is a solution that is used to share Exchange mail objects between multiple forests. This will allow us to represent each user object as a contact in other forests. If a user has a mailbox in one forest, it will be joined with the contacts in the other forests.
    - If users are in an account resource forest topology that has an extended AD schema with Exchange and Lync, they will be matched using the `objectSid` and `sExchangeMasterAccountSid` attributes.

These options can be selected during the AD Connect configuration. There is no support for having multiple AD Connect servers in each forest syncing to one Azure AD tenant.

Staging the server

By design, it isn't possible to have multiple Azure AD Connect servers sync the same directory data to the same Azure AD tenant.

However, Azure AD Connect supports maintaining a second server in staging mode, which is ideal for HA. A server in staging mode reads data from all connected directories but will not sync it to the Azure AD tenant. It runs sync jobs as a normal Azure AD Connect server, so in the case of a disaster, it already has the latest data.

In the event of a primary server failure, we can use the Azure AD Connect wizard to fail over to the staging server. This method can be used to replace the existing AD Connect server. We can make all the relevant changes in staging mode, and when everything is ready, we can fail over to the newly implemented server. Maintaining multiple staging servers in an infrastructure is also supported.

Before installing the AD Connect server, we need to check whether the existing environment meets the following requirements. They can be found at `https://bit.ly/3xiU9Lk`:

- The AD forest functional level must be Windows Server 2003 or later.
- If you plan to use the password writeback feature, then the DCs must be on Windows Server 2008 (with the latest SP) or later. If your DCs are on 2008 (pre-R2), then you must also apply the `KB2386717` hotfix.
- The DC used by Azure AD must be writable. Using a **Read-Only Domain Controller** (**RODC**) is not supported, and Azure AD Connect does not follow any write redirects.
- There is no support for using on-prem forests/domains using **Single Label Domains** (**SLDs**).
- There is no support for using on-prem forests/domains using dotted NetBIOS names (names with a period in them).
- Azure AD Connect cannot be installed on a small business server or Windows Server Essentials. The server must use Windows Server Standard or better.
- The Azure AD Connect server must have the full GUI installed. There is no support for installing it on Server Core.
- Azure AD Connect must be installed on Windows Server 2008 R2 or later. This server can be a DC or a member server if you're using express settings. We also can use a standalone server with custom settings if required. However, the best practice is to use a domain member server for Azure AD Connect. If it is a dedicated server, it's more appropriate.
- If you install Azure AD Connect on Windows Server 2008 R2, then make sure to apply the latest hotfixes from Windows Update. The installation cannot be started with an unpatched server.

- If you plan to use the password synchronization feature, then the Azure AD Connect server must be on Windows Server 2008 R2 SP1 or later.
- If you plan to use a group-managed service account, then the Azure AD Connect server must be on Windows Server 2012 or later.
- The Azure AD Connect server must have .NET Framework 4.5.1 or later and Microsoft PowerShell 4.0 or later installed.
- If AD FS is being deployed, the servers where AD FS or Web Application Proxy are installed must be Windows Server 2012 R2 or later. Windows Remote Management must be enabled on these servers for remote installation.
- If AD FS is being deployed, you need SSL certificates.
- If AD FS is being deployed, then you need to configure name resolution.
- If your Global Administrators have MFA enabled, then the `https://secure.aadcdn.microsoftonline-p.com` URL must be in the trusted sites list. You are prompted to add this site to the trusted sites list when you are prompted for an MFA challenge and it has not been added before. You can use Internet Explorer to add it to your trusted sites.
- Azure AD Connect requires a SQL Server database to store identity data. By default, SQL Server 2012 Express LocalDB (a light version of SQL Server Express) is installed. SQL Server Express has a 10 GB size limit that allows you to manage approximately 100,000 objects. If you need to manage a higher volume of directory objects, you need to point the installation wizard to a different installation of SQL Server.
- If you use a different SQL Server version, then these requirements apply:
    - Azure AD Connect supports all flavors of Microsoft SQL Server from SQL Server 2008 (with the latest service pack) to SQL Server 2016. Microsoft Azure SQL Database is not supported as a database.
    - You must use a case-insensitive SQL collation. These collations are identified as having `_CI_` in their name. There is no support for using case-sensitive collation, which is identified by `_CS_` in the name.
    - You can only have one sync engine per SQL instance. There is no support for sharing a SQL instance with **Forefront Identity Manager** (**FIM**)/**Microsoft Identity Manager** (**MIM**) Sync, DirSync, or Azure AD Sync.

Azure AD Connect is a lightweight software that we need to install on an on-prem server. The configuration is always managed locally. But now we have another option for the synchronization of on-prem users and groups.

Azure AD Connect cloud sync

With this new service, we do not have to configure synchronization settings in an on-prem server anymore. We only need to install a lightweight agent on selected servers and the rest of the configuration will be completely managed from the cloud. This service can also be installed and used with the Azure AD Connect sync tool side by side. The Azure AD Connect cloud sync service provides a number of advantages:

1. This service allows you to enable synchronization from multiple disconnected AD forests to one Azure AD tenant. This is ideal for organizations that are going through a merger/acquisition. This allows you to accelerate the hybrid identity journey with minimal infrastructure changes.
2. In a domain, we can only have one active Azure AD Connect sync server. We can keep another Azure AD Connect server in staging mode but in the event of primary server failure, we need to manually bring the staging server to production. With Azure AD Connect cloud sync, we can use multiple agents for HA. We do not need to change any configuration in the event of a single agent server failure.
3. The agent configuration process is very straightforward. We only need to install a lightweight agent in on-prem servers and the rest of the configuration is completely handled in the cloud.

This new method also has a few limitations compared to Azure AD Connect sync. So it is important to understand these limitations before deciding on the synchronization method:

1. This service cannot connect to LDAP directories.
2. Azure AD Connect cloud sync cannot sync device objects from AD to Azure AD.
3. It also cannot sync custom AD attributes. But Azure AD Connect syncs custom attributes to Azure AD; I demonstrate the process in *Chapter 8*.
4. Azure AD Connect cloud sync does not support the pass-through authentication method.
5. Another major limitation to this service is it doesn't support Azure **AD Domain Services** (**AD DS**).
6. It also can't filter on objects' attribute values and does not support write back (passwords, groups).
7. Azure AD Connect cloud sync also has limitations on the number of objects it can sync. This service can sync up to 150,000 AD objects per domain. It also can only sync groups with less than 50,000 members.

Please note, the above-mentioned list may change with future releases.

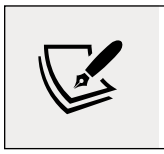
Azure AD Connect cloud sync prerequisites

Before we start installing the agent, there are certain prerequisites for Azure AD Connect cloud sync that we need to look into:

1. We need Azure AD administrator account access to configure the Azure AD Connect cloud sync.
2. We need a Domain Administrator/Enterprise Administrator account in AD to create a new **group managed service account** (**gMSA**) for the agent service.
3. The on-prem AD schema needs to be Windows Server 2016. If you are running older DCs, you can do this by adding at least one DC with Windows Server `2016/2019/2022`.
4. The servers selected for the cloud provisioning agent installation must be domain joined and running at least Windows Server 2016.
5. The servers should also have the .NET 4.7.1+ runtime.
6. The PowerShell policy for agent servers should be set to Undefined or the RemoteSigned setting.
7. Agent servers should be able to access the following URLs:
    - `*.msappproxy.net`
    - `*.servicebus.windows.net`
    - `login.windows.net`
    - `login.microsoftonline.com`
    - `mscrl.microsoft.com:80`
    - `crl.microsoft.com:80`
    - `ocsp.msocsp.com:80`
    - `www.microsoft.com:80`
8. Agent servers should be able to make outbound requests to Azure AD and Microsoft (to the above URLs) on TCP ports 80, 443, and 8080.
9. Agent servers must have TLS 1.2 enabled. You can enable TLS 1.2 by running the following PowerShell commands:

```
New-Item 'HKLM:\SOFTWARE\WOW6432Node\Microsoft\.NETFramework\
v4.0.30319' -Force | Out-Null

New-ItemProperty -path 'HKLM:\SOFTWARE\WOW6432Node\Microsoft\.
NETFramework\v4.0.30319' -name 'SystemDefaultTlsVersions' -value
'1'
```

```
PropertyType 'DWord' -Force | Out-Null

New-ItemProperty -path 'HKLM:\SOFTWARE\WOW6432Node\Microsoft\.
NETFramework\v4.0.30319' -name 'SchUseStrongCrypto' -value '1'
-PropertyType 'DWord' -Force | Out-Null

New-Item 'HKLM:\SOFTWARE\Microsoft\.NETFramework\v4.0.30319'
-Force | Out-Null

New-ItemProperty -path 'HKLM:\SOFTWARE\Microsoft\.NETFramework\
v4.0.30319' -name 'SystemDefaultTlsVersions' -value '1'
-PropertyType 'DWord' -Force | Out-Null

New-ItemProperty -path 'HKLM:\SOFTWARE\Microsoft\.NETFramework\
v4.0.30319' -name 'SchUseStrongCrypto' -value '1' -PropertyType
'DWord' -Force | Out-Null

New-Item 'HKLM:\SYSTEM\CurrentControlSet\Control\
SecurityProviders\SCHANNEL\Protocols\TLS 1.2\Server' -Force |
Out-Null

New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\
SecurityProviders\SCHANNEL\Protocols\TLS 1.2\Server' -name
'Enabled' -value '1' -PropertyType 'DWord' -Force | Out-Null

New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\
SecurityProviders\SCHANNEL\Protocols\TLS 1.2\Server' -name
'DisabledByDefault' -value 0 -PropertyType 'DWord' -Force | Out-
Null

New-Item 'HKLM:\SYSTEM\CurrentControlSet\Control\
SecurityProviders\SCHANNEL\Protocols\TLS 1.2\Client' -Force |
Out-Null

New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\
SecurityProviders\SCHANNEL\Protocols\TLS 1.2\Client' -name
'Enabled' -value '1' -PropertyType 'DWord' -Force | Out-Null

New-ItemProperty -path 'HKLM:\SYSTEM\CurrentControlSet\Control\
SecurityProviders\SCHANNEL\Protocols\TLS 1.2\Client' -name
'DisabledByDefault' -value 0 -PropertyType 'DWord' -Force | Out-
Null
Write-Host 'TLS 1.2 has been enabled.'
```



Source: `https://bit.ly/32vWsj1` You need to restart the server to complete the TLS configuration.

Once all the prerequisites are in place, we can go ahead with the installation.

Azure AD Connect cloud sync configuration

In this section, I am going to demonstrate how we can enable Azure AD Connect cloud sync. In my demo environment, I already have a valid Azure AD subscription and an on-prem AD environment that is running Windows Server 2022.

As the first step, we need to install the cloud provisioning agent. To do that,

1. Log in to the Azure portal as **Global Administrator** (`https://bit.ly/3FIofeu`).
2. Go to **Azure Active Directory | Azure AD Connect** and then click **on Manage Azure AD cloud sync**:

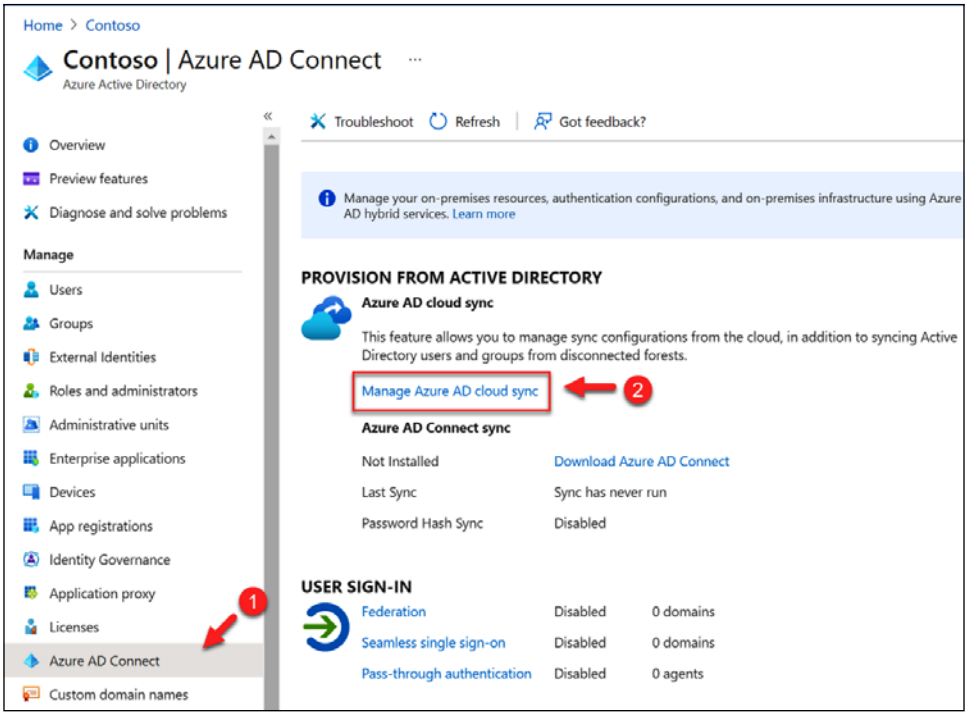


Figure 18.5: Manage Azure AD cloud sync

3. It will open the Azure AD Connect cloud sync page. In there click on the **Download agent** option to download the cloud provisioning agent:

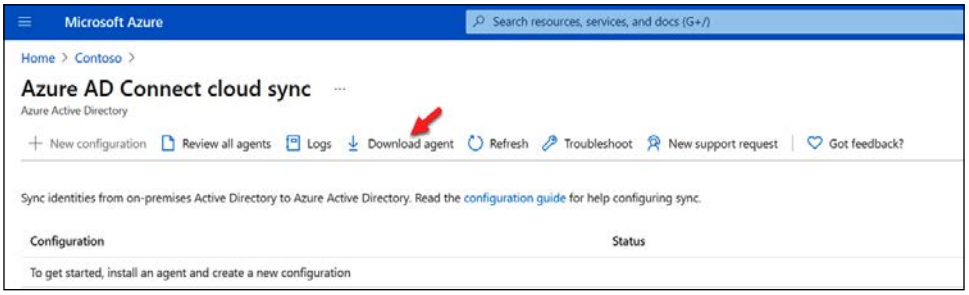


Figure 18.6: Download Azure AD cloud sync agent

4. In the new window, accept the terms and download the agent. Then copy the file to the agent server and run it as an administrator.
5. In the first window of the wizard, accept the license terms and click on **Install**.
6. After the agent is installed, the system will open the configuration wizard. In the initial window, click on **Next** to proceed.
7. Then in the next window, the system will prompt for Azure AD authentication. Use an Azure AD administrator account for authentication.
8. After that, the next step of the configuration is to create a gMSA account. For that, provide Domain Administrator account login details and click on **Next** to proceed:

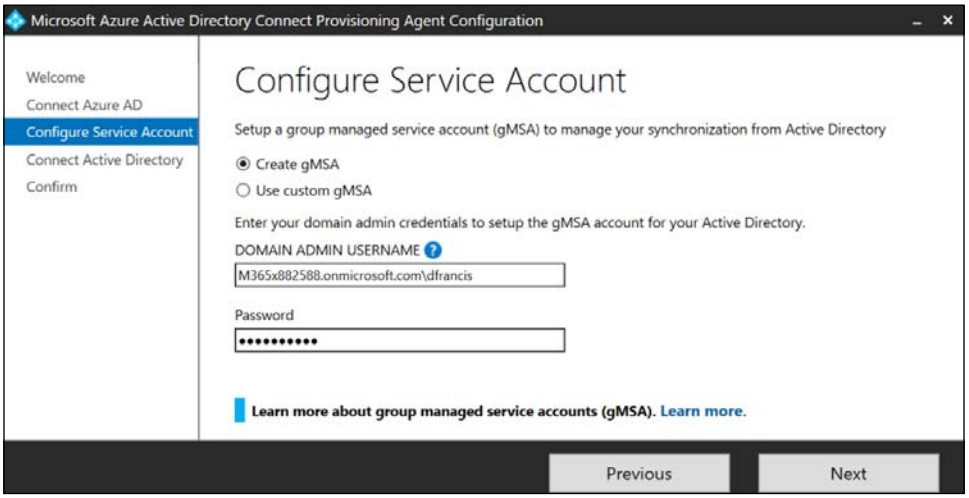


Figure 18.7: Create a gMSA account

9. In the next window, confirm the domain details and click on **Next** to proceed further:

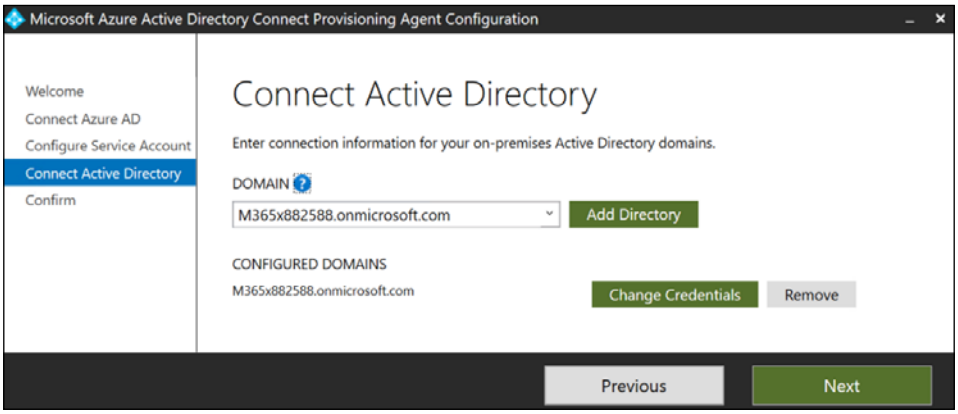


Figure 18.8: Connect to local AD

10. In the final window, review the settings and click on **Confirm** to complete the agent installation process.
11. Once installation is completed go back to the **Azure AD Connect cloud sync** page and then click on the **Review all agents** option:

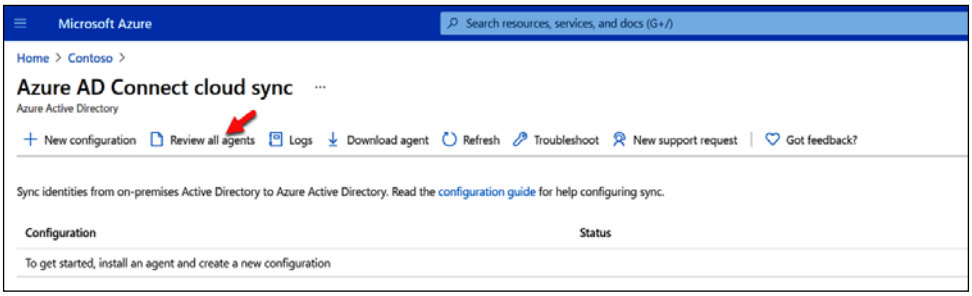


Figure 18.9: Review all agents option

12. On the **On-prem provisioning agents** page, we can see the status of the agents:

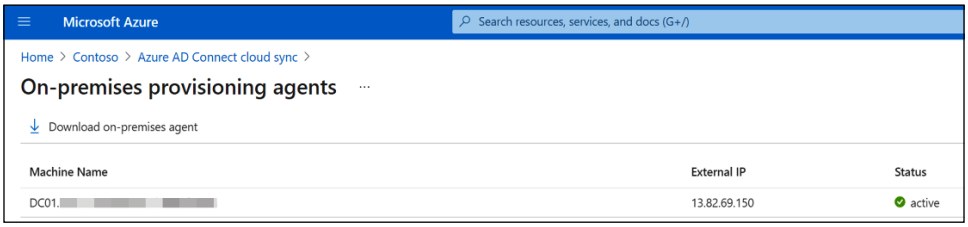


Figure 18.10: Health status of provisioning agents

13. Now we have the agents in place, the next step is to configure the sync. To do that, go back to the **Azure AD Connect cloud sync** page and click on **+ New configuration**:

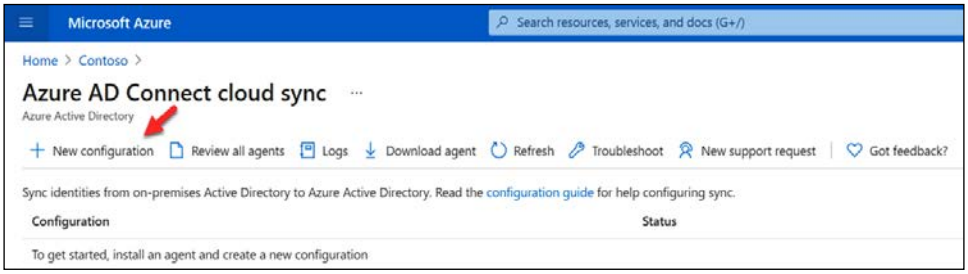


Figure 18.11: Set up new Azure AD cloud sync configuration

14. On the new page, confirm the on-prem domain name and click on the **Enable password hash sync** option. After that, click on **Create** to proceed:

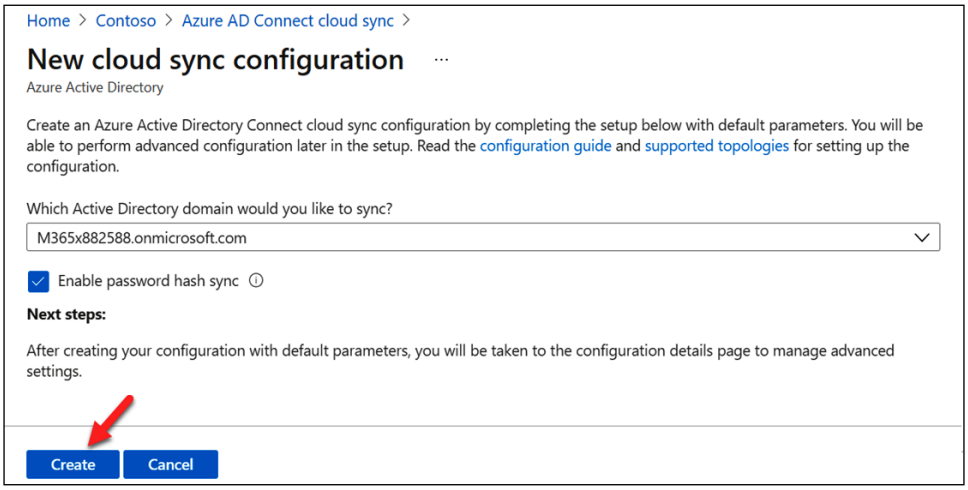


Figure 18.12: Azure AD cloud sync configuration – domain settings

15. After a few minutes, we can see the configuration settings. In there you can see five sections:
    - **Scope** – By default, the agent is going to sync all the objects from AD groups or OUs to sync instead of the entire directory.
    - **Manage attributes** – In this section, we can enable/disable password hash sync. Also, by using the **Map attributes** option we can map certain attributes from AD to Azure AD attributes.
    - **Validate (recommended)** – This is to verify if the sync is working as expected.

- **Settings –** In this section, we can define where to send email notifications and accidental deletion thresholds for additional object protection.
- **Deploy –** In here we can enable/disable sync from the agent as required.

16. In this demo, I used the default settings for the configuration except for the email address. Once the relevant settings are in place, click on **Save**:

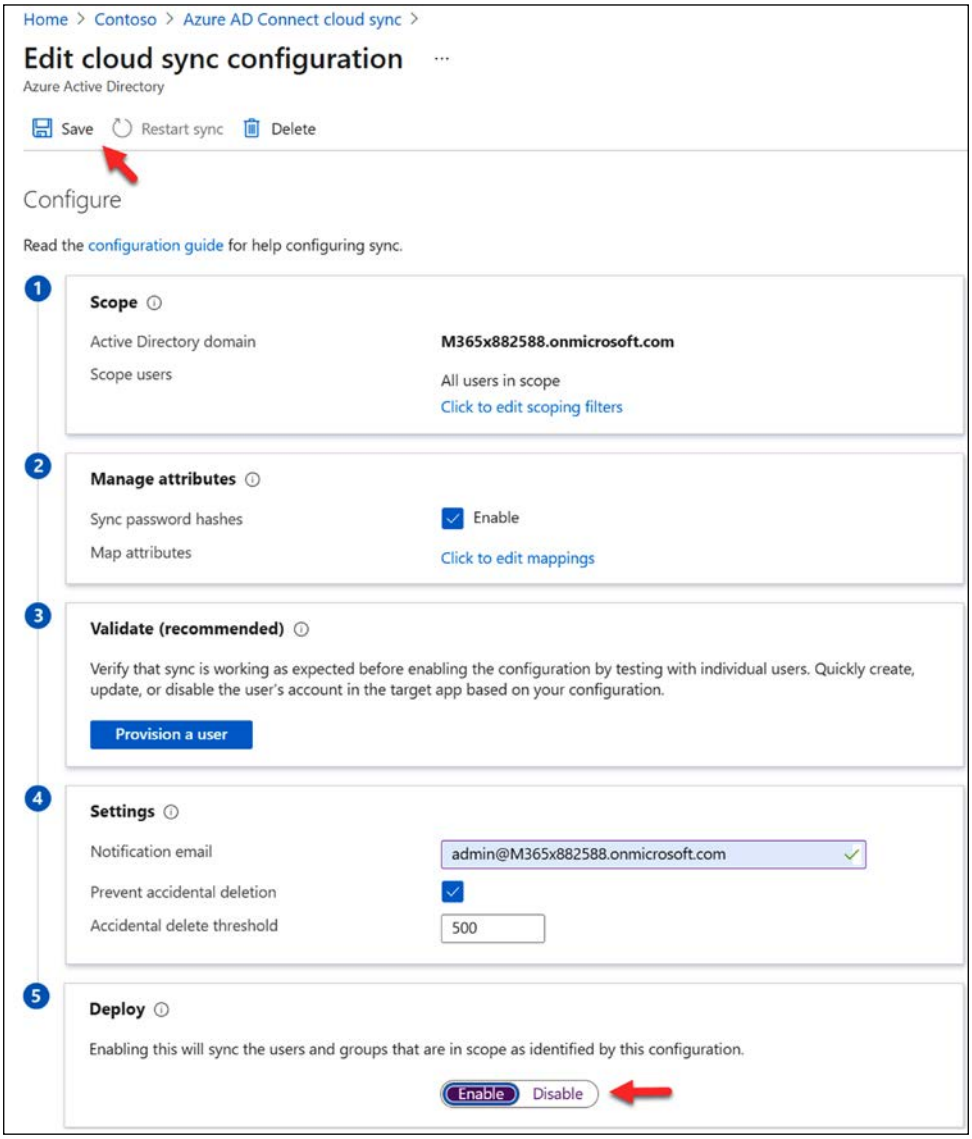


Figure 18.13: Save Azure AD cloud sync configuration settings

17. This will start the sync process. In a few minutes time, I can see AD objects start to appear in Azure AD:

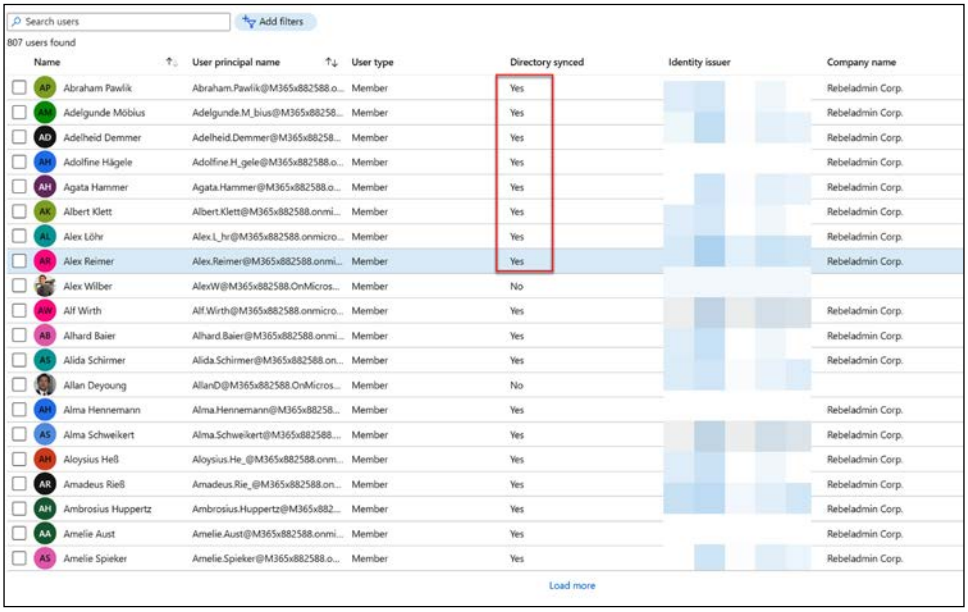


Figure 18.14: Sync status of user accounts

As we can see, the Azure AD Connect cloud sync is working as expected. In the next section, I am going to demonstrate how to set up an Azure AD managed domain and then use Azure AD Connect Pass-through Authentication to integrate with on-prem AD.

Step-by-step guide to integrating an on-prem AD environment with Azure AD

Before we start with the integration process, we need the following:

- **Valid Azure subscription**: We need to have a valid Azure subscription. It can be a pay-as-you-go subscription or a partner subscription. You can also get a free Azure demo account with £150 in credit. More information can be found at `https://bit.ly/3oV95Ma`.
- **Global Administrator account**: To set up Azure AD, you need to log in to Azure with an account that has Global Administrator account privileges.
- **Access to DNS**: If you are going to add a custom domain name, as part of the process, you need to verify the ownership of the domain name.

This is done by using a DNS record. Therefore, engineers need to have access to DNS servers.

- **Enterprise Administrator account**: In order to set up and configure Azure AD Connect, the engineers need to be members of the Enterprise Administrator group in the on-prem AD setup.
- **Connectivity**: The server running Azure AD Connect needs to have connectivity to Azure services. If your DCs do not have direct access to the internet prior to deployment, firewall rules need to be modified to allow the Azure service access on recommended ports.

More information about ports can be found at `https://bit.ly/3r5zJo5`. The service URL and IP range information can be found at `https://bit.ly/3nMancK`.

Once the aforementioned prerequisites are ready, we can move on to the implementation process. In this demo, I am going to cover the following:

- Creating a virtual network
- Creating an Azure AD instance
- Adding DNS server details to the virtual network
- Creating an Azure AD DC administrator group
- Creating a Global Administrator account for Azure AD Connect
- Setting up Azure AD Connect:
    - Enabling Pass-through Authentication
    - Enabling Azure AD Seamless SSO
- Enabling synchronization of NTLM and Kerberos credential hashes to Azure AD

If you are going to use Azure AD DS, we need to prepare things first such as a virtual network, DNS server records, administrator groups, etc. But if you're just going to use Azure AD, these steps can be skipped. It is not a must to use Azure AD DS. But if you're looking to leverage services such as domain join, group policy, LDAP, and Kerberos/NTLM authentication then you have to use Azure AD DS.

Creating a virtual network

Azure AD and other workloads should use the same virtual network so that they can be operated under the same managed domain. If you already have a subscription and have your virtual network set up, this step can be skipped:

1. Log in to the Azure portal as Global Administrator (`https://bit.ly/30XzSig`).
2. Click on **Virtual networks** from the left-hand navigation panel. Then, click on **+ Add**.
3. On the first page of the wizard, provide the following details:
    - **Name**: Provide a name for the virtual network. In this demo, I am using `REBELVMNet` as my virtual network name.
    - **Resource group**: Select or create a resource group for the virtual network. In my demo, I am using a new resource group for this called `REBELBOOK`.
    - **Location**: Select a location for the virtual network. Please note that we need to use the same location for the managed domain.
4. After the settings are in place click on **Next: IP Addresses**:

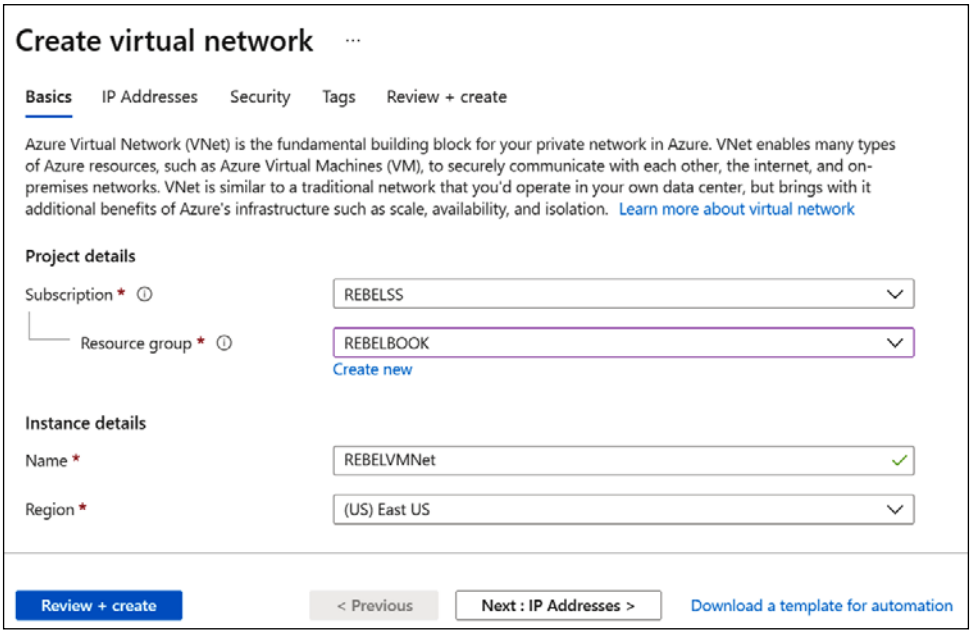


Figure 18.15: Create a new virtual network

5. In the next window, provide the following data:
    - **IPv4 address space**: This defines the **Classless Inter-Domain Routing** (**CIDR**) notation of the IP address range for the virtual network. Always use a large subnet for this as we can create a different subnetwork under this. In my demo, I am using `10.2.0.0/16` as the address space.
    - **Subnet name**: Here, we need to define a name for the subnet. I have used `REBEL-VN01`.
    - **Subnet address range**: We can define a subnet using this option. The Azure AD managed domain will also use this subnet. In my demo, I am using `10.2.0.0/24` as the address range.

Once the preceding details have been provided, click on **Review + create** to proceed with the virtual network deployment:

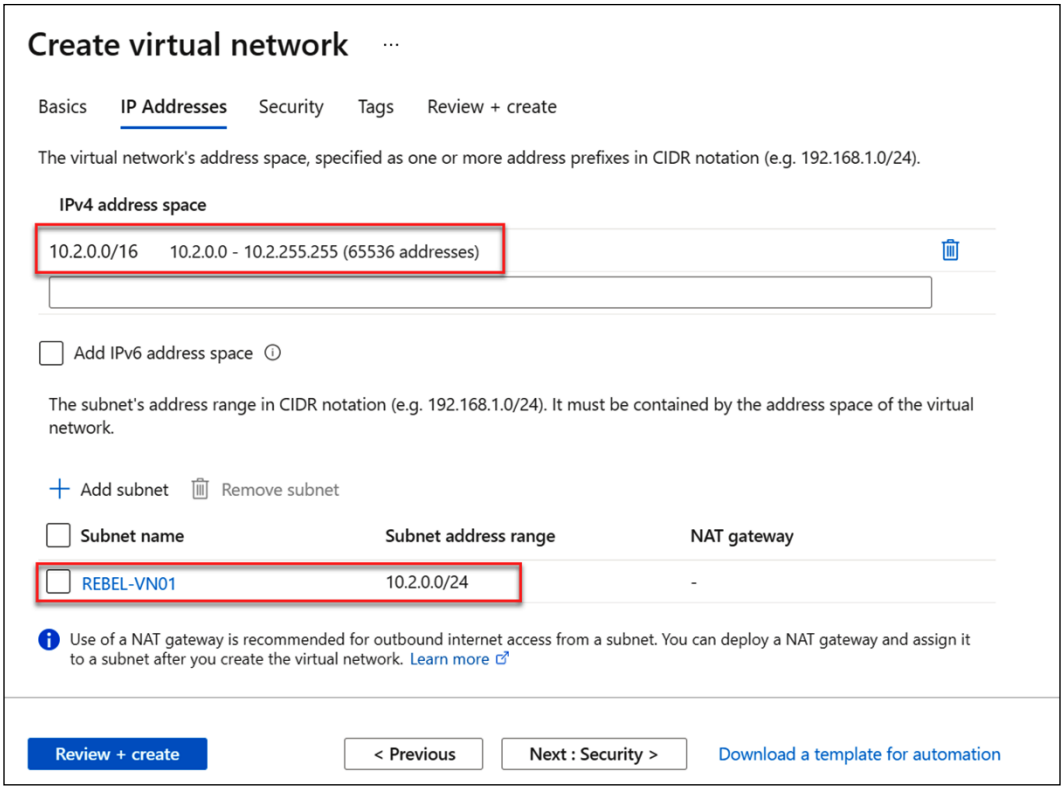


Figure 18.16: Virtual network subnet settings

Once the deployment has completed, the next step of the confirmation is to set up the Azure AD managed domain.

Setting up an Azure AD managed domain

The next step of the configuration process is to set up an Azure AD managed domain:

1. In order to do that, log in to the Azure portal (`https://bit.ly/3oUVqo7`) as Global Administrator.
2. Go to **All Services** | **Azure AD Domain Services**.
3. Click on **Create Azure AD Domain Services**:

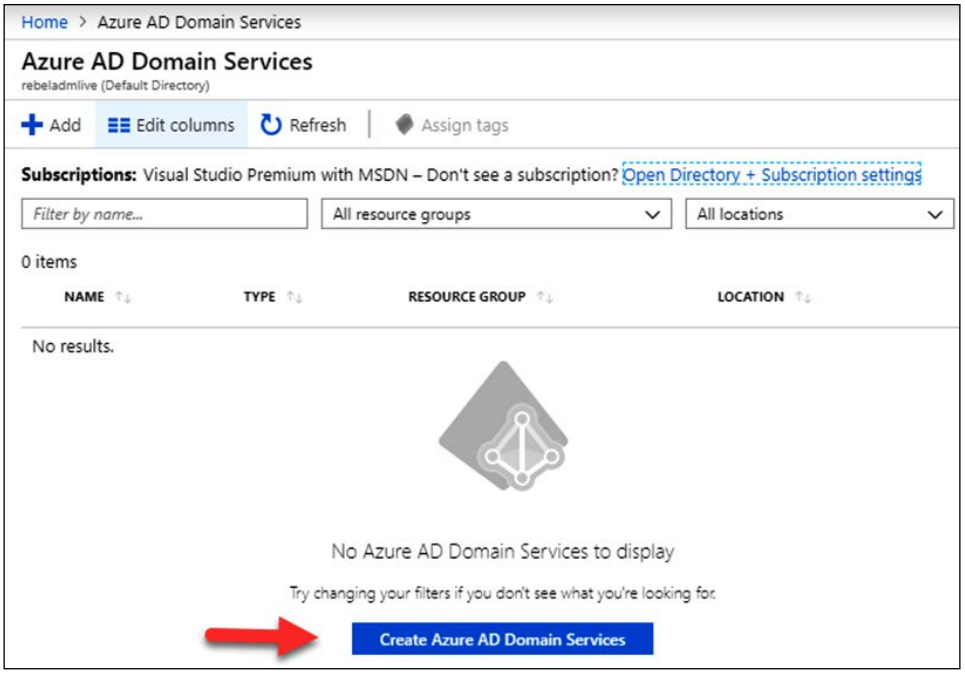


Figure 18.17: Initiating Azure AD DS setup

4. This will open up a wizard. Type in a **DNS domain name** for the service. It is recommended that you use the default tenant domain name in the beginning as we can add a custom domain later on, if required. Also, select the resource group we created in the previous step. In here, we also need to select the **SKU** for the service.

There are three levels of subscription for the service and you can find more details about these on `https://bit.ly/3nLCc4Y`. By default the forest type is set to **User**. If you need to, you can also set this up as a resource forest:

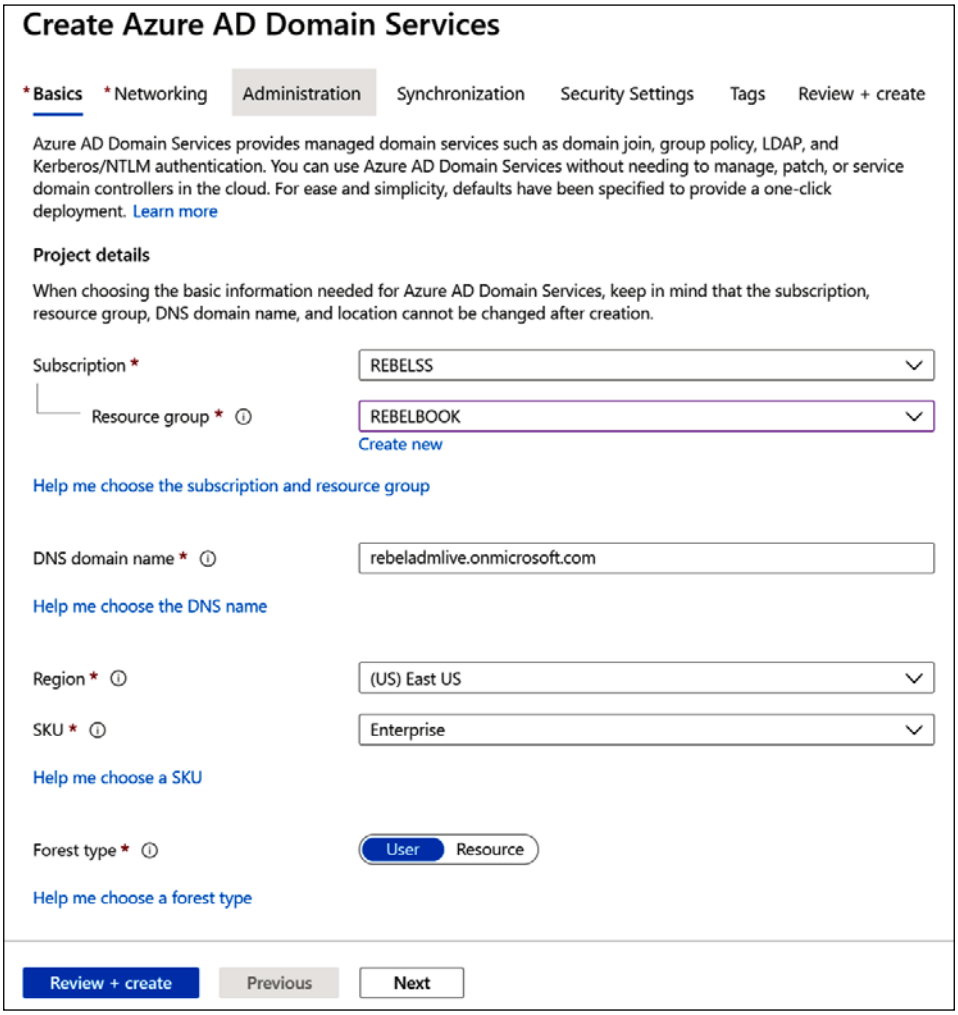


Figure 18.18: Azure AD DS configuration

5. In the next window, select the virtual network and subnet we created in the previous section. If needed, we can also create a new virtual network and subnet on this page:

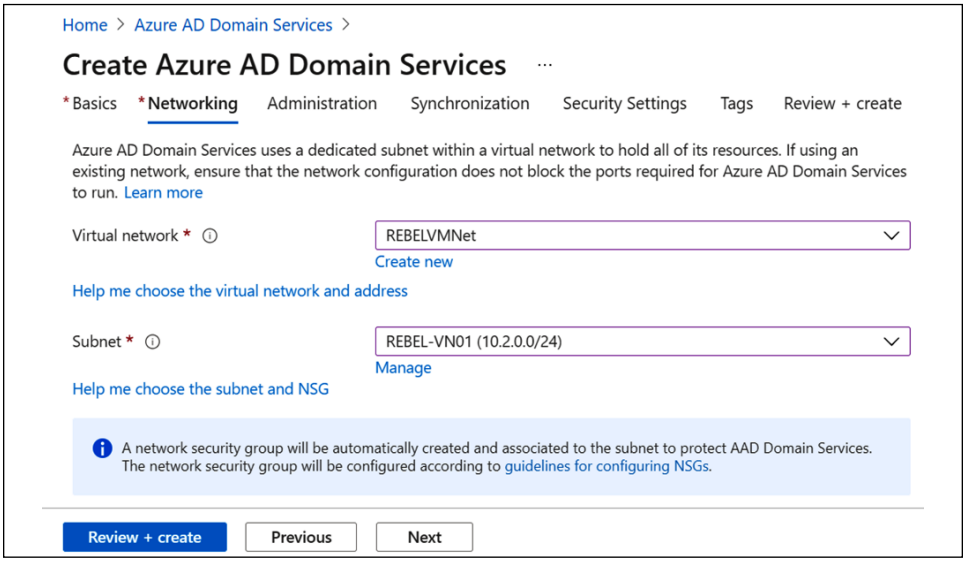


Figure 18.19: Azure AD DS – virtual network settings

6. In the next window, we can manage the members of the default **AAD DC Administrators** group. Members of the **AAD DC Administrators** group have administrative privileges over the managed domain. In there, we can add new members using the **Manage group membership** option:

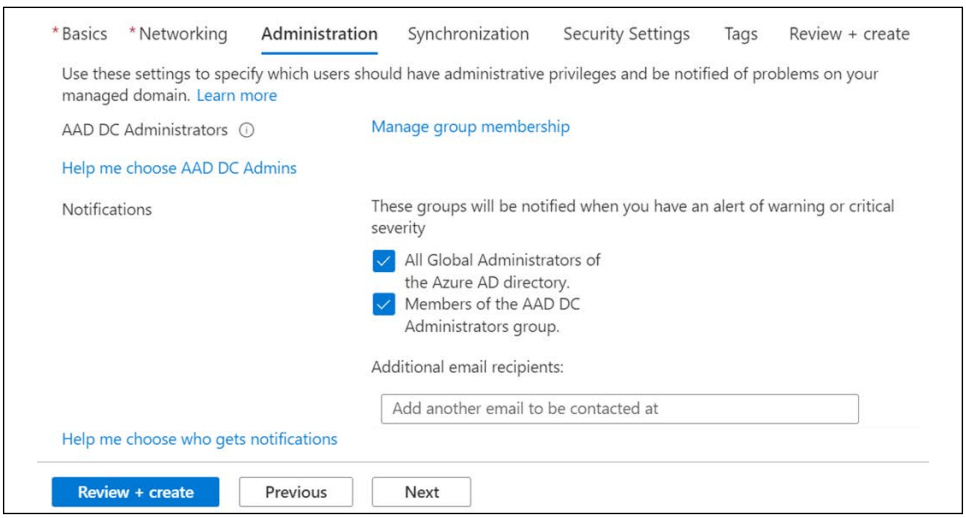


Figure 18.20: Azure AD DS – Domain Admin settings

7. On the next page, we can define the synchronization scope from the Azure AD tenant. In my demo, I will keep the default and sync all users and groups:

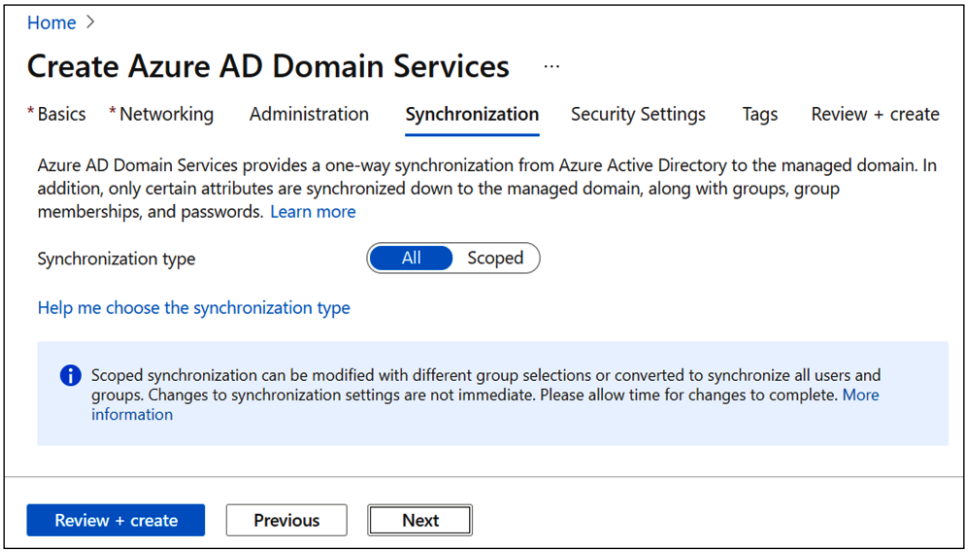


Figure 18.21: Azure AD DS – synchronization scope

8. On the **Summary** page, review the configuration settings and click **OK** to complete the setup process.

It will take around 30 minutes to complete the setup process. Once it is completed, we will be able to see the managed domain under the Azure AD Domain Services page:

Figure 18.22: Azure AD DS instance

The next step of the configuration is to configure the DNS server details. This helps us join machines to our managed domain.

Adding DNS server details to the virtual network

If we need to add a VM to this managed domain, the VM should be able to resolve the DNS name of the managed domain. This is done via a DNS server that belongs to the managed domain. We need to add this DNS server to the virtual network so that we can add VMs to the managed domain at a later time.

To do this, perform the following steps:

1. Go to **All Services** | **Azure AD Domain Services**.
2. Click on the managed domain we just created.
3. On the next page, click on **Configure**, which is under **Update DNS server settings** for your virtual network. This will add DNS servers to the relevant virtual network:

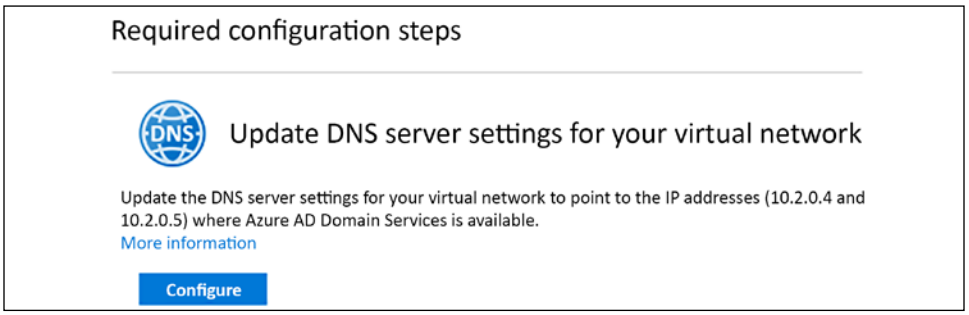


Figure 18.23: Update DNS records

4. Once the DNS update process has completed, we will be able to see the records under the relevant virtual network's DNS settings:

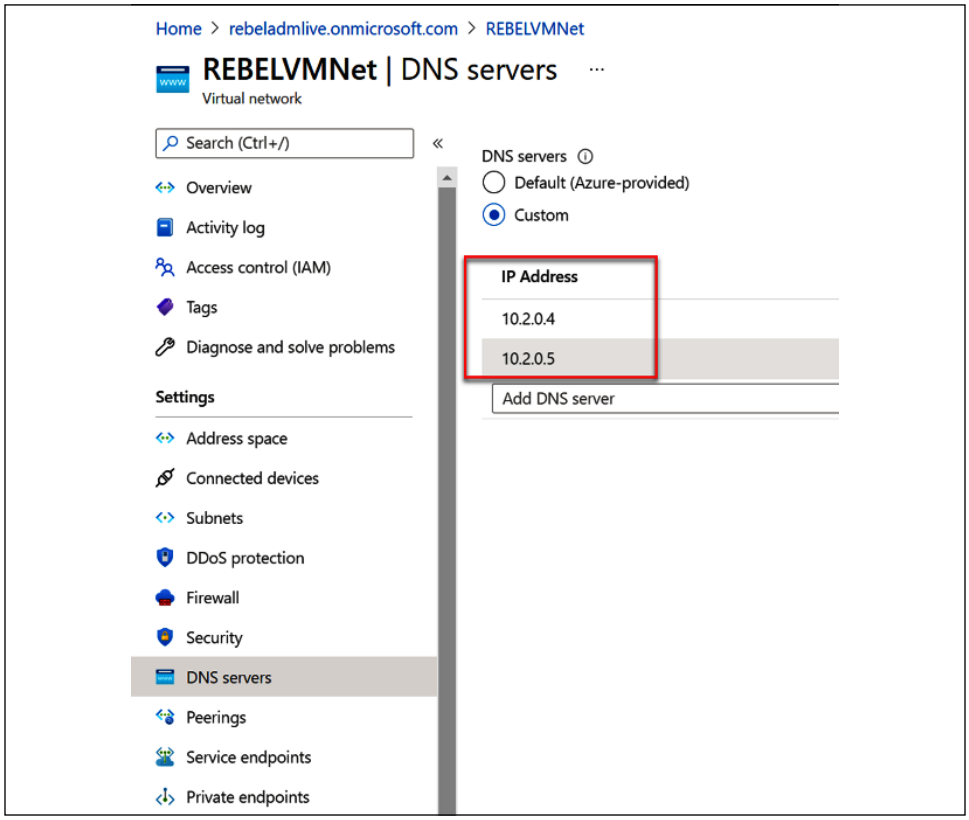


Figure 18.24: Custom DNS servers

This completes the configuration of Azure AD DS; the next step is to configure Azure AD Connect and sync the accounts from an on-prem domain. Before we do that, first we need to create an administrator account to use during the Azure AD Connect sync configuration.

Creating a Global Administrator account for Azure AD Connect

During the Azure AD Connect configuration, we require an account that has Global Administrator privileges (in Azure). It is recommended to use a separate account for this.

In order to create a user account, perform the following steps:

1. Click on **Azure Active Directory** in the Azure portal.
2. Click on **All users** | **+ New user**:

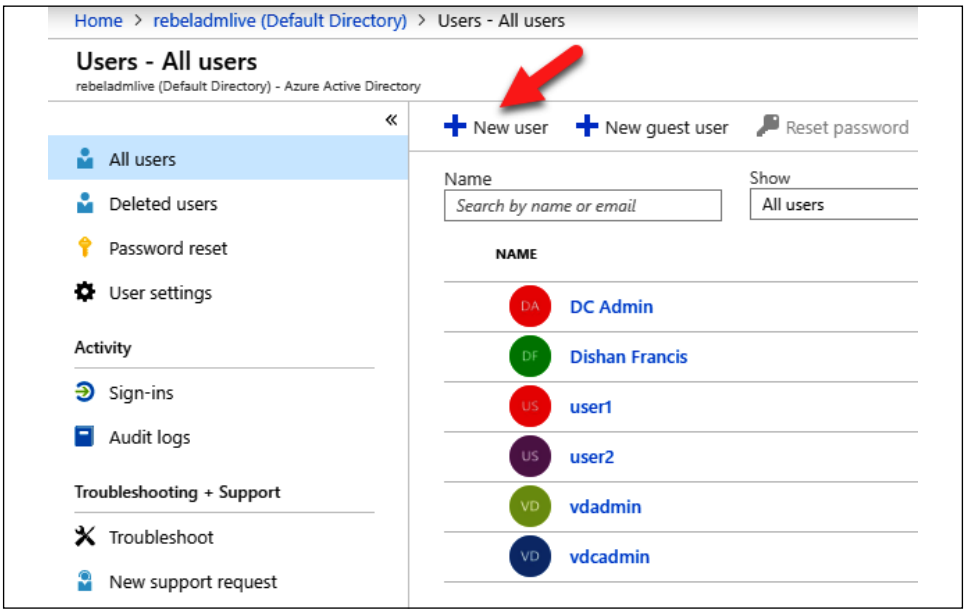


Figure 18.25: Create new user

3. Then, type the account name and username in the relevant fields. After that, click on **Directory role** and make sure that you **select Global Administrator**.
4. After creating the user, log in to the Azure portal using the new account details and make sure that the account is in a working state before using it for AD sync.

5. Also, make sure that this account is a member of the **AAD DC Administrators** group. This will provide administrative privileges to the managed domain.

The next step of the configuration is to set up Azure AD Connect. I have shown how to configure Azure AD Connect with federation in *Chapter 13*, *Active Directory Federation Services*. But here I am going to demonstrate the Azure AD Connect setup with pass-through authentication.

Setting up Azure AD Connect

In my demo environment, I have an on-prem DC running. It is operating in the Windows Server 2016 domain and forest functional levels. I would like to integrate it with the Azure AD managed domain we just created. In my setup, the on-prem AD uses the same domain name as the managed domain. In the production environment, you can use the custom domain name option and register the domain under Azure AD before going into the Azure AD Connect configuration.

With the Azure AD Connect configuration, I would like to do the following:

1. Sync all the users and groups to the Azure AD tenant
2. Configure Pass-through Authentication
3. Configure Azure AD Seamless SSO

The first step of the configuration will be to configure Pass-through Authentication agents.

Installing the Pass-through Authentication agent

Before we move into Azure AD Connect, we need to install the Pass-through Authentication agent. It is recommended to install the Pass-through Authentication agent on the same server as Azure AD Connect. In a production environment, it is recommended to install this agent in at least three servers:

1. Log in to the Azure portal (`https://bit.ly/30RTYdW`) as **Global Administrator**.

2. Click on **Azure Active Directory | Azure AD Connect | Pass-through authentication**:

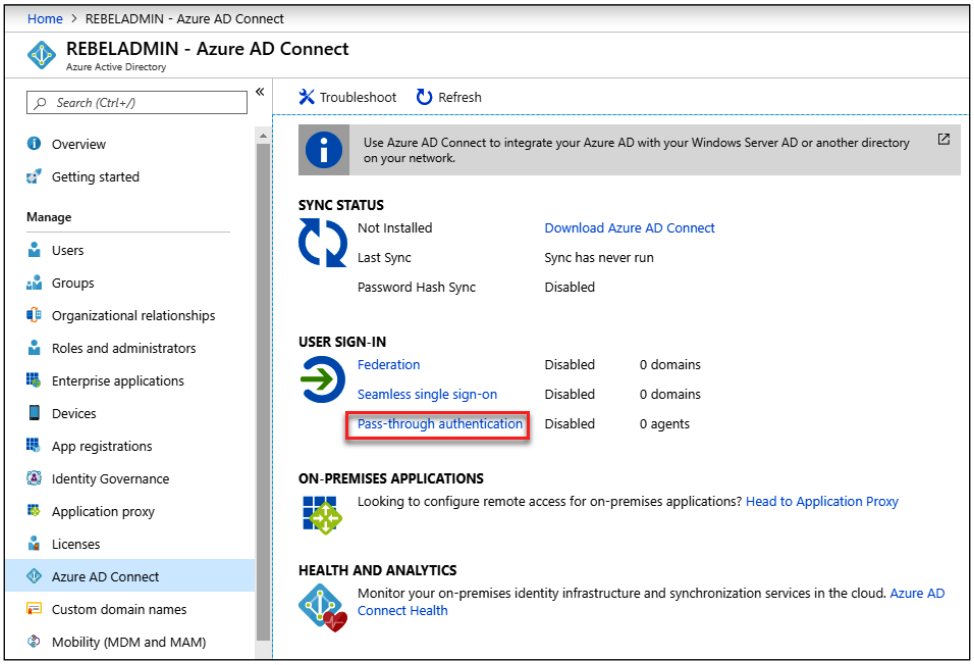


Figure 18.26: Pass-through authentication settings

3. Click on **Download** to download the file.
4. Once the `.exe` file has been downloaded, move to the server where it is going to be installed.
5. Then, double-click on the file and proceed with the installation:

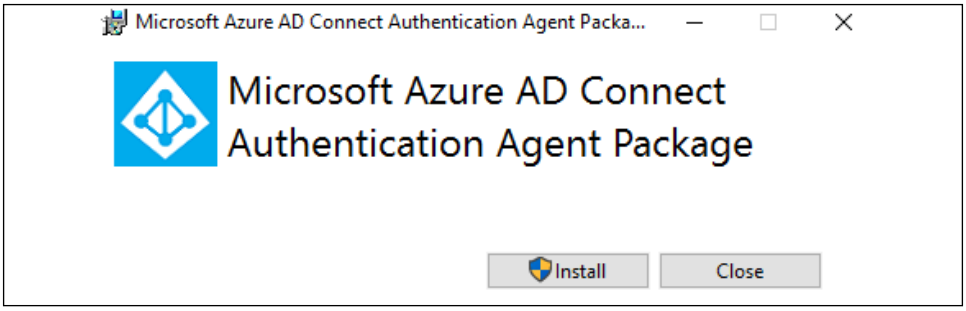


Figure 18.27: Start Azure AD Connect authentication agent installation

6. During the installation process, it will prompt for authentication. Use the Global Administrator user account we just created:

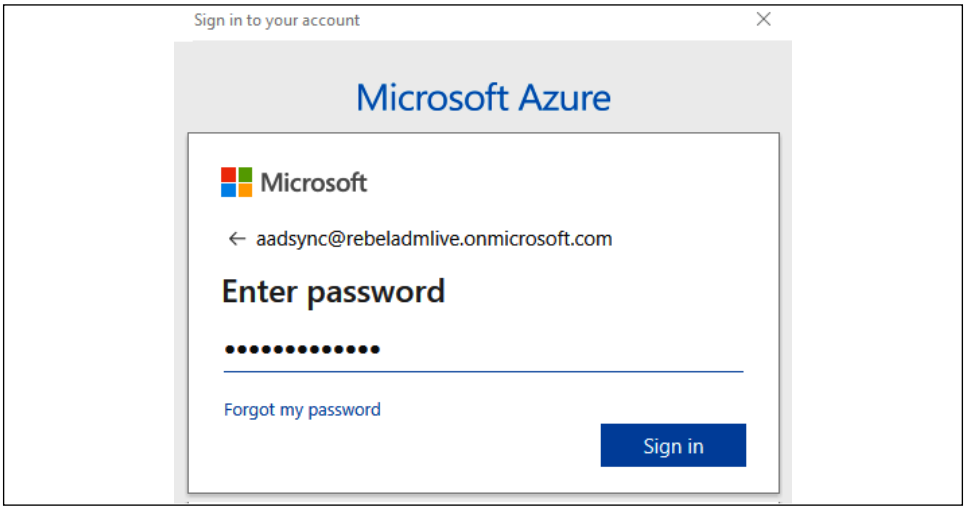


Figure 18.28: Azure AD authentication

7. Once it is installed successfully, we will be able to see it under **Azure Active Directory | Azure AD Connect | Pass-through authentication**:

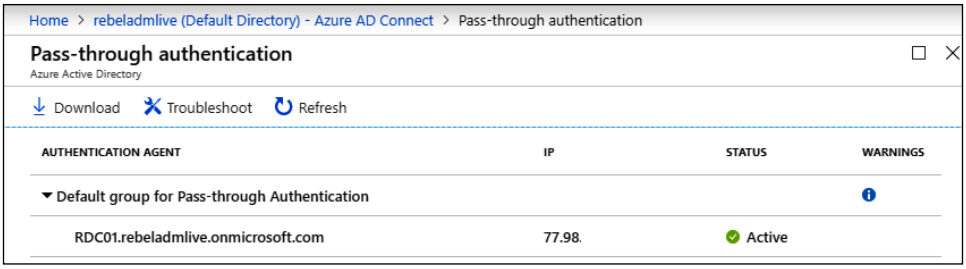


Figure 18.29: Pass-through authentication agent status

Now we have the agents in place. The next step is the configure Azure AD Connect.

Azure AD Connect configuration

Now, we have everything ready so that we can go ahead with the Azure AD Connect installation and configuration:

1. Log in to the on-prem server as a **Domain Administrator**.

2. Download the latest version of Azure AD Connect from `https://bit.ly/3l6JZZr`.
3. Run the `.msi` file as an administrator.
4. On the first page, accept the license terms and click on **Continue**.
5. On the next page, select the **Customize** configuration option:

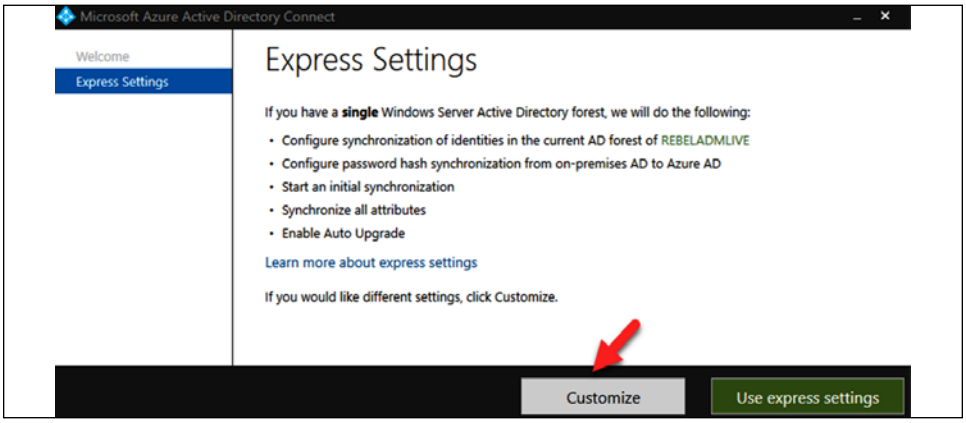


Figure 18.30: Azure AD Connect express setup

6. On the next page, keep the default selection and click on **Install**.
7. On the user sign-in page, select the **Pass-through authentication** (*step 1*) and **Enable single sign-on** (*step 2*) options, and then click **Next** (*step 3*) to proceed. This will enable pass-through authentication for the directory:

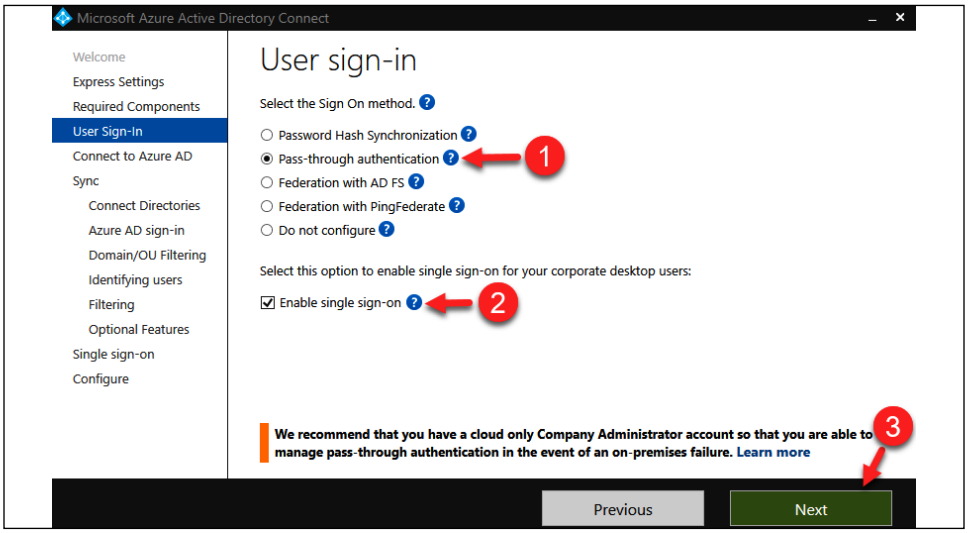


Figure 18.31: Azure AD Connect pass-through authentication setting

8. On the next page, log in to the user Azure AD Global Admin account we created in order to connect to Azure AD. After login validation, click on **Next** to continue.
9. On the **Connect Directories** page, provide an on-prem Domain Admin account and select **FOREST**.
10. For the next few pages, I kept the selections as their default values as I would like to sync all the user objects to Azure AD.
11. On the **Optional features** page, I am selecting **Password hash synchronization** to work as the backup sign-in option if pass-through authentication doesn't work:

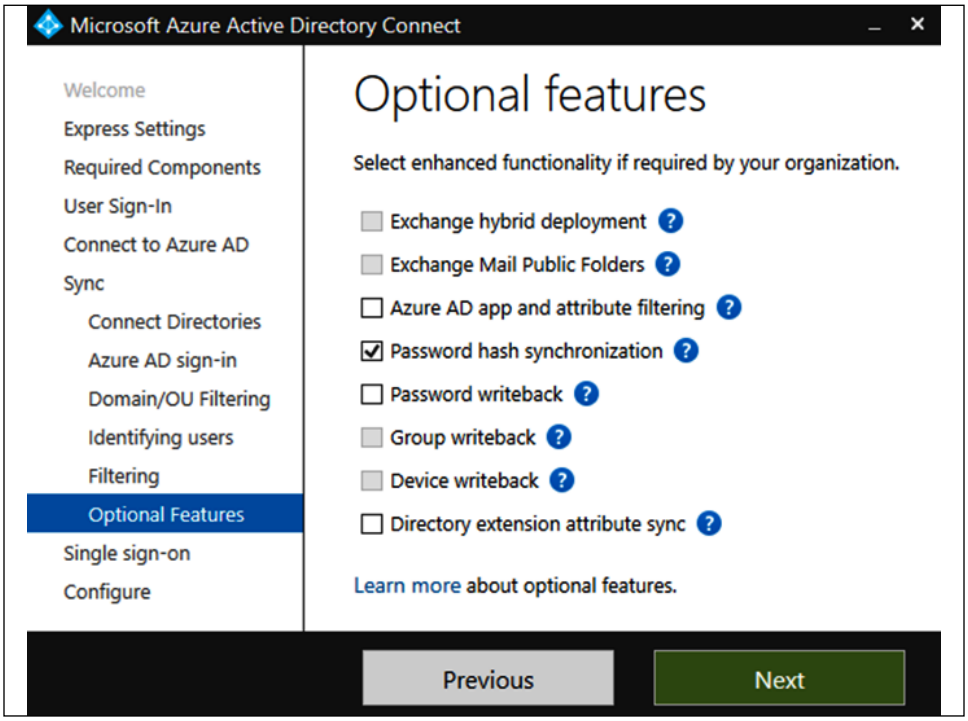


Figure 18.32: Azure AD Connect Optional features selection

12. Then, on the **Enable single sign-on** page, provide the Domain Admin credentials.
13. This will complete the configuration process of AD Connect. On the next page, click on **Install**.

14. After completing the installation, go to **Synchronization Service** in **Programs** and view the sync progress:

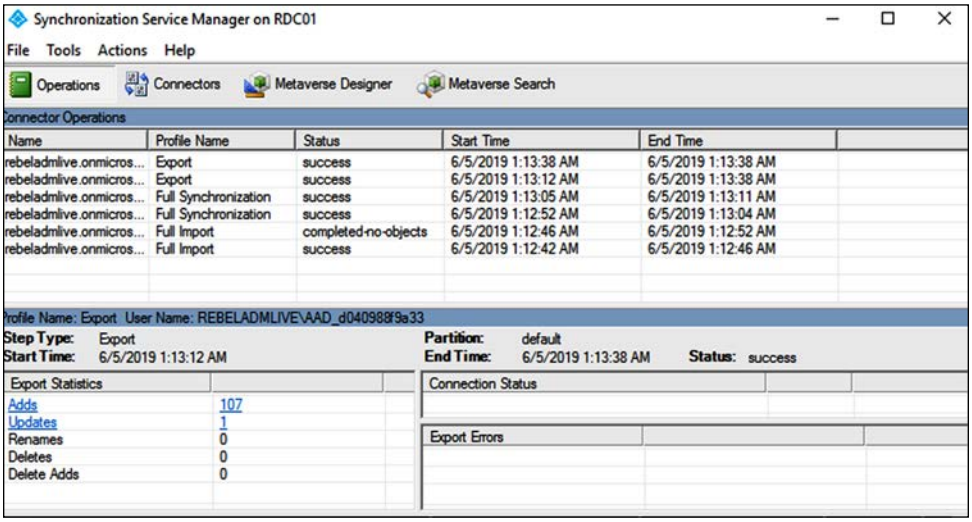


Figure 18.33: Azure AD Connect synchronization status

15. After a few minutes, we should be able to see synced users in the Azure AD tenant under **Azure Active Directory | Users**:

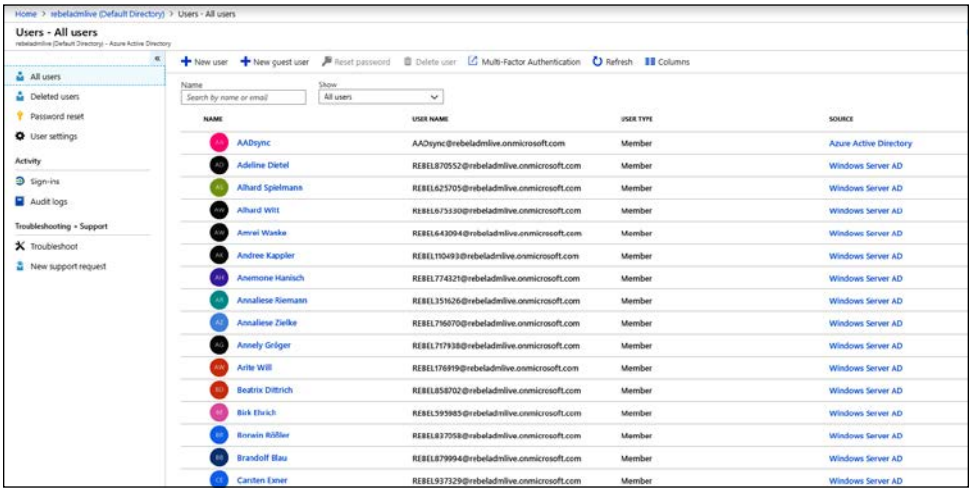


Figure 18.34: Azure AD user sync status

In the next part of the configuration, we are going to look into NTLM hash synchronization with Azure AD.

Syncing NTLM and Kerberos credential hashes to Azure AD

Azure AD Connect does not synchronize NTLM and Kerberos credential hashes to Azure AD by default. To use AD DS, we need to configure Azure AD Connect so that it synchronizes the credential hashes that are required for NTLM and Kerberos authentication. To do that, we need to run the following PowerShell script:

```
$adConnector = "<CASE SENSITIVE AD CONNECTOR NAME>"
$azureadConnector = "<CASE SENSITIVE AZURE AD CONNECTOR NAME>"
Import-Module adsync
$c = Get-ADSyncConnector -Name $adConnector
$p = New-Object Microsoft.IdentityManagement.PowerShell.ObjectModel.
ConfigurationParamter "Microsoft.Synchronize.ForceFullPasswordSync",
String, ConnectorGlobal, $null, $null, $null
$p.Value = 1
$c.GlobalParameters.Remove($p.Name)
$c.GlobalParameters.Add($p)
$c = Add-ADSyncConnector -Connector $c
Set-ADSyncAADPasswordSyncConfiguration -SourceConnector $adConnector
-TargetConnector $azureadConnector -Enable $false
Set-ADSyncAADPasswordSyncConfiguration -SourceConnector $adConnector
-TargetConnector $azureadConnector -Enable $true
```

We can find the AD connector and Azure AD connector names under **Start** | **Synchronization Service** | **Connectors**:

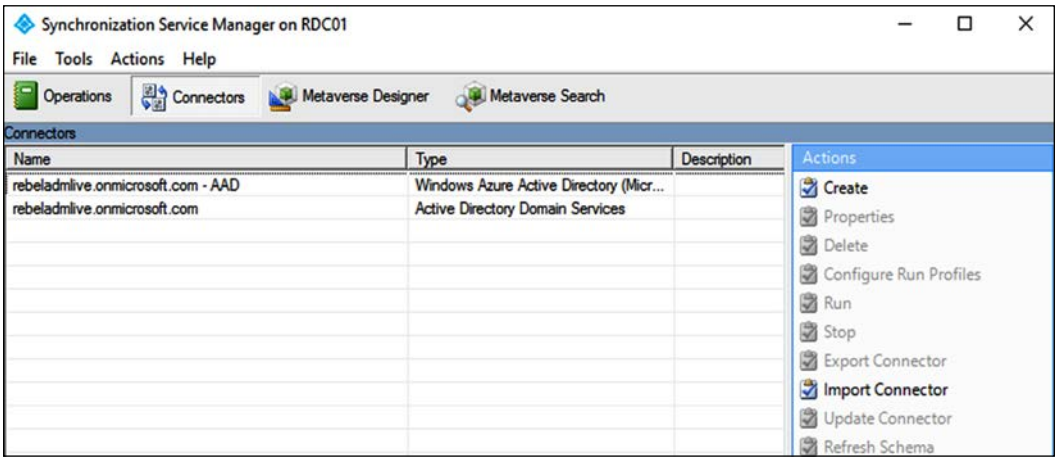


Figure 18.35: Connectors

This completes the configuration part of the Azure AD hybrid setup. Depending on the size of the directory, it can take up to 30 minutes to complete the sync process. Once the process is completed, users can authenticate into Azure AD using their on-prem usernames and passwords.

Enabling secure LDAP (LDAPS) for an Azure AD DS managed domain

In an on-prem AD environment, there can be applications or services that require integration with AD. With AD integration, the application can search for AD users, allow login, assign permissions, etc. This integration part is usually done using LDAP. By default, traffic over LDAP is not encrypted. Due to the vulnerabilities, Microsoft now recommends only using **secure LDAP** (LDAPS, LDAP over SSL) connections to DCs. In *Chapter 15*, I have demonstrated how we enable secure LDAP access with an on-prem DC. **Azure AD DS** also supports secure LDAP connections. Most of the time the LDAP connection to Azure AD DS will be initiated over the public internet. So, it is important to have encryption in place to prevent man-in-the-middle attacks.

In this section, I am going to demonstrate how to enable secure LDAP for Azure AD DS. Before we start, make sure you have the following prerequisites in place:

1. **A valid Azure subscription**
2. **A healthy Azure AD DS instance**
3. **A valid SSL certificate**

In this demo environment, the Azure AD DS instance is using the default Microsoft domain name `rebeladmlive.onmicrosoft.com`. So, I have to use a self-signed certificate for it as I cannot get a valid SSL certificate from a third-party SSL provider. If you have a valid certificate for your domain name, this step can be skipped. To generate a self-signed certificate, we can use the following PowerShell commands. This can be run from any PC:

```
$domainname="rebeladmlive.onmicrosoft.com"
$certlife=Get-Date
New-SelfSignedCertificate -Subject *.$domainname -NotAfter $certlife.
AddDays(365) -KeyUsage DigitalSignature, KeyEncipherment -Type
SSLServerAuthentication -DnsName *.$domainname, $domainname
```

In the above, replace `rebeladmlive.onmicrosoft.com` with your Azure AD DS instance name. As we can see, it created a wildcard SSL for rebeladmlive.onmicrosoft.com under the **Computer** account:

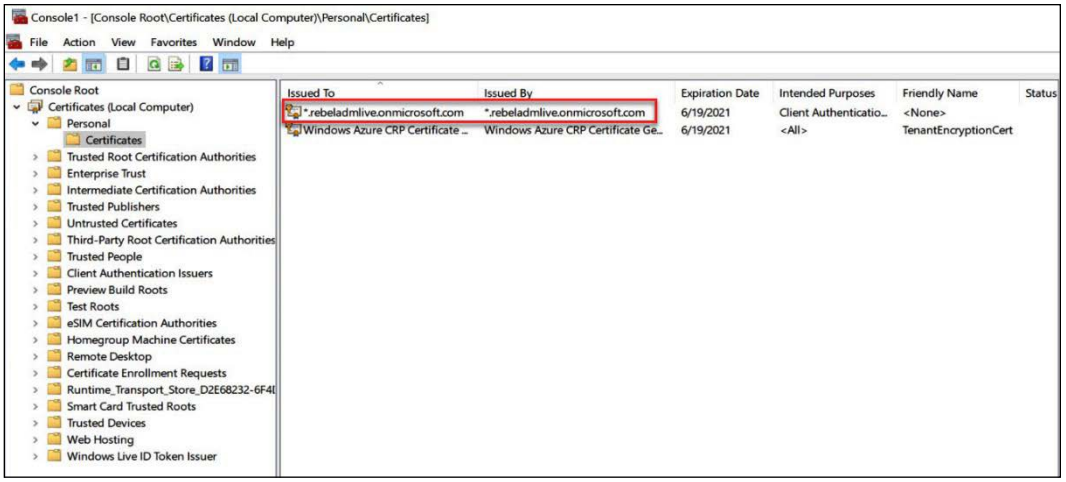


Figure 18.36: SSL certificate

We need to export this certificate as a .PFX file with a private key to:

1. Use it during the secure LDAP setup and upload it to Azure
2. Import it to any other PC that would like to initiate a secure LDAP connection (the certificate must be imported into `Computer Account\Personal\Certificates` as well as `Trusted Root Certification Authorities\Certificates`. This is because the certificate is a self-sign certificate and it doesn't have a trusted root certificate)

To export a certificate:

1. Right-click on the certificate and go to **All Tasks** | **Export...**:

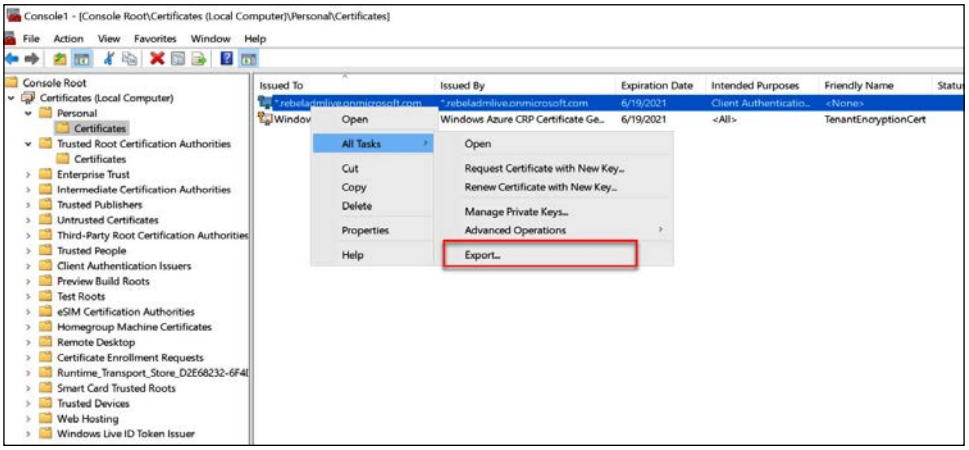


Figure 18.37: Export SSL certificate

2. It will open up the Certificate Export Wizard. Click on **Next** to start the process.
3. In the next window, select **Yes, export the private key** and click on **Next**:

Figure 18.38: Export private key

4. In the **Export File Format** window, match the following selection, and click on **Next**:

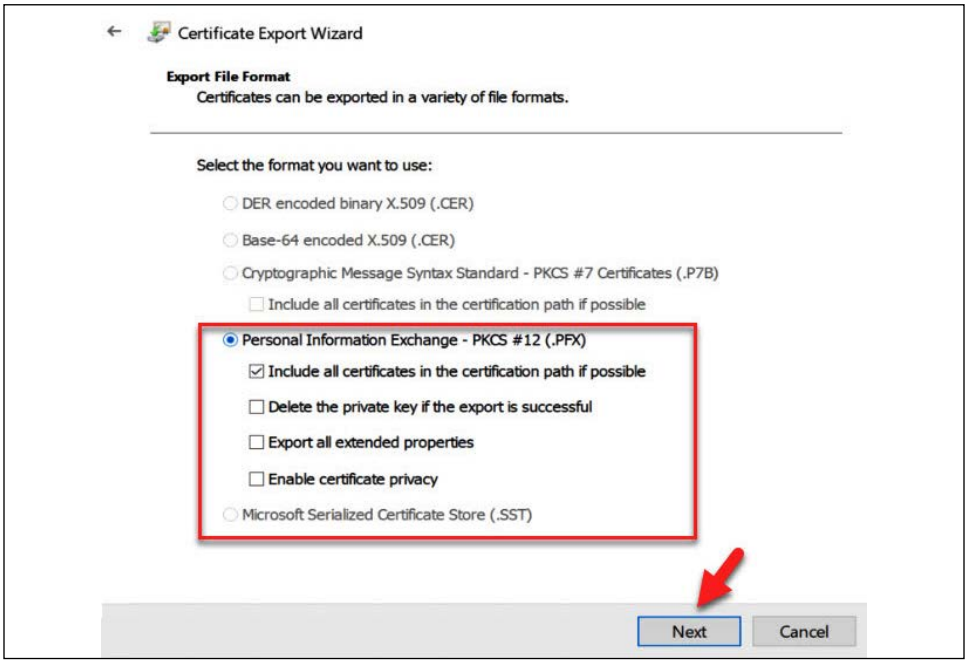


Figure 18.39: Export as PFX

5. In the next window, define the password for the export file and click on **Next**.
6. Then, select the file path for the output and click on **Next**.
7. In the next window, click on **Finish** to complete the export process.
8. Now we have the .PFX file. The next step is to import it again to `Trusted Root Certification Authorities\Certificates`. This is required as we are using a self-signed certificate for the exercise.

To do that,

1. Go back to certificate mmc and then `Trusted Root Certification Authorities\Certificates`.
2. Right-click and then go to **All Tasks** | **Import**.
3. It will open a new wizard for cert import. Click on **Next** to initiate the import process.
4. In the next window, select the PFX file we created, and click on **Next**:

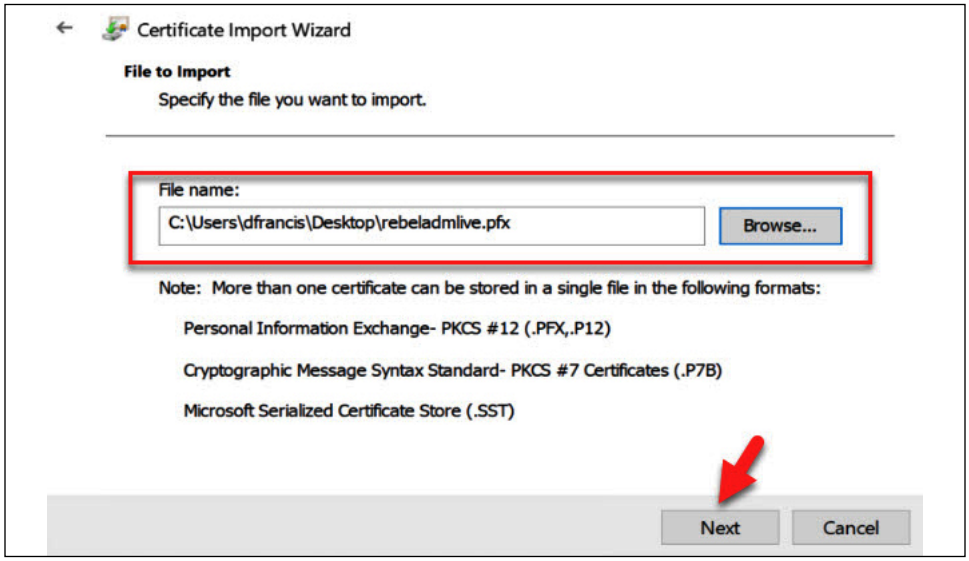


Figure 18.40: PFX file path

5. Then type the password we defined for the certificate file and click on **Next**.
6. In the certificate store window, leave the defaults and click on **Next**.
7. Then click on **Finish** to complete the import process.

8. Now we have a trusted self-signed certificate in place:

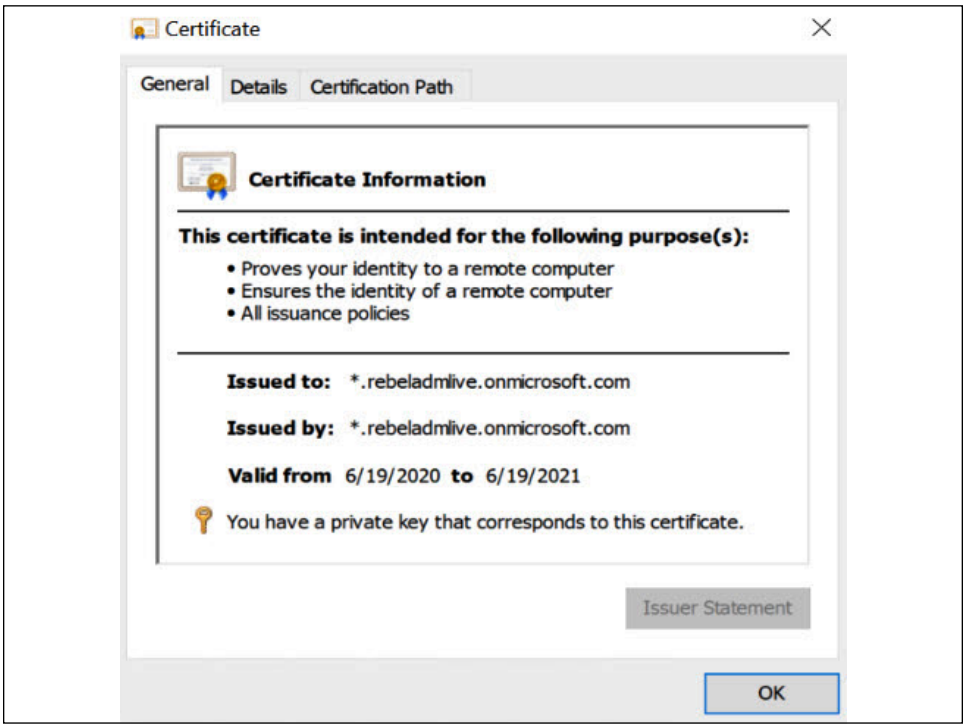


Figure 18.41: Certificate status

It is important to keep note of the certificate expiry date. If the certificate is renewed/reissued, we need to update the configuration of the service accordingly.

Enable secure LDAP (LDAPS)

The next step of the configuration process is to enable secure LDAP. To do that,

1. Log in to the Azure portal (`https://bit.ly/3oV9pKS`) as **Global Administrator**.
2. Then go to **Azure AD Domain Services**.
3. Click on the Azure AD DS instance.
4. It will open up the Azure AD DS settings page. On this page, click on **Secure LDAP**. It will open up a new window:

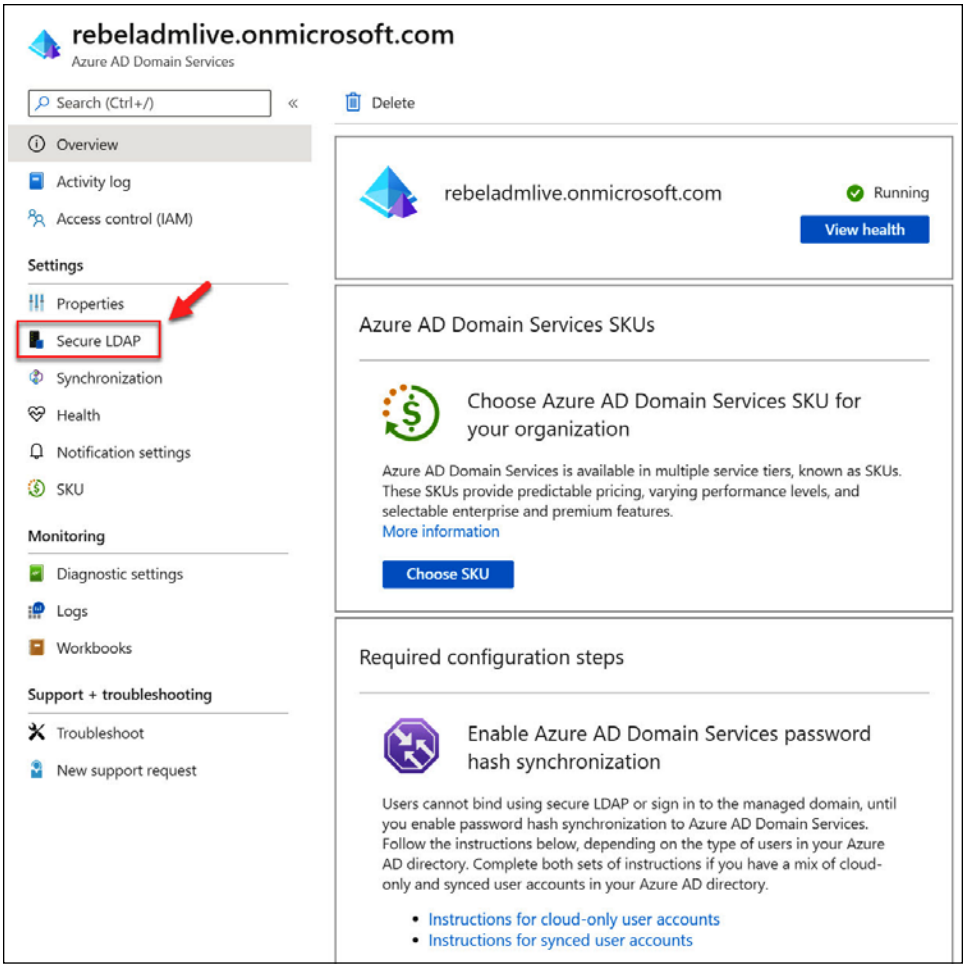


Figure 18.42: Secure LDAP feature

5. To enable secure LDAP, click on **Enable** under **Secure LDAP**. We also need to enable secure LDAP over the internet as in this demo I am going to access it via the public internet. To do that click on **Enable** under **Allow secure LDAP access over the internet**.

We also need to upload the PFX file we created. We can do that under the option for uploading aPFX file with a secure LDAP certificate. Click on the file icon and select the PFX file. Under the **Password to decrypt .PFX file** option, type the password for the PFX file. Finally, click on **Save** to apply changes:

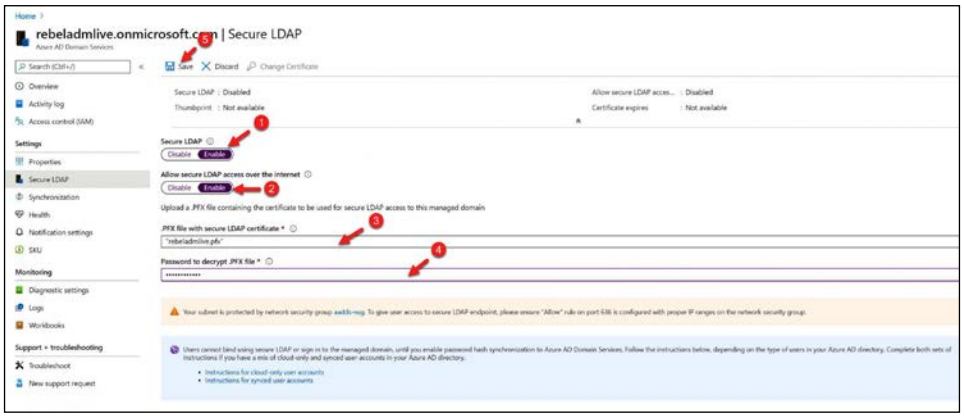


Figure 18.43: Configure secure LDAP

6. It will take a few minutes to enable secure LDAP. Once it is enabled, we can see a public IP is assigned for the secure LDAP communication. This info is available under the **Properties** page:

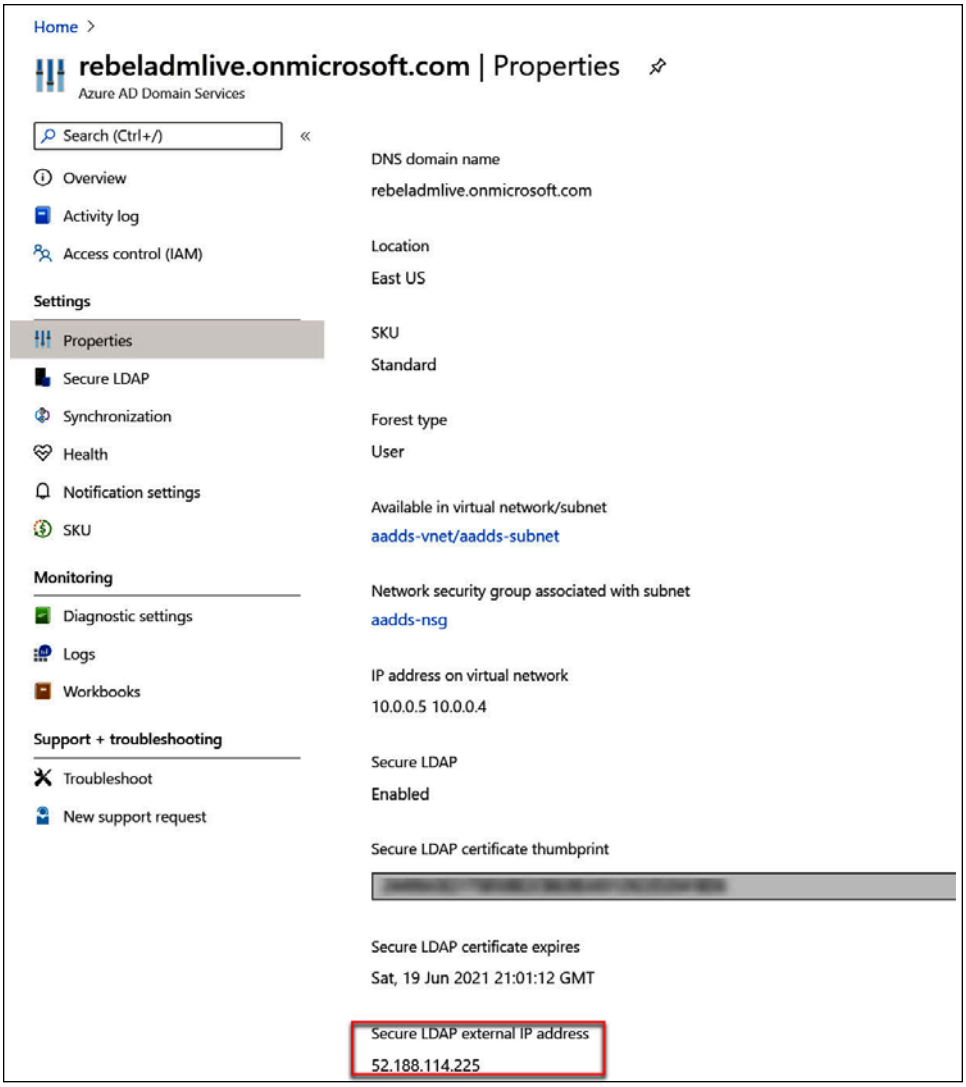


Figure 18.44: Public IP details

Allow secure LDAP traffic

Now we have the Azure AD DS instance with secure LDAP. The next step of the configuration is to allow secure LDAP traffic via an **network security group** (**NSG**).

It is recommended to use point-to-point communication rather than allowing a complete subnet, especially if it is over the public internet. In this demo, I have a VM behind public IP 52.136.127.107. Let's go ahead and create an NSG rule to allow secure LDAP communication.

To do that,

1. Go to the **Azure AD DS instance properties** page.
2. Click on the **NSG associated with the subnet**:

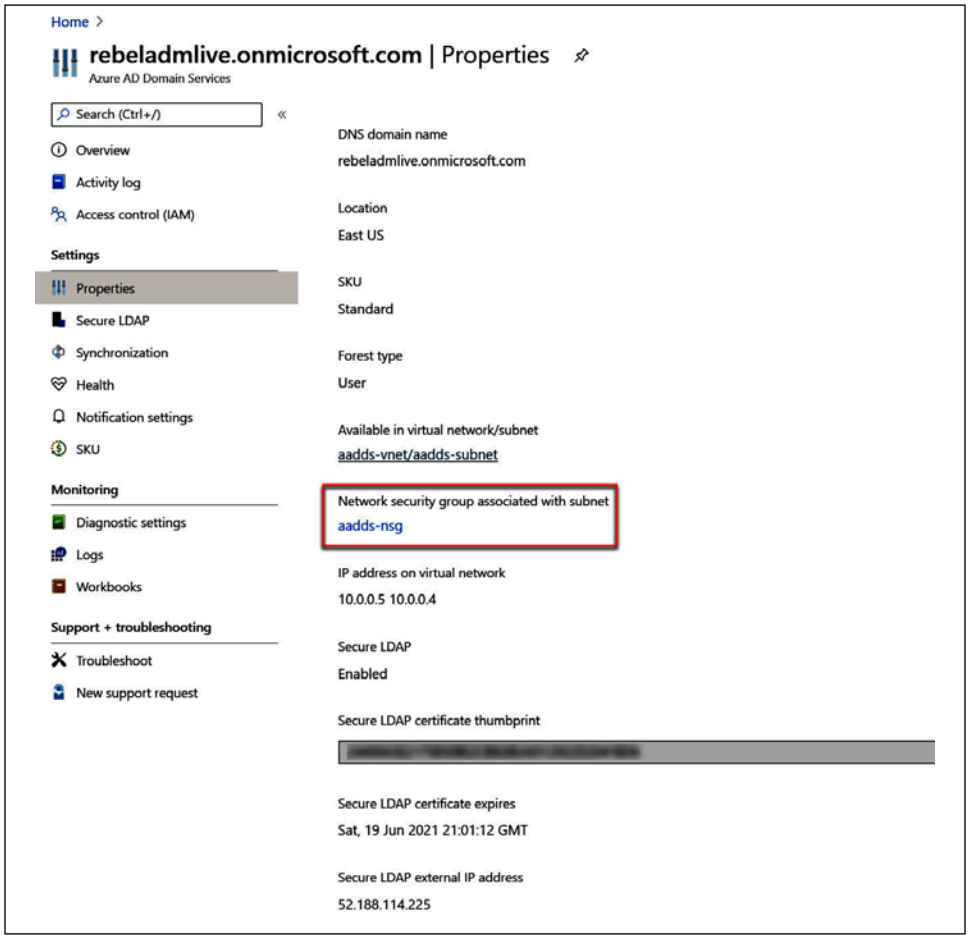


Figure 18.45: NSG details

3. On the NSG page, click on **Inbound security rules**.
4. Then click on **+Add** to create a new rule.
5. In the rule properties, allow **TCP port 636** access from `52.136.127.107`. Once the settings are in place, click on **Add** to create the rule:

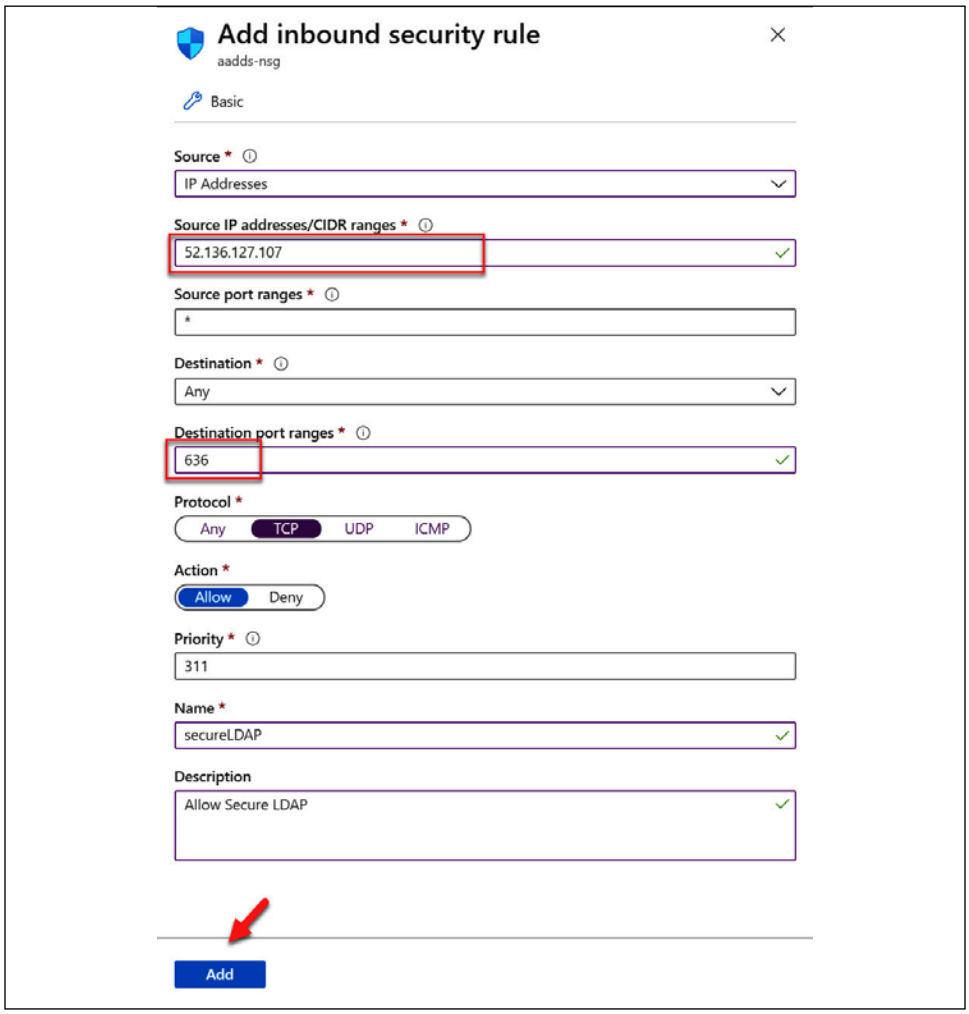


Figure 18.46: Create a new inbound security rule

Testing

It is time to test the secure LDAP access. To do that, first, we need to install RSAT tools: `https://bit.ly/3xgPTMu`. We need to use `ldp.exe` to test the connectivity.

Once tools are installed, we need to create a host entry (we can create DNS entries if required) to map the secure LDAP external IP address to a hostname.

In my demo environment, I created the following host entry:

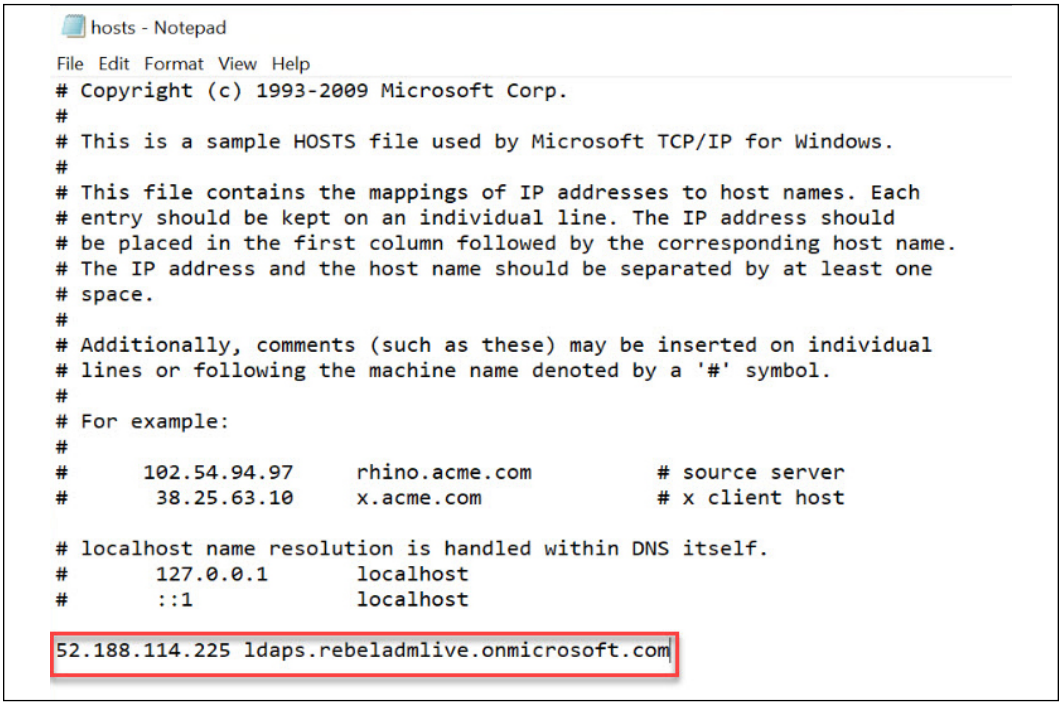


Figure 18.47: Host entry

Open `ldp.exe` and go to **Connections** | **Connect**. Then type `ldaps.rebeladmlive.onmicrosoft.com` in the **Server** field. Use `636` for **Port**. Also, select **SSL** for a secure LDAP connection. Once the values are in place click on **OK**:

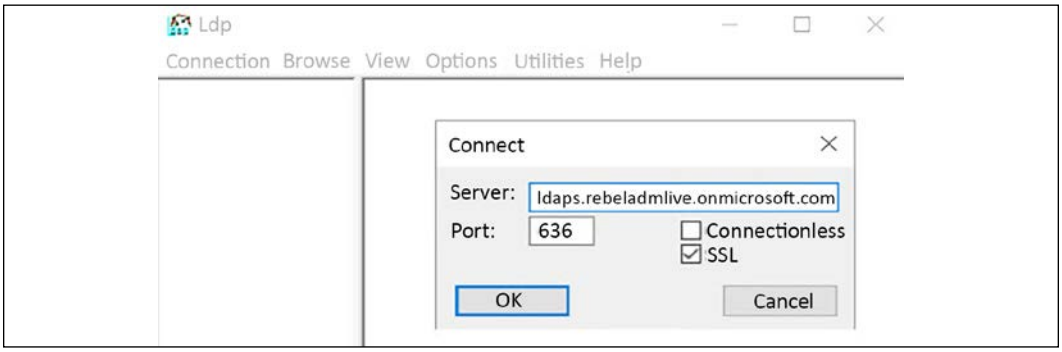


Figure 18.48: ldp.exe connection settings

As expected, I was able to connect successfully:

Figure 18.49: Secure LDAP connection status

Let's also try binding. To do that go to **Connections | Bind**.

You can't perform LDAP simple binds if you have disabled NTLM password hash synchronization on your Azure AD DS instance. You can use the following PowerShell commands to enable NTLM password hash synchronization for your Azure AD DS instance:

```
Login-AzAccount
$DomainServicesResource = Get-AzResource -ResourceType "Microsoft.AAD/
DomainServices"
$securitySettings = @{"DomainSecuritySettings"=@
{"NtlmV1"="Disabled";"SyncNtlmPasswords"="Enabled";"TlsV1"="Disabled"}}
Set-AzResource -Id $DomainServicesResource.ResourceId -Properties
$securitySettings -Verbose -Force
```

Then, in the connection window, use a user from **AAD DC Administrators**. As the domain, use the Azure AD DS instance name. For **Bind type**, choose the option **Bind with credentials**.

At the end select **OK** to bind:

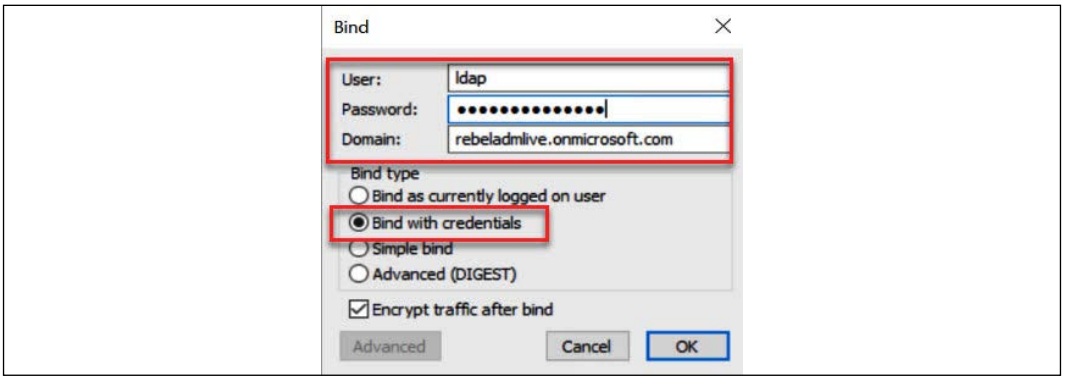


Figure 18.50: Bind with credentials

On a successful bind, you will see a message like below:

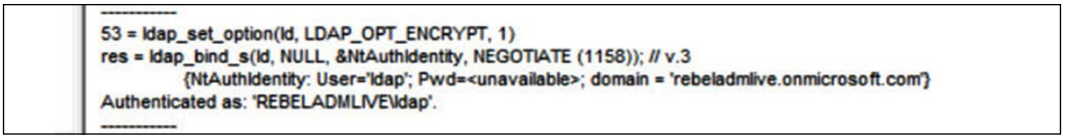


Figure 18.51 : Bind status

Then go to **View | Tree**. Leave **BaseDN** empty and click **OK**.

In the ldp window you can now see the data. Expand `OU=AADDC Users,DC=xxxxxxx,DC=onmicrosoft,DC=com` and see if you can see the user accounts under it:

Figure 18.52: Browse data

As we can see we were able to initiate a secure LDAP connection to Azure AD DS successfully. In the next section, we are going to look into Azure AD DS replica sets.

Azure AD DS resiliency with replica sets

In an AD environment, we keep additional DCs to improve resiliency. In this way, if one DC fails it will not make a big impact. We can further improve the resiliency of infrastructure by keeping an additional DC and mission-critical servers in a different location. So, in the event of a site failure, we will still have a DC and mission-critical servers running on a remote location. When we create the Azure AD DS managed domain, we provide a unique domain name. In the back end, Azure will deploy two DCs with this unique domain in your selected Azure region. This setup is called a **replica set**. Now, to improve resiliency, we can create additional replica sets in other Azure regions.

Each replica set uses a virtual network. All these virtual networks must be peered to create a mesh network to support the replication between replica sets. Azure VNet peering allows connecting virtual networks seamlessly via Azure backbone infrastructure. This is similar to inter-VLAN routing in on-prem networks. VNet peering can be used to connect virtual networks in the same Azure region or different Azure regions. If it is between regions, we call it **Azure global VNet peering**.

Azure AD DS replica sets have the following characteristics:

- Each replica set contains the same data. You can't use different domains for different replica sets.
- All replica sets will be placed in the same AD site. Because of that the replication between replica sets is faster.
- All replica sets should be created under the same subscription. You cannot have replica sets between different subscriptions.
- Replica sets only ensure the availability of the authentication services. But to use the full benefits of it, you need to consider how your Azure VMs and applications will work during a site failure.

In this section, I am going to demonstrate how to create a replica set of an existing Azure AD DS managed domain.

In my current demo setup, I already have an Azure AD DS managed domain configured. It is using the REBELVN1 virtual network and aadds-subnet:

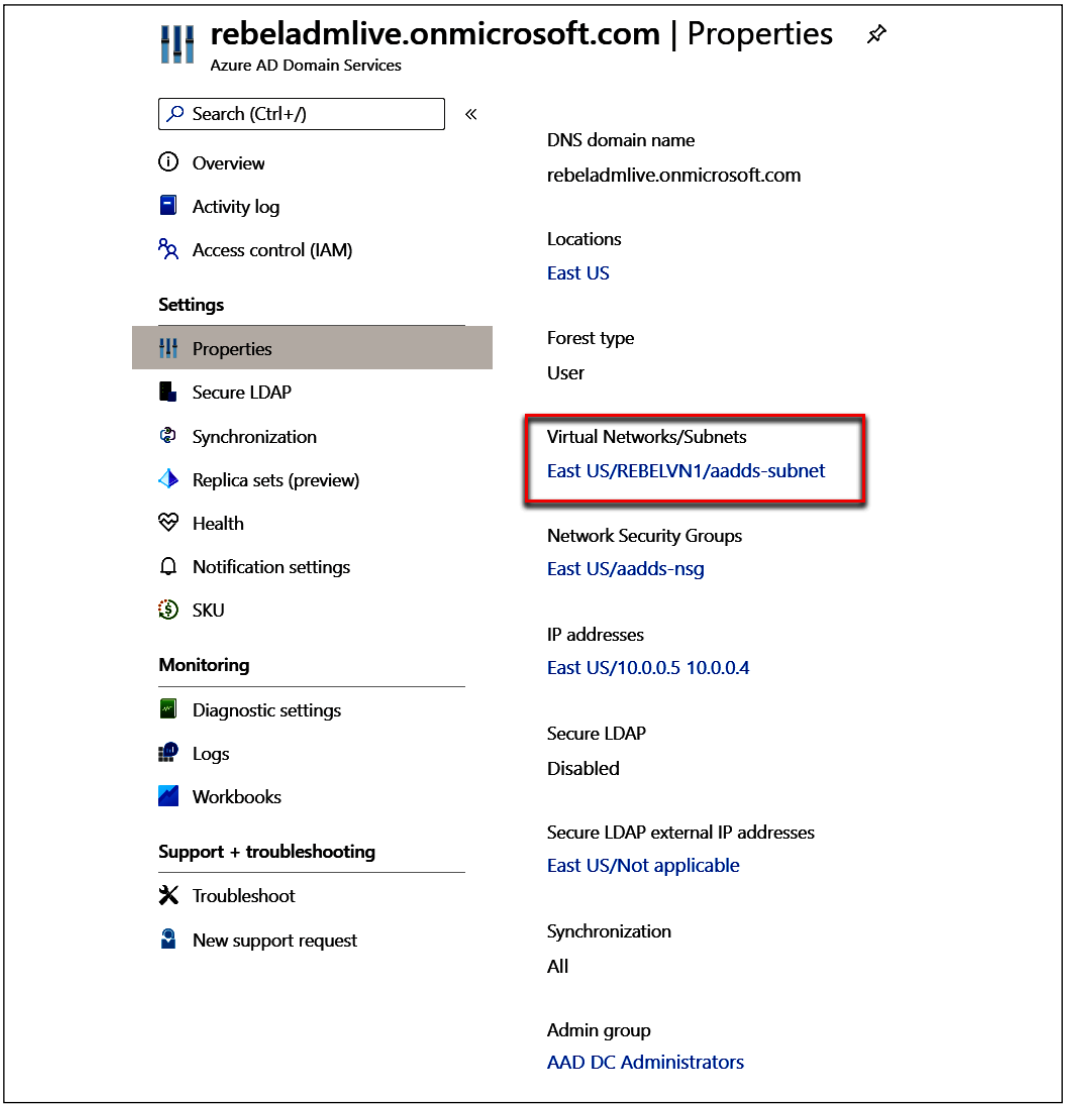


Figure 18.53: Virtual network details

The **REBELVN1** virtual network is set up under the **REBELRG1** resource group. This resource group is using the `East US` Azure region. The virtual network is using the `10.0.0.0/16` address space. It has two subnets:

- `vmsubnet` – 10.0.2.0/24 for VMs
- `aadds-subnet` – 10.0.0.0/24 for the Azure AD DS managed domain

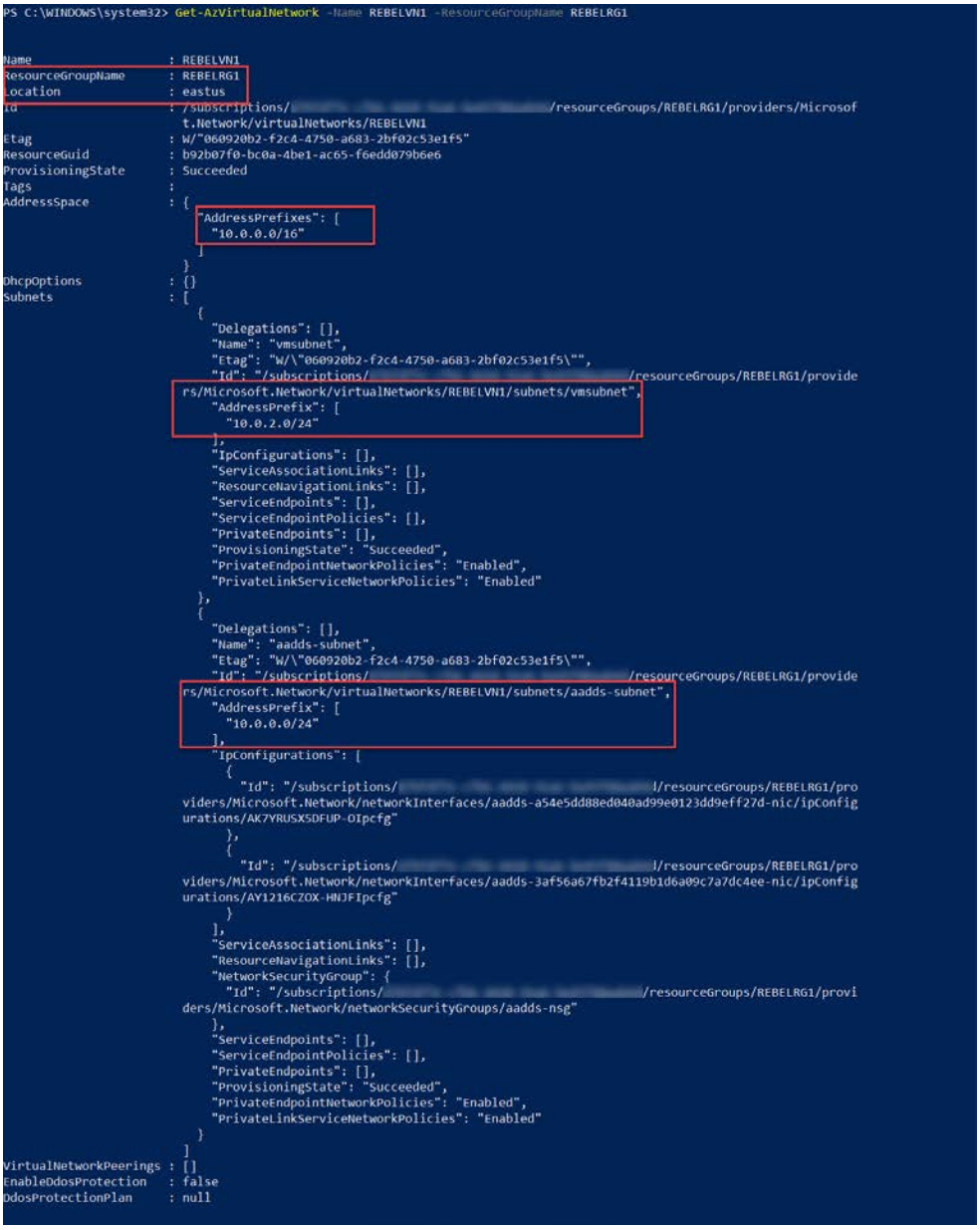


Figure 18.54: Current configuration of the virtual network

In this demo, I am going to create a new virtual network in the `West US` Azure region. Later this will be used to host the additional replica test.

Set up a new resource group for an additional replica set

As the first part of the configuration, I am going to create a new resource group in the `West US` Azure region.

For the configuration process, I will be using PowerShell. Therefore, please make sure you have an Azure PowerShell module installed. More info about it can be found here: `https://bit.ly/30WK0rS`:

1. Launch the PowerShell console and connect to Azure using **Connect-AzAccount**
2. Create a new resource group using **New-AzResourceGroup -Name REBELDRRG1 -Location "West US"**

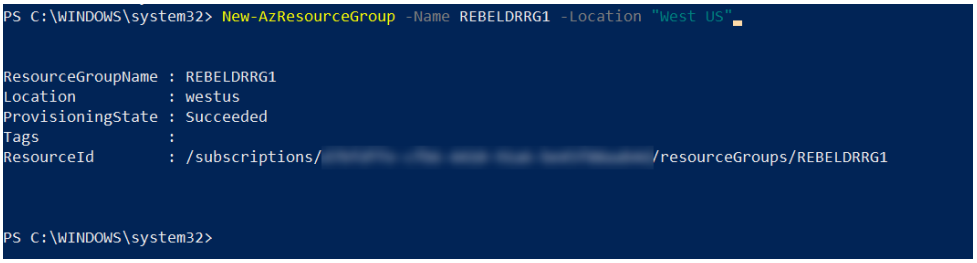


Figure 18.55: Create a resource group

In the above, `REBELDRRG1` is the resource group name and it is created in the Azure `West US` region.

Set up a new virtual network for an additional replica set

The next step is to create a new virtual network under the `REBELDRRG1` resource group:

```
$drvmsubnet = New-AzVirtualNetworkSubnetConfig -Name drvmsubnet
-AddressPrefix "10.1.3.0/24"
```

```
New-AzVirtualNetwork -Name REBELDRVN1 -ResourceGroupName REBELDRRG1
-Location "West US" -AddressPrefix "10.1.0.0/16" -Subnet $drvmsubnet
```

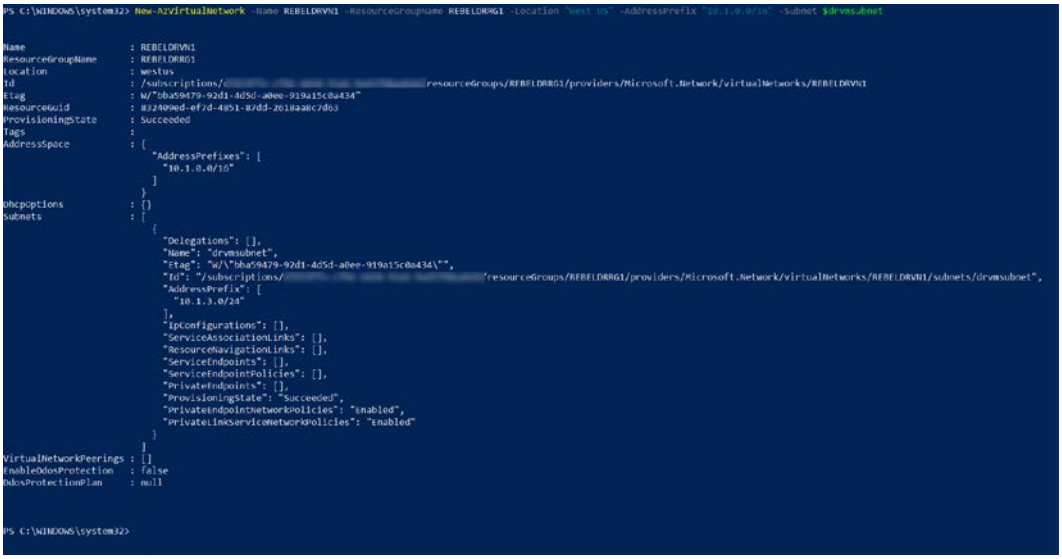


Figure 18.56: Set up a new virtual network for a replica set

In the above, `REBELDRVN1` is the new virtual network name. It has the `10.1.0.0/16` address space. It also has a new subnet, `10.1.3.0/24 (drvmsubnet)`, for VMs.

Set up global VNet peering between two virtual networks

Replica sets can only be created between peered networks. So, I am going to create global VNet peering between two virtual networks. I have written a blog post about global VNet peering. I recommend you read it before proceeding further so you have a better understanding of how it works:

Step-by-Step Guide: How to setup Azure Global VNET Peering? (PowerShell Guide): `https://bit.ly/32kurL6`

As the first step of the configuration, I am going to create peering from `REBELVN1` to `REBELDRVN1`:

```
$vnet1 = Get-AzVirtualNetwork -Name REBELVN1 -ResourceGroupName
REBELRG1
$vnet2 = Get-AzVirtualNetwork -Name REBELDRVN1 -ResourceGroupName
REBELDRRG1
```

```
Add-AzVirtualNetworkPeering -Name REBELVN1toEBELDRVN1 -VirtualNetwork
$vnet1 -RemoteVirtualNetworkId $vnet2.Id
```

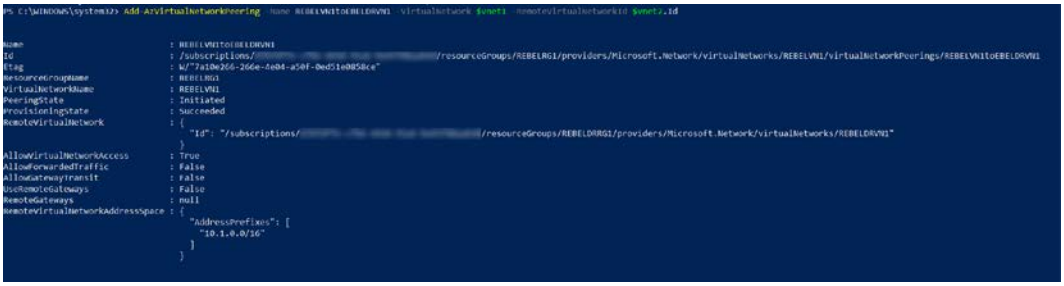


Figure 18.57: Global VNet peering

The above creates peering from `REBELVN1` to `REBELDRVN1`. We need to do the same from `REBELDRVN1` to `REBELVN1`:

```
$vnet1 = Get-AzVirtualNetwork -Name REBELVN1 -ResourceGroupName
REBELRG1
$vnet2 = Get-AzVirtualNetwork -Name REBELDRVN1 -ResourceGroupName
REBELDRRG1
Add-AzVirtualNetworkPeering -Name REBELDRVN1toREBELVN1 -VirtualNetwork
$vnet2 -RemoteVirtualNetworkId $vnet1.Id
```

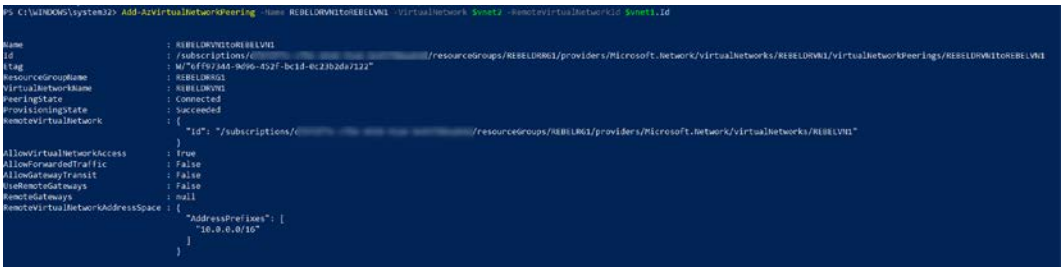


Figure 18.58: Global VNet peering

This completes the peering configuration.

Create an Azure AD DS managed domain replica set

Now we are ready to create the new replica set. To do that,

1. Log in to the Azure portal as Global Administrator.

2. Then search for **Azure Active Directory Domain Services** and click on it.
3. On the **Azure Active Directory Domain Services** page, click on the domain.
4. On the Properties page click on **Replica sets**.
5. There we can see the existing replica set. To add a new one, click on **+ Add**:

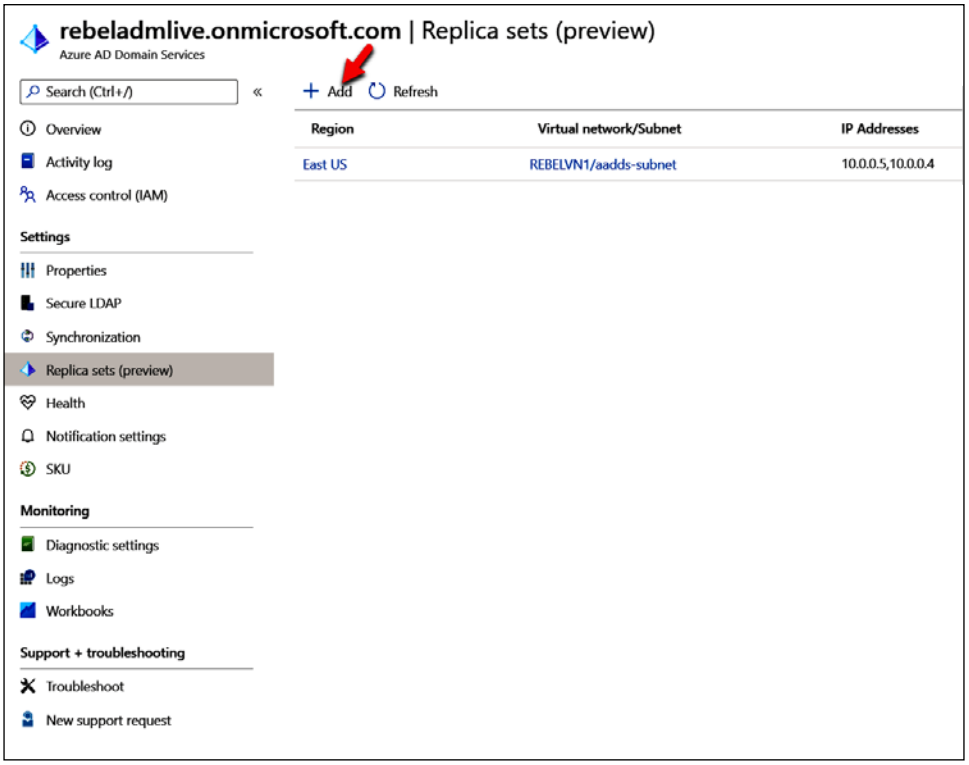


Figure 18.59: Add replica set

6. On the configuration page, select **REBELDRRG1** as a resource group. The region should be set to **West US**. For the virtual network select **REBELDRVN1**. For the subnet select the subnet we created in one of the previous steps.

You also can add a new subnet. At the end click on **Save** to complete the configuration:

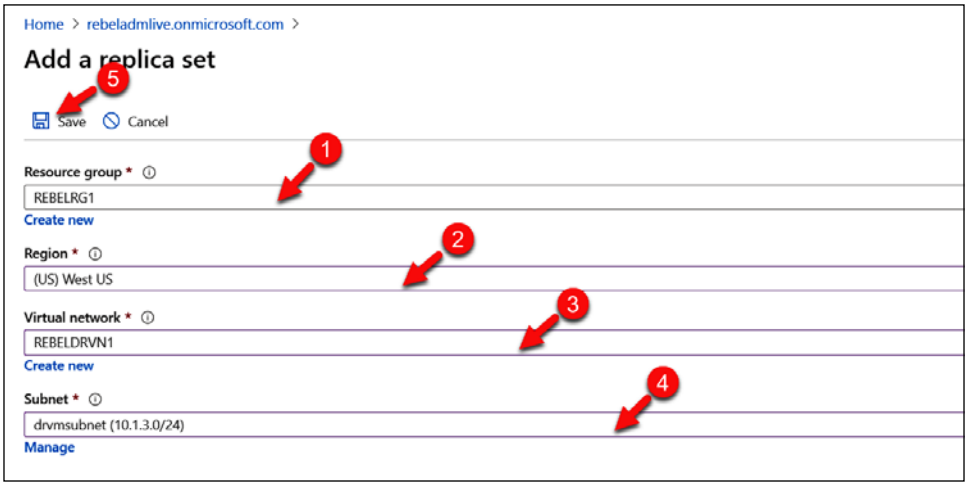


Figure 18.60: Configure a replica set

7. Then you can see Azure start provisioning relevant services. This will take some time to complete:

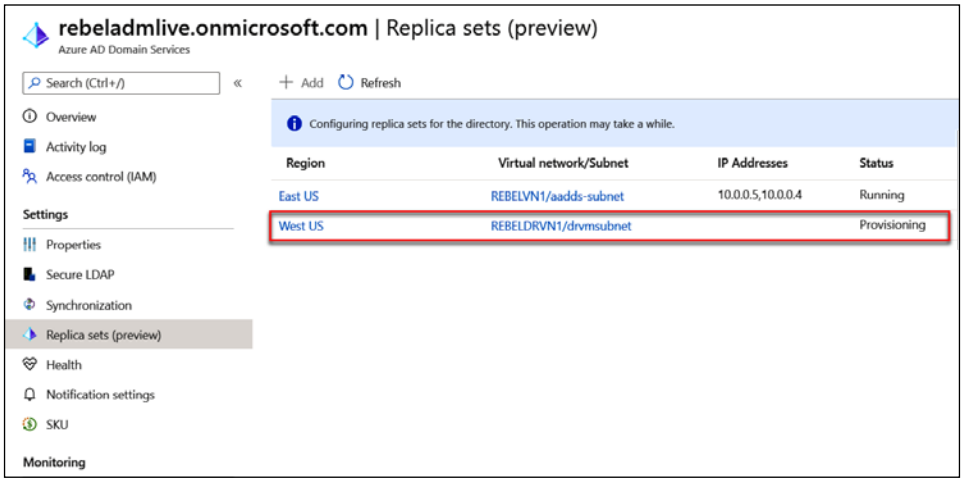


Figure 18.61: Replica set provisioning

8. Make sure the deployment completes without errors:

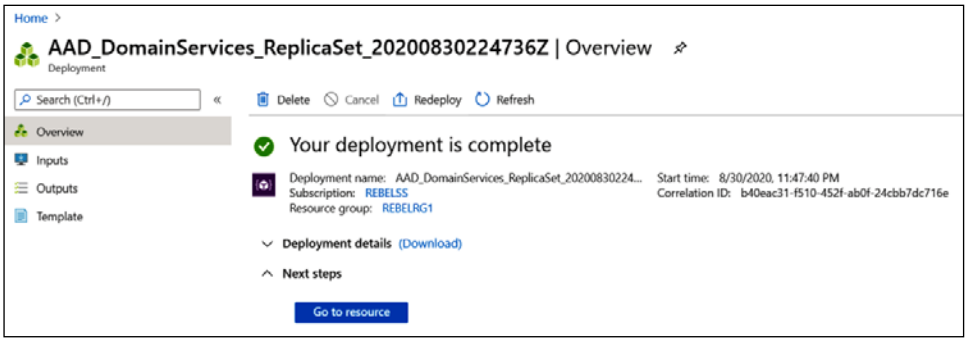


Figure 18.62: Status of the provisioning task

In the end, we have a new replica set running in the `West US` region:

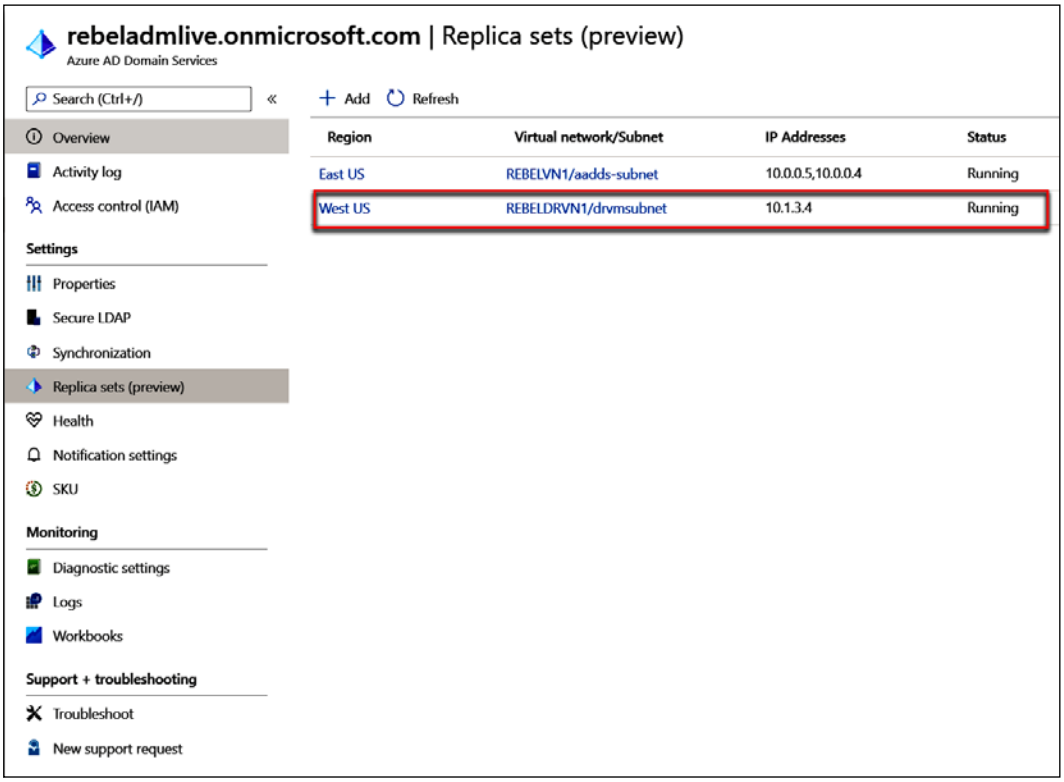


Figure 18.63: Status of the replica set

However, the replica set can only ensure availability within the region it is running on. If you have systems and applications that rely on Azure AD DS and require global redundancy, those systems and applications should also reside in the same region as the replica set.

At the time of writing this section, the replica set feature was in preview. But now it is generally available. So, it can be used in production environments now.

Summary

Azure AD is a Microsoft-managed, cloud-based, and multi-tenant directory service. It can be used in a cloud-only infrastructure or in a hybrid infrastructure. When used in a hybrid infrastructure, it allows us to use the same identities so that we can work with resources on-prem and in the cloud. It extends local AD infrastructure functionalities to the cloud.

In this chapter, we learned what Azure AD DS is and what its capabilities are. We also looked into the different types of sign-in options that we can use in a hybrid setup, including password hash synchronization, pass-through authentication, and SSO. After that, we looked at a step-by-step guide for integrating our on-prem directory service with Azure AD. We also learned how to set up secure LDAP with an Azure AD managed domain and how to improve the resiliency of the Azure AD managed domain by using replica sets. In this chapter, I was only able to demonstrate a very limited number of features and capabilities of Azure AD. For more Azure AD-related topics, refer to my blog at `https://bit.ly/3HPFCvv`.

In the next chapter, we will look at AD auditing and monitoring, which is crucial for maintaining a healthy AD infrastructure.

19

Active Directory Audit and Monitoring

Microsoft SQL Server is a database server. If I need a database server for a project, I can simply spin up a server and install Microsoft SQL Server on it and then use it to store data. In this situation, my requirement is static. As long as I am using the software/service, I need a working database server. But when it comes to cyber security, the requirements are not static – they change very often. There are new bugs and security threats found every day. A security solution we use today may not work well against new threats found in the future. The only possible way to address this challenge is to "continuously improve" the solutions in place. To do this, we can use the following steps as guidelines. These four steps are connected to each other and work as a life cycle rather than one-time process:

1. Identify
2. Protect
3. Detect
4. Respond

Before we come up with a solution, we first need to "identify" what to protect. The tools and services we can use to protect identities are different from what we can use to protect applications. Also, we need to be mindful when choosing where to make investments. As an example, we need to invest more to protect sensitive business data than protecting five-year-old pictures from a marketing campaign. Once we identify what we need to protect, then we can use the relevant tools and services to "protect" it.

One of the pillars of the Zero-Trust security approach is "assume a breach". We can't close all the doors. We need to assume a breach has already happened. But the important thing is that when it happens, there should be a way to "detect" it. If the solution or services in place are not in a healthy state, we need to detect those events as well. Once we detect a breach or issue, then we can "respond" in a timely manner. Each step through this life cycle is critically important. However, "detect" has more weight as we are not only looking to identify a breach, we can also use that information to confirm whether the solution in place is working as expected. This is why auditing and monitoring are important.

Before I use the London Underground service, I always check the **Transport for London** (**TfL**) website to see the status of the tube line services. This is a service provided by TfL to make sure its users are planning their journeys properly to avoid delays. The system TfL has in place to monitor line statuses gives us two benefits. As a service provider, TfL can detect problems and start to address them immediately. At the same time, some filtered information is passed to the public that is important for planning their journey.

Similarly, auditing and monitoring are not only there to enable engineers to find problems. They should also provide filtered, structured information that is important to the roles and responsibilities of different parties in the business. As an example, the IT manager would like to know the overall domain service availability over the last month, but it is not important for them to know each and every event that happened in the system during the last 30 days, although this can be important for the IT engineers.

In auditing and monitoring, we also need to identify what to monitor and what should be reported. Knowing each and every thing that happens in the system is good, but at the same time, unless it has been analyzed and prioritized, it will not deliver any benefit to engineers in detecting issues properly. Therefore, we need systems to audit and monitor the correct things and present it in a useful way.

Active Directory (**AD**) itself is not a security solution, but in previous chapters, I explained how we can use the features of AD to protect identities. So, it is important for us to monitor the health of AD.

In this chapter, we will look at the following:

- Monitoring **Active Directory Domain Services** (**AD DS**) related events and logs
- Advanced auditing for AD infrastructure
- Using **Microsoft Defender for Identity** to monitor identity infrastructure threats

- Azure AD Connect Health

Before we look into the other tools and services we can use to monitor AD, let's see what the built-in Windows tools can provide.

Auditing and monitoring AD using built-in Windows tools and techniques

Microsoft offers built-in features and tools to monitor and audit AD environments. In this section, we are going to review these features and tools.

Windows Event Viewer

As an engineer, I am sure you are well aware of Windows Event Viewer. It is a built-in tool that can be used to view and filter event logs on a local or remote computer. The events shown there are generated by the operating system, services, server roles, and applications. This is the most commonly used tool in Windows systems for auditing and troubleshooting purposes.

We also can write custom events to event logs. This is useful if you plan to run a script or action based on a particular event ID. This can be done by using the `Write-Eventlog` cmdlet.

As shown in the following screenshot, Windows Event Viewer (Local) has four different categories to group event logs:

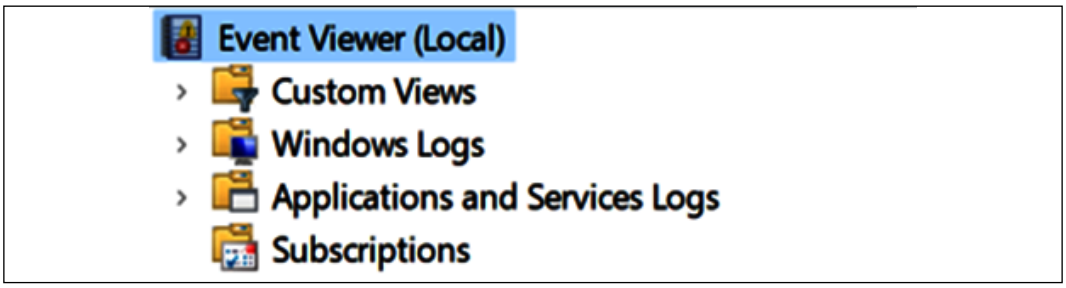


Figure 19.1: Windows Event Viewer

In the preceding list, **Windows Logs** and **Application and Service Logs** both have additional predefined subcategories.

Custom Views

Event Viewer allows the creation of **Custom Views** based on event level, time, log type, source type, event ID, task category, keywords, users, or computers. Event Viewer catches thousands of different events. Using Custom Views, we can filter events and access the information we need. All these custom-made views will be listed under the Custom Views section. It also has predefined custom views.

These predefined custom views are based on the server roles. When AD DS roles are added, it also creates a custom view in the event log to group all the AD DS-related events, as shown in the following screenshot:

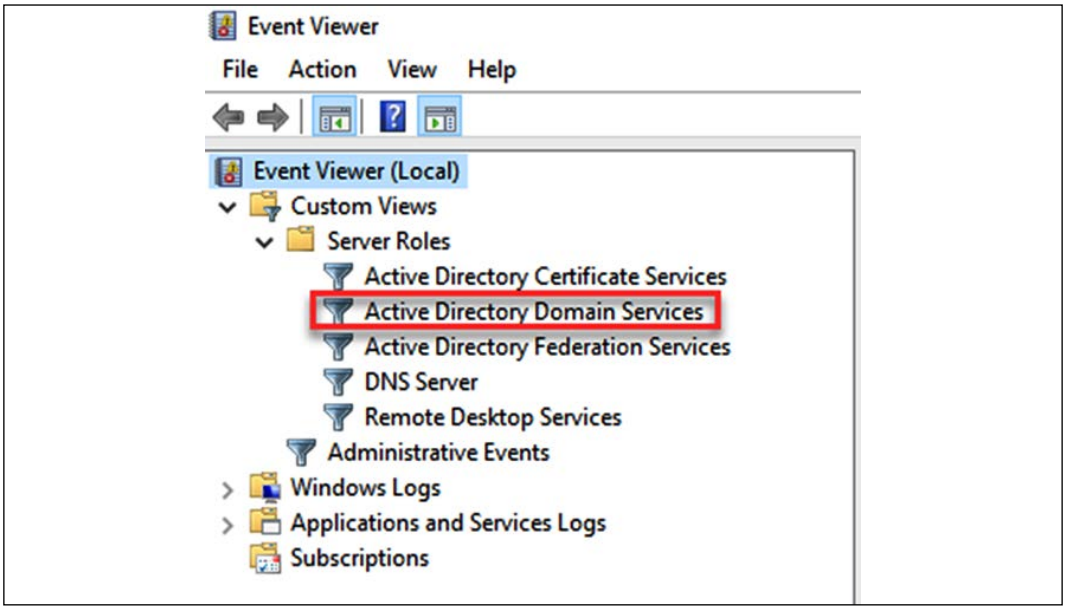


Figure 19.2: Windows Event Viewer Role logs

The data that appears in these custom views will also be available on other logs such as Windows Logs.

Windows Logs

The Windows Logs section includes five Windows log files. These mostly contain OS-related events:

- **Application log**: This log contains the events collected from various applications running on the system. These events can be from Microsoft or any other application.

- **Security log**: This log includes events such as successful and failed system login attempts. Engineers can specify which security events need to be recorded using audit policies.
- **Setup log**: This includes events related to application setup and adding/removing server roles.
- **System log**: This log includes events related to Windows system components. As an example, an event related to an automatic service start failure will be listed under this log.
- **Forwarded Events**: Using Event Viewer, we can connect to another remote computer and view the events. However, it may be required to watch for specific events from multiple sources. As an example, let's assume we need to collect events with ID `4321` from three computers. Using the Subscriptions event, we can collect only those events and forward them to the Forwarded Events log.

Forwarded Events is the default location to push subscribed events. However, if needed, these events can also be forwarded to other log files.

Applications and Services Logs

The Application and Services Logs category was introduced after Windows Server 2008. This stores the events related to applications and their components. Most of the events listed here are more suited for application developers doing debugging and application-level troubleshooting.

This category has four log types:

- **Admin**: Events listed in this log are understandable by end users and IT professionals. This information can be used for basic application troubleshooting. Most of these log entries will include instructions or links to knowledge base articles from the application vendor to find out more about the given issue or to fix it.
- **Operational**: Operational events include information about configuration or status changes of an application/service. These events are useful for application diagnosis.
- **Analytic**: This log is hidden and disabled by default. This is usually only enabled during the application or service diagnosis process as this generates a high volume of events.

- **Debug**: This is purely used for troubleshooting purposes by application developers and vendors. Similar to the Analytic log, it is, by default, hidden and disabled.

As mentioned before, we also can see events data from remote locations by using subscriptions. Let's see what subscription data includes.

Subscriptions

This category lists down the event subscriptions created with remote computers. Here, we can create/edit/disable event subscriptions, check the runtime status, and forcibly run subscription jobs.

When we open up an event, it gives different levels of information, such as the following:

- A general description about the problem
- The log file name
- The event source to indicate where it came from
- The event ID number
- The level of the error (critical, information, or warning)
- The username of the error owner
- Links to TechNet, KB, or other sources to get more information about the event
- The time of the event
- Hostname of the source computer

In the previous section, I explained Applications and Services Logs and the types of data available under it. There are a few different Applications and Services Logs related to the AD service.

AD DS event logs

Apart from the events under the Windows Logs category, AD DS and related service events can be found under the following logs. These are located under the Applications and Services Logs category:

- AD Web Services
- DFS Replication

- Directory Service
- DNS Server
- File Replication Service (only if using FRS)

Apart from events, AD DS and related services have other system log files that record data about service installation/uninstallation, performance, service errors/failures, and so on.

AD DS log files

These log files can be used for auditing, troubleshooting, or debugging purposes.

The default location for these log files is `%SystemRoot%\Debug`:

- `DCPromo.log`: This log file is created during the AD promotion process. It also records events during the demotion process. This log contains events such as the following:
    - AD DS configuration settings
    - Information about schema preparation
    - Information about directory partition creation/modifications
    - Information about data replication
    - Service configuration status
    - Information about creating AD databases and the `SYSVOL` directory
- `DCPromoUI.log`: This log file can be considered as a progress report for the AD DS promotion/demotion process. It starts the logging process as soon as the AD DS configuration wizard opens, and ends when it completes the installation successfully (until the reboot request is accepted) or when it is aborted due to errors. This includes the results of each and every act of the system during the service installation and removal process. This log includes useful information, such as the following:
    - A timestamp for when the installation or removal process started
    - Detailed results of each validation test
    - The name of the domain controller used for the initial replication
    - A list of directory partitions that were replicated
    - The number of objects replicated in each and every partition
    - Configuration summary
    - Information about registry key changes related to configuration

- `DFSR.log`: This log file includes events related to DFS replication. This can be used for `SYSVOL` replication troubleshooting and the debugging process (if `SYSVOL` uses DFS replication).

After Windows Server 2008, AD uses DFS replication by default, but if domain controllers have been introduced to a Windows Server 2003 environment, it will use FRS by default.

AD audit

The only way to identify potential security threats and security breaches in infrastructure is through continuous monitoring and auditing. When it comes to auditing, the Windows system itself provides advanced auditing capabilities to identify such security issues. However, by default, only certain types of actions are audited. These auditing settings are handled by Windows audit policies.

Here, we are only going to look at advanced security audit policies, which were first introduced with Windows Server 2008 R2.

There are 10 categories of events we can audit in a Windows system:

- System events
- Logon/logoff events
- Object access events
- Privilege use events
- Detailed tracking events
- Policy change events
- Account management events
- **Directory Service** (**DS**) access events
- Account logon events
- Global object access auditing

Each and every event category also has subcategories.

Legacy Windows auditing provides nine categories and each category also has subcategories. These are located under **Computer Configuration | Windows Settings | Security Settings | Local Policies | Audit Policies**. Also, categories and subcategories can be listed using `auditpol /get /category:\*`. Advanced security audit policies provide 53 options to tune up the auditing requirements, and you can collect more granular-level information about your infrastructure events than by legacy auditing.

Auditing on these categories can be enabled using group policies. These are located under **Computer Configuration | Windows Settings | Security Settings | Advanced Audit Policy Configuration | Audit Policies**, as shown in the following screenshot:

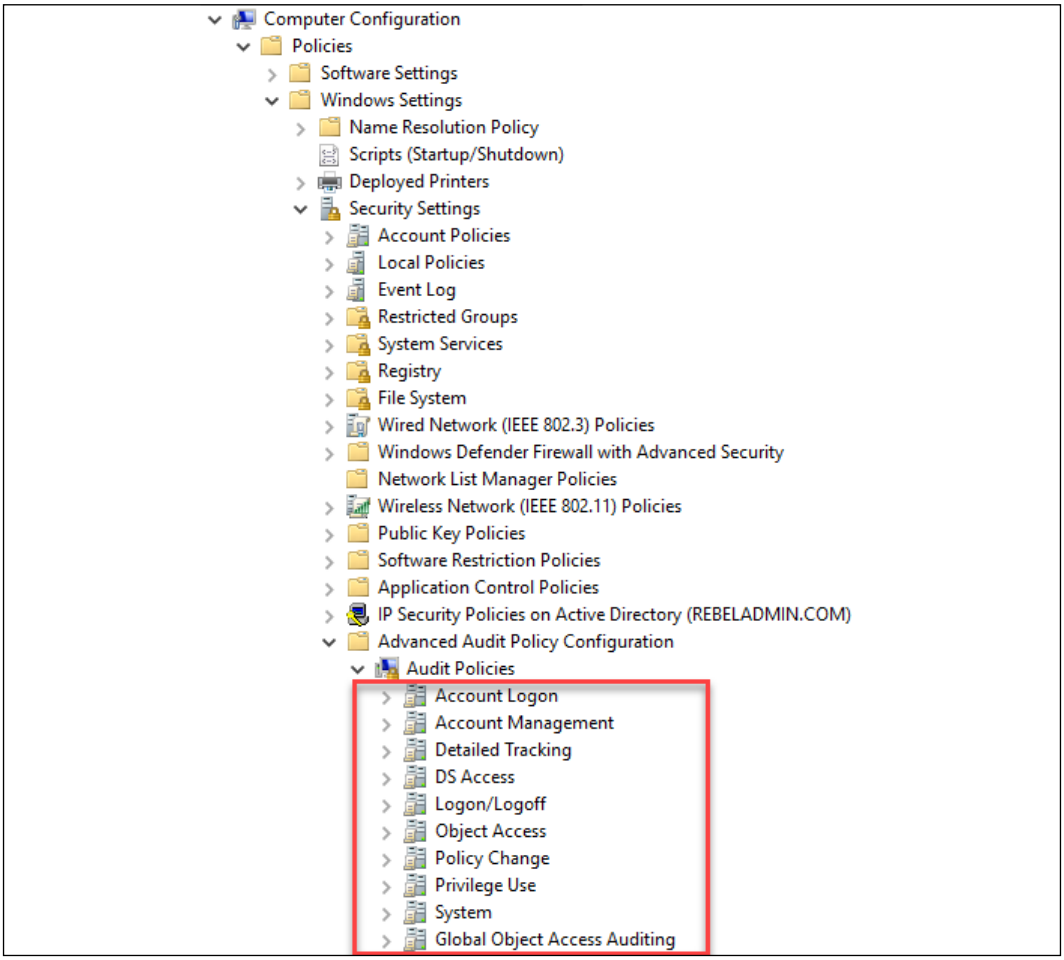


Figure 19.3: Advanced audit logs

All these categories can collect a lot of system and service events related to AD infrastructure activities. But in this section, we are only going to focus on the DS Access events category and its subcategories. It audits events related to AD objects access and AD object modification. Also, the settings under this category only apply to domain controllers.

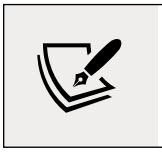
The DS Access events category includes four subcategories:

- Audit Directory Service Access
- Audit Directory Service Changes
- Audit Directory Service Replication
- Audit Detailed Directory Service Replication

As our next step, let's explore each of the above subcategories in more detail.

Audit Directory Service Access

This category records events when an AD DS object is accessed. This will only work if the **system access control list** (**SACL**) is configured and the relevant objects have been added. This is similar to directory service access in legacy auditing.



SACL allows engineers to log access attempts to secured objects. SACL can generate audit records when an access attempt fails, when it succeeds, or both.

When auditing is enabled under this category, the following event can be found under security logs:

Event ID	Event message
4662	An operation was performed on an object

Audit Directory Service Changes

This category records events related to AD DS object changes, such as the following:

- Create
- Delete
- Modify
- Move
- Undelete

When an object value is changed, it records the old value and the new value it was changed to. Again, the event will only be generated for the entries listed under SACL. Once auditing is enabled, the following events can be found under security logs:

Event ID	Event message
5136	A directory service object was modified.
5137	A directory service object was created.
5138	A directory service object was undeleted.
5139	A directory service object was moved.
5141	A directory service object was deleted.

Audit Directory Service Replication

This category logs events when replication between two domain controllers begins and ends. When auditing is enabled, we will be able to find the following events in the logs:

Event ID	Event message
4932	Synchronization of a replica of an AD naming context has begun.
4933	Synchronization of a replica of an AD naming context has ended.

Audit Detailed Directory Service Replication

This category records detailed information about data replicated between domain controllers. Once auditing is enabled, it will generate a high volume of events and will be useful for troubleshooting replication issues. It will log the following types of events:

Event ID	Event message
4928	An AD replica source naming context was established.
4929	An AD replica source naming context was removed.
4930	An AD replica source naming context was modified.
4931	An AD replica destination naming context was modified.
4934	Attributes of an AD object were replicated.
4935	Replication failure start.
4936	Replication failure end.
4937	A lingering object was removed from a replica.

Demonstration

In this section, let's go ahead and see how we can use built-in Windows monitoring and audit capabilities. In order to do these configurations, you need to have domain administrator or enterprise administrator privileges.

Reviewing events

Event Viewer can simply be opened by running `eventvwr.msc`. The same MMC can also be used to connect to a remote computer using the **Connect to Another Computer...** option, as highlighted in the following screenshot:

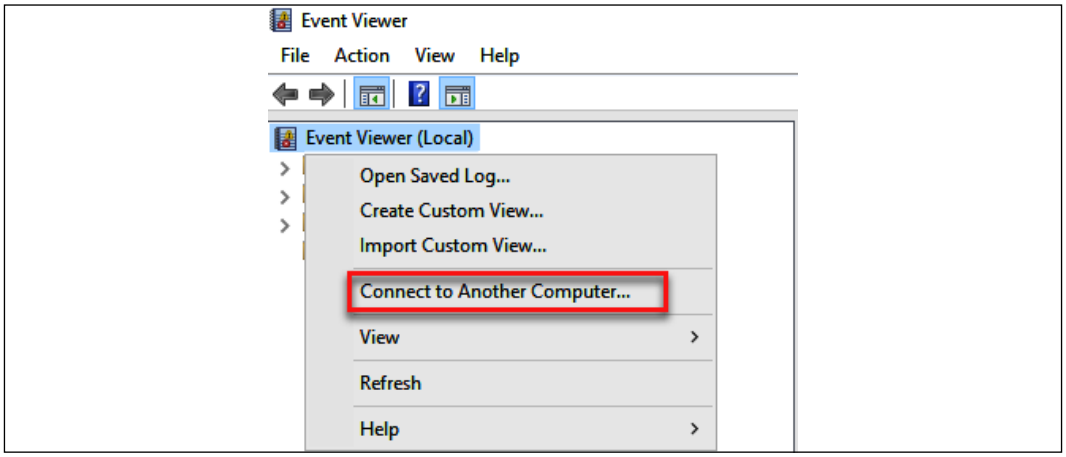


Figure 19.4: Review events on another computer

We can simplify this by creating server groups in Server Manager. Server groups allow us to group systems running similar server roles or acting as part of a distributed system.

Before we go ahead and create server groups, we need to take note of the following information:

1. We need an account that has administrator privileges for all the member servers to create and use server groups.

2. We must enable **Windows Remote Management** (**WinRM**); after Windows Server 2012, WinRM is enabled by default. The existing WinRM configuration can be reviewed using the PowerShell `winrm get winrm/config` command. If it's not enabled, we can enable it using the `winrm quickconfig` command.
3. Even if we are logged in as a domain administrator or an enterprise administrator, by default, it is not allowed to collect events from remote computers. In order to do that, we need to add a collector computer account (the server where the server group is created) to the `Event Log Readers` group. This is a built-in local group. Members of this group can read event logs from the local machine. We can add a computer account to the group using the following command:

```
Add-ADGroupMember –identity 'Event Log Readers'
–members REBELNET-PDC01$
```

`REBELNET-PDC01` can be replaced with the collector computer account.

4. In order to create a server group, go to **Server Manager** from the dashboard and select **Create a server group**, as shown in the following screenshot:

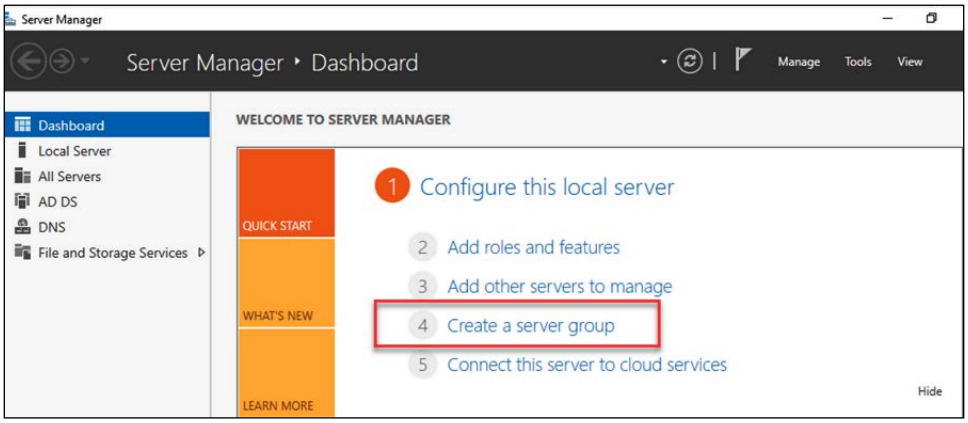


Figure 19.5: Create a server group

5. In the new window, we can provide a name for the group and add members to the group. It provides different methods to select from in order to search for the members, as shown in the following screenshot:

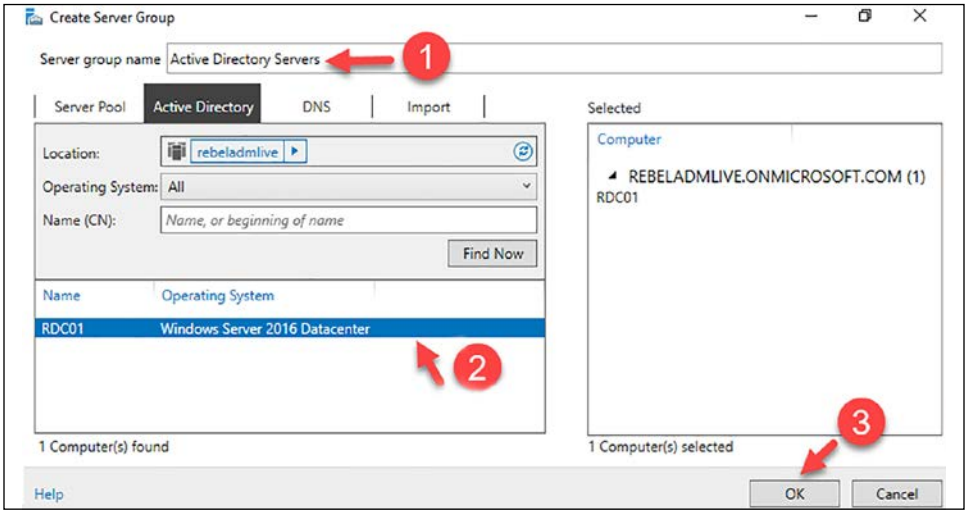


Figure 19.6: Add servers

6. Once a group is created, you can access it using the left-hand panel in **Server Manager**. Inside the group window, there is a separate section called **EVENTS**. When we navigate through each member, it will show us events related to each member in the **EVENTS** window, as shown in the following screenshot:

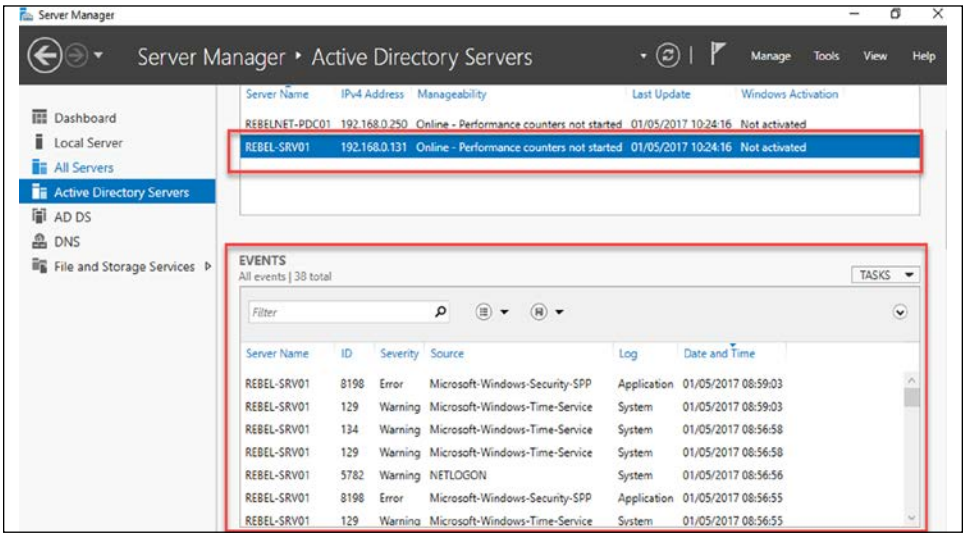


Figure 19.7: Events from the remote server

We can configure the event data and modify it as follows:

- Event severity levels
- Event time frames
- Event log files where the data will be gathered, as shown in the following screenshot:

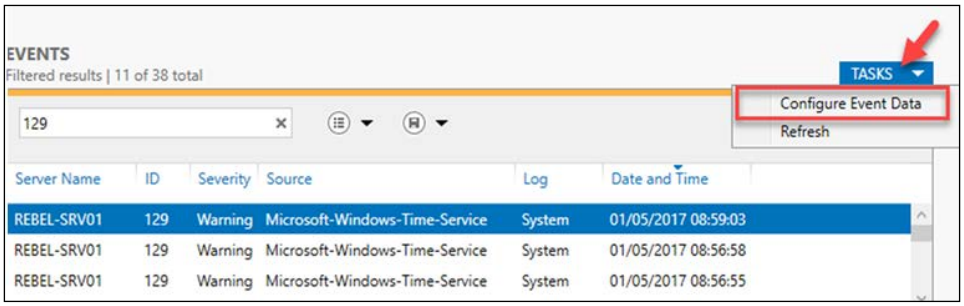


Figure 19.8: Configure Event Data option

The following screenshot explains how we can configure event data using different options:

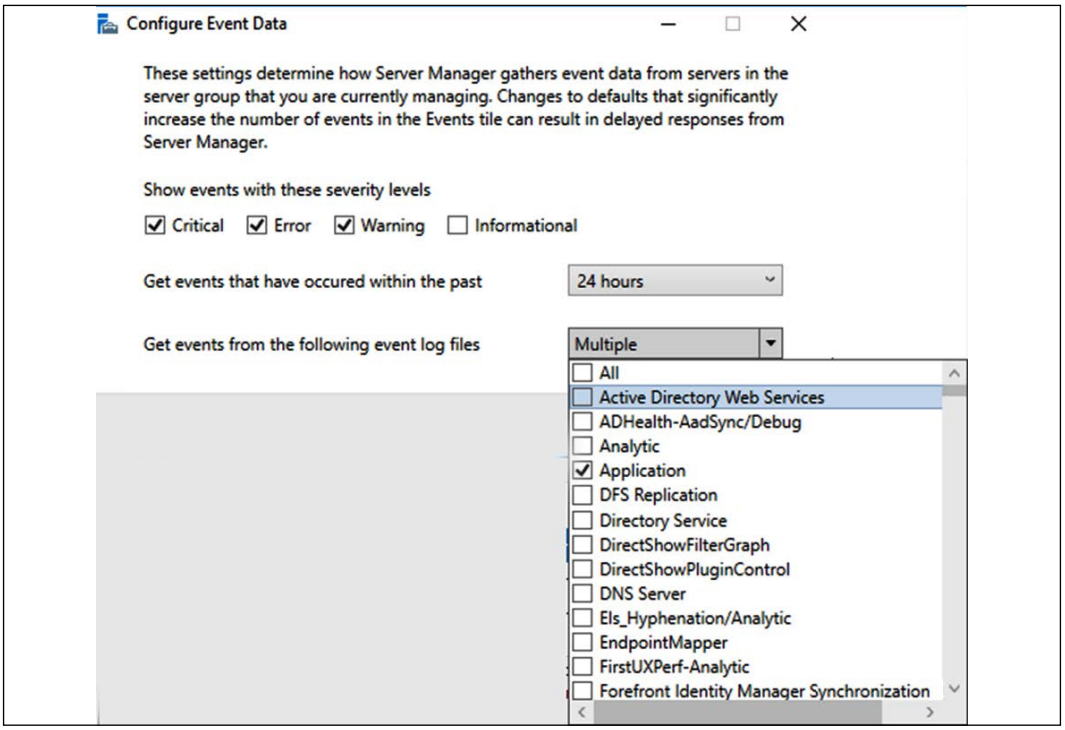


Figure 19.9: Configure Event Data window

We also can filter events and save them as queries for future use. As an example, I need to list events with ID `129`. I can just filter it out by typing `129` in the filter field. But at the same time, I can create a query for it and save it for future use. So, next time, I can just run the query to filter out the data, as shown in the following screenshot:

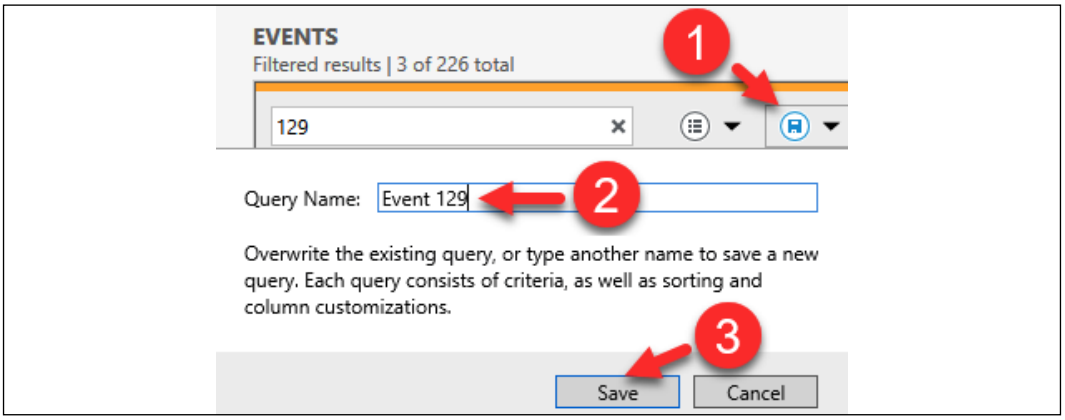


Figure 19.10: Save a query

The following screenshot shows how once the query is created; it can be accessed whenever needed:

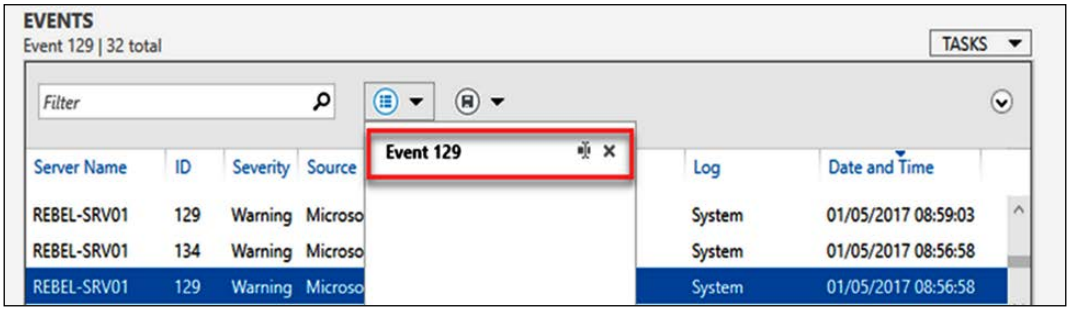


Figure 19.11: Access a saved query

So far, we have only considered accessing Event Viewer data locally. In the next section, we are going to look into accessing Event Viewer data centrally.

Setting up event subscriptions

Event Viewer contains lots of different event entries. There can be several thousand events per day. Even if every event provides some useful information, we do not need to go through each and every one when we are troubleshooting a particular application or performing a service audit. There are specific events relevant to each server role, application, service, and system component.

On some occasions, when we're auditing or troubleshooting, we need to review events on multiple computers. Event Viewer only allows us to connect to one computer at a given time. It can be a local or a remote computer. Event subscriptions allow us to collect event logs from remote computers and review them on one console.

Before we configure event subscriptions, we need to perform the following steps:

1. Enable WinRM.
2. Add a collector computer account to the `Event Log Readers` group.

Configuration steps for the aforementioned tasks are explained in the previous section.

Once the prerequisites are fulfilled, follow these steps:

1. Log in to the Collector server.
2. Open **Event Viewer** and go to **Actions | Create Subscription**.
3. In the new window, enter the following details:

    - **Subscription name**: The name of the subscription job.
    - **Destination log**: The log file where collected events should appear. By default, it is the **Forwarded Events** log file.

      We can select any log file available in the drop-down menu, as shown in the following screenshot:

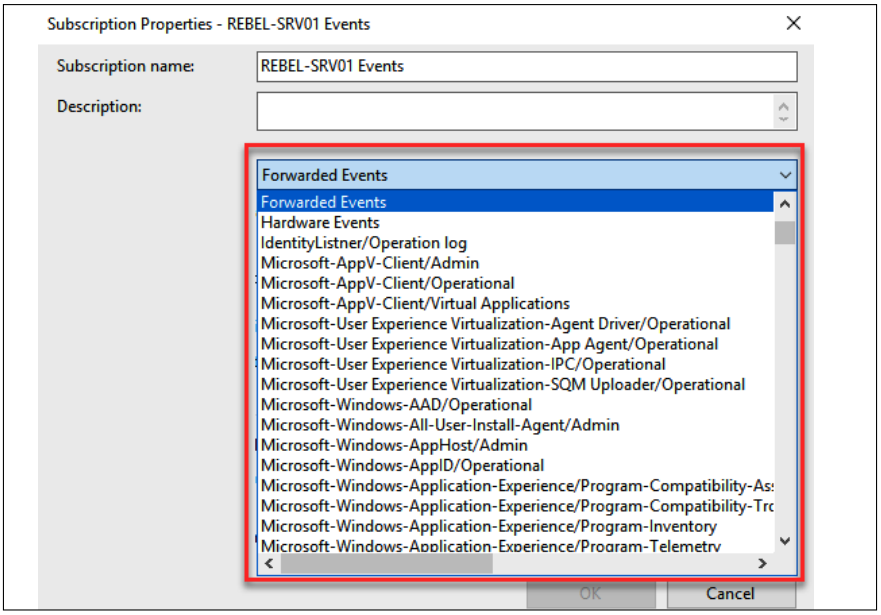


Figure 19.12: Event subscription

- **Collector initiated**: We can list the source computers here. It is not a one-to-one connection. It can be any number of computers, as shown in the following screenshot:

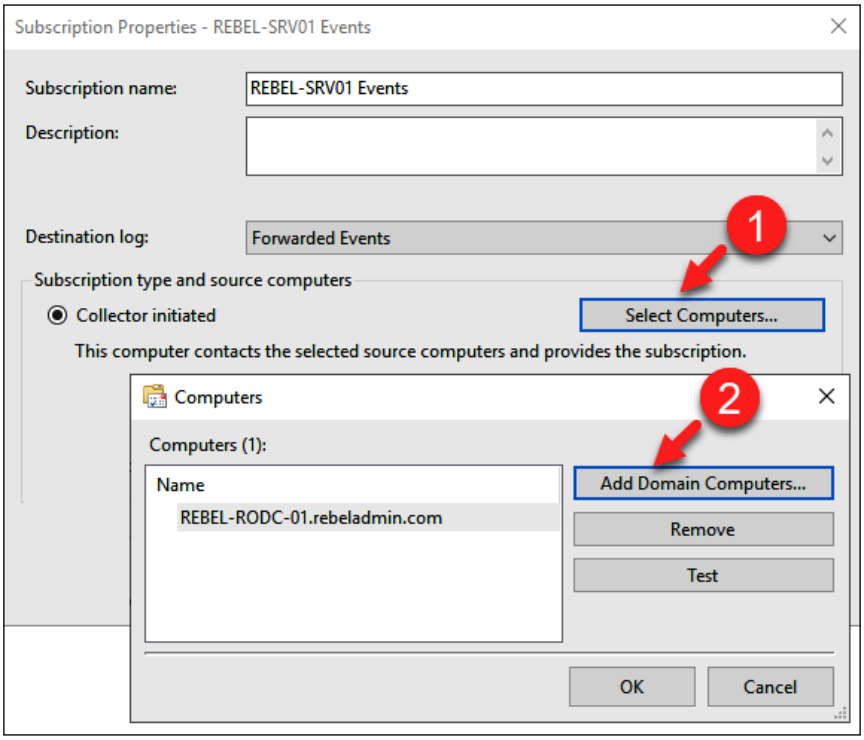


Figure 19.13: Select source computer

- **Source computer initiated**: This allows you to define a subscription without defining the event source computers. Then, the source computers will be defined using a group policy setting located at **Computer Configuration | Policies | Administrative Templates | Windows Components | Event Forwarding | Configure Target Subscription Manager**.

  In there, the collector should be added in the `Server=http://<eventcollector FQDN>:5985/wsman/SubscriptionManager/WEC,Refresh=10` format.

- **Event to collect**: Using this option, we can define which events are to be selected from the source computers. It is similar to a typical event filter window, as can be seen in the following screenshot:

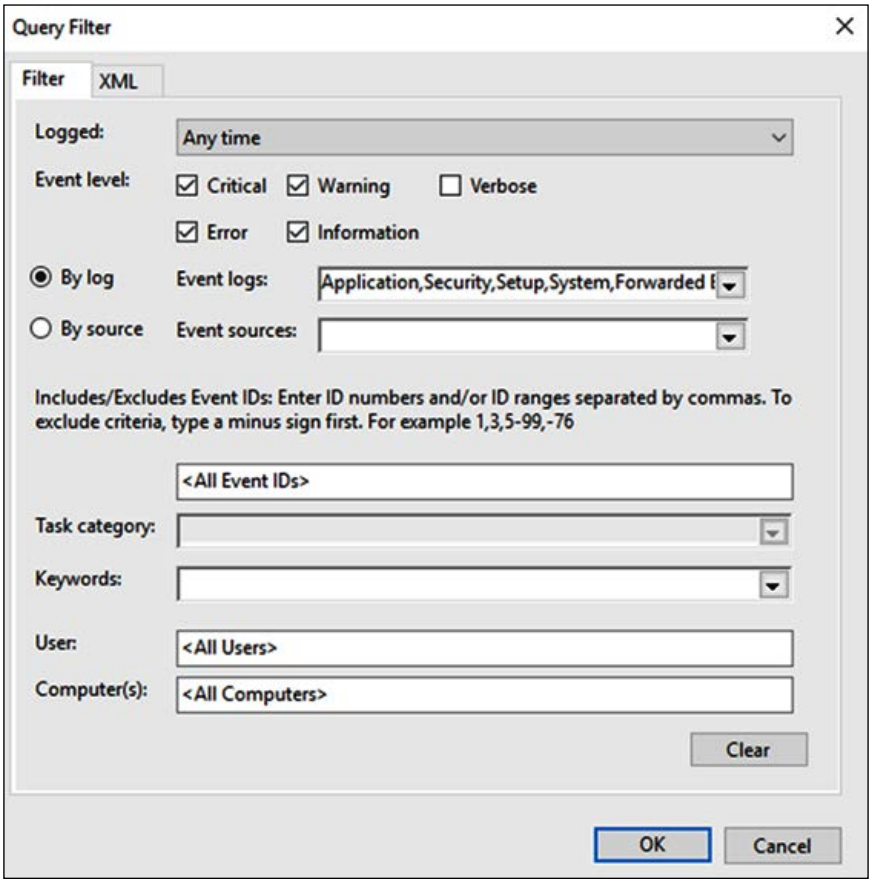


Figure 19.14: Query events

- **Change user account or configure advanced settings**: In this option, we can define a separate account that can be used by a collector to extract events from source computers. It also gives us options to optimize the event delivery settings. This is important if a large number of events have been collected. An example can be seen in the following screenshot:

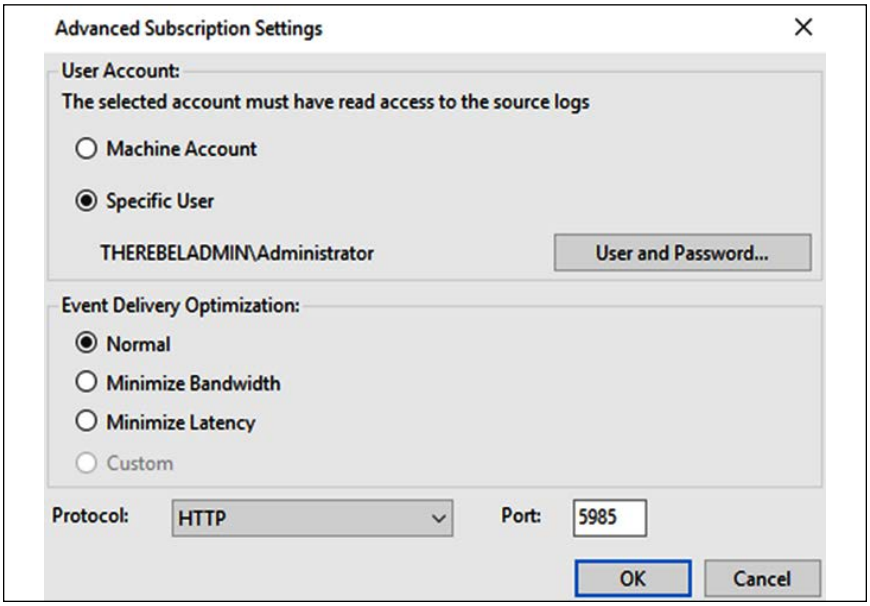


Figure 19.15: Advanced Subscription Settings

By default, we can't collect security logs from remote domain controllers due to permissions issues. Let's see how we can update the permissions manually to access the relevant data.

Security event logs from domain controllers

In order to collect security logs from remote domain controllers, we need to add a network service account to the channel access permissions of the security event log.

This is because the WinRM service is running under the network service account. This can be done by running the following code:

```
wevtutil sl security /ca:'O:BAG:SYD:(A;;0xf0005;;;SY)(A;;0x5;;;BA)
(A;;0x1;;;S-1-5-32-573)(A;;0x1;;;S-1-5-20)'
```

`O:BAG:SYD:(A;;0xf0005;;;SY)(A;;0x5;;;BA)(A;;0x1;;;S-1-5-32-573)(A;;0x1;;;S-1-5-20)` contains READ permission settings for network service account `(A;;0x1;;;)`. In the preceding code, the SID value for the network service account is `(S-1-5-20)`, and the channel access value is `(O:BAG:SYD:(A;;0xf0005;;;SY)(A;;0x5;;;BA)(A;;0x1;;;S-1-5-32-573))`. Once all this is done, after a few minutes, we can see the Forwarded Events.

Enabling advanced security audit policies

As we have seen previously, for successful auditing, we need to have a SACL configured for the relevant AD objects. If there is no SACL entry, no events will be generated against that object. In order to configure the SACL, we need Domain Admin or Enterprise Admin privileges. To add a SACL entry, perform the following steps:

1. Open **AD Users and Computers**.
2. Click on **View | Advanced Features**.
3. Right-click on the OU or the object that you'd like to enable auditing for. Then click on Properties. In my example, I am using the root container, as I wish to enable it globally.
4. Click on the **Security tab** and then on **Advanced**.
5. Click on the **Auditing tab** and then click on the **Add button** to add a new security principle to the SACL. In our scenario, I am using Everyone as I'd like to audit everything.

6. For Type, I have selected the Success event type. Also, I've applied it to This object and all descendant objects, as can be seen in the following screenshot:

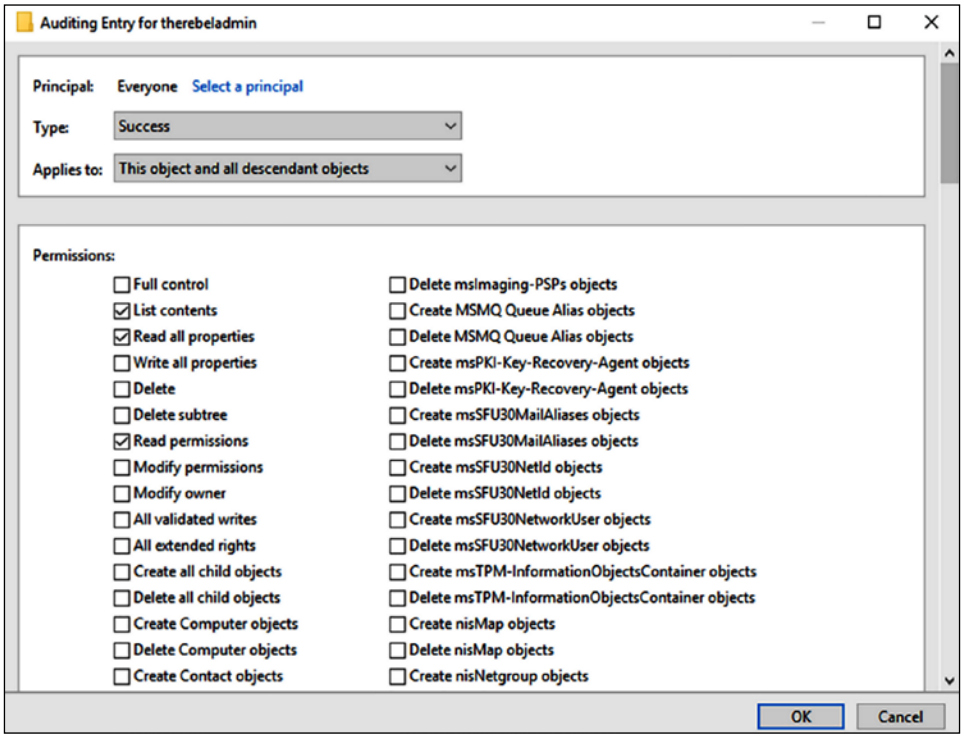


Figure 19.16: SACL entry

Once the SACL entries are in place, we can enable advanced audit policy configuration. In order to do that, perform the following steps:

1. Go to **Group Policy Management**.
2. In the MMC, expand the **Domain Controllers OU**.
3. Right-click on Default Domain Controller Policy and select **Edit**.
4. Then navigate to **Computer Configuration | Policies | Windows Settings | Security Settings | Advanced Audit Policy Configuration | Audit Policies**.
5. In there, we can find all 10 audit categories. In this demo, we are only going to enable audit categories under DS Access.
6. Navigate to **DS Access** and double-click on the **Subcategory entry**. To enable auditing, select Configure the following audit events and then select the events you'd like to audit. It's recommended to audit both Success and Failure, as shown in the following screenshot:

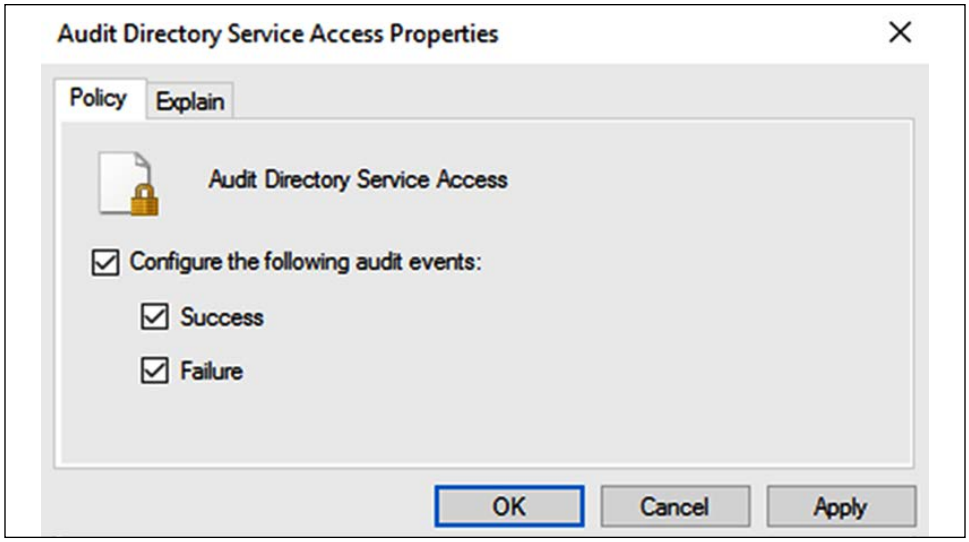


Figure 19.17: Audit Directory Service Access events

I have repeated the same configuration for the rest of the audit categories, as shown in the following screenshot:

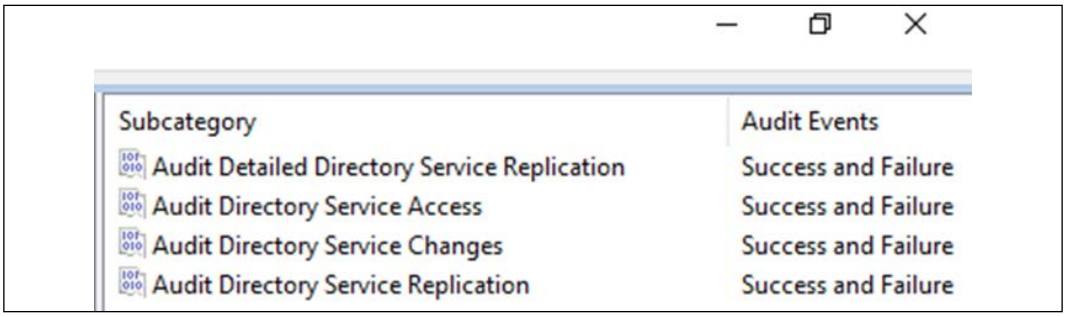


Figure 19.18: Audit categories

Once the group policy is applied successfully, it will start to log new events according to the audit policy.

Enforcing advanced auditing

Before Windows Server 2008, there were nine main auditing categories and subcategories. Those still continue to appear under Windows Server 2022. It is recommended not to mix them up, and only use advanced auditing instead. We can enforce the system to only accept advanced auditing policy settings if legacy audit policy settings are applied to the same category.

This can be done by enabling the Group Policy setting under **Computer Configuration | Windows Settings | Security Settings | Local Policies | Security Options | Audit**: Force audit policy subcategory settings (Windows Vista or later) to override audit policy category settings.

Reviewing events with PowerShell

We also can use PowerShell commands to review event logs or filter events from local and remote computers without any additional service configurations. `Get-EventLog` is the primary cmdlet we can use for this task, as shown in the following example:

```
Get-EventLog -List
```

The previous command will list the details about the log files in your local system, including the log file name, max log file size, and number of entries.

```
Get-EventLog -LogName 'Directory Service' | fl
```

The previous command will list all the events under the `Directory Service` log file. We can also limit the number of events we need to list. As an example, if we only need to list the latest `5` events from the `Directory Service` log file, we can use the following command:

```
Get-EventLog -Newest 5 -LogName 'Directory Service'
```

We can further filter it down by listing events according to entry type, as shown in the following example:

```
Get-EventLog -Newest 5 -LogName 'Directory Service' -EntryType Error
```

The previous command will list the first `5` errors in the `Directory Service` log file. We also can add a time limit to filter events further, as follows:

```
Get-EventLog -Newest 5 -LogName 'Directory Service' -EntryType Error –
After (Get-Date).AddDays(-1)
```

The previous command will list the events with error type `Error` within the last 24 hours under the `Directory Service` log. We can also get the events from remote computers as follows:

```
Get-EventLog -Newest 5 -LogName 'Directory Service' -ComputerName
'REBEL-SRV01' | fl -Property *
```

The previous command will list the first `5` log entries in the `Directory Service` log file from the `REBEL-SRV01` remote server.

We can also extract events from several computers simultaneously, as follows:

```
Get-EventLog -Newest 5 -LogName 'Directory Service' -ComputerName
"localhost","REBEL-SRV01"
```

The previous command will list the log entries from the local computer and the `REBEL-SRV01` remote computer. When it comes to filtering, we can further filter events using the event source, as follows:

```
Get-EventLog -LogName 'Directory Service' -Source "NTDS KCC"
```

The previous command will list the events with the source `NTDS KCC`. It also allows us to search for the specific event IDs, as shown in the following example:

```
Get-EventLog -LogName 'Directory Service' | where {$_.eventID -eq 1000}
```

The previous command will list the events with `eventID` as `1000`.

There is a recommended list of events that we need to audit periodically to identify potential issues in an AD environment. The complete list is available for review at `https://bit.ly/3nKlQK4`.

Microsoft Defender for Identity

In several places in this book, I have talked about why we need the Zero-Trust approach to security. Zero-Trust is not a product or service it is a mindset. We need to understand the importance of this approach and implement relevant controls where ever possible. Especially with the COVID-19 pandemic, the word "Zero-Trust" is resounding in the tech industry, and this completely makes sense due to the following reasons:

1. Today, IT security is getting more and more complex. Security is no longer someone's job alone. Everyone has a role to play when it comes to IT security. We have more and more devices connecting to corporate networks and data coming from remote locations. Attacks are also getting more and more sophisticated.
2. The "perimeter defense" security strategy is no longer working. We have data and applications running on on-prem as well as in the cloud. Users are connecting to these services from everywhere. We no longer can define boundaries for networks. As the attack surface expands, new attack scenarios are developed and used by attackers.

Once attackers achieve the initial breach, they don't stop there. Instead, they move laterally to the cloud or other remote networks. The Solorigate attack is a great example of this.

3. Attackers have shifted their attention to identity attacks. Over the years we have seen the number of identity attacks rising. It has been the most successful method for attackers, due to assets leaving corporate networks.

Out of those concerns, if we consider "identities" specifically, we can see the complexity of enterprise identities is growing, which can create room for attackers to find success. Enterprises are moving assets to the cloud more due to the accelerated pace of digital transformation. This means that enterprise identities are starting to appear in public clouds, SaaS applications, partner systems, BYOD, and mobile devices. When complexity grows,

1. It is possible for engineers to miss certain security settings, rules, or configurations. In most scenarios, these are only noticed during a pen-test or vulnerability scan.
2. Threats can come from anywhere – on-prem, BYOD, applications, clouds, and so on. So, protecting all of these areas can be difficult and costly.
3. Security teams can get overwhelmed by the amount of data, logs, and alerts being gathered and it is possible to miss important data.

Microsoft identified these challenges and came up with Defender for Identity, a cloud-based solution that monitors signals from on-prem AD Domain Controllers and AD FS servers to detect, identify, and investigate identity threats and malicious activities.

What is Microsoft Defender for Identity?

In the previous two editions of this book, this section was allocated to **Microsoft Advanced Threat Analytics** (**ATA**). This is an on-prem platform to help us protect our identity infrastructure from advanced targeted attacks by automatically analyzing, learning, and identifying normal and abnormal behavior (from users, devices, and resources). Microsoft also had a cloud version of it called **Azure Advanced Threat Protection** (**Azure ATP**). This cloud service has now been renamed **Defender for Identity**. Microsoft ATA mainstream support ended on January 12, 2021, so going forward, users only can use the cloud-based Defender for Identity service.

When we consider a typical attack kill chain, we can identify four main areas to protect:

1. Applications
2. Endpoints

3. Identity
4. Data

Microsoft offers security solutions to protect all these areas:

1. Applications – Microsoft Defender for Office 365, **Microsoft Cloud App Security** (**MCAS**)
2. Endpoints – Microsoft Defender for Endpoints
3. Identity – Microsoft Defender for Identity, MCAS
4. Data – Microsoft Defender for Office 365, MCAS

Microsoft has grouped these four products – Microsoft Defender for Office 365, Microsoft Defender for Endpoints, Microsoft Defender for Identity, and MCAS – and created a unified pre- and post-breach enterprise defense suite called **Microsoft 365 Defender**, which can provide integrated protection for all four areas. So, Microsoft Defender for Identity is one pillar of a far greater solution. If you want to achieve unified protection in stages, it is recommended to start with identity protection first and then move to other products in the suite, as identity is the key to everything.

Defender for Identity benefits

Defender for Identity has the following key capabilities that help to streamline SecOps operations:

1. **Proactive –** Detect vulnerabilities proactively and prevent attacks before they happen.
2. **Efficient –** Automatic analysis and automatic response help SecOps teams to allocate their time to investigating critical issues.
3. **Prioritize –** By reducing false positives, Defender for Identity helps SecOps teams to prioritize spending their time on dealing with the real issues.

When it comes to identity protection, Microsoft Defender for Identity focuses on four main deliverables.

Prevent

Defender for Identity helps SecOps teams to identify hidden vulnerabilities in their environments. These are present mostly due to misconfiguration of services/ products and a lack of system updates.

The Defender for Identity security posture assessment can detect vulnerabilities such as:

- Exposing credentials in clear text
- Legacy protocols in use
- Unsecure Kerberos delegation
- Dormant accounts in sensitive groups
- Weak cipher usage
- Print spooler service in domain controllers
- Not managing local administrator accounts centrally (Microsoft LAPS)
- Lateral movement paths
- Unmonitored domain controllers
- Unsecure account attributes
- Unsecure SID history attributes

Defender for Identity not only detects these issues, but also advises what should be done to fix these issues. Defender for Identity's reports show how many times these issues have been reported, their impact levels, and their resolution statuses. This rich information helps SecOps teams to improve their secure posture and prevent attacks proactively.

Detect

In the Zero-Trust security approach, we need to assume a breach. We can't close all the doors. But more importantly, if there is breach, we need a way to "detect" it quickly so we can prevent further damage and also use that information to further improve our security. Microsoft Defender for Identity can detect identity attacks faster due to its analysis of rich information from a variety of data sources:

- **Network traffic analytics –** Defender for Identity analyzes network traffic passing through domain controllers in real time. It can inspect traffic from protocols including NTLM, LDAP, DNS, SMB, and RPC.
- **AD security events and AD Data –** One of the requirements during Defender for Identity deployment is to enable advanced auditing for domain controllers. This allows Defender for Identity to collect sensitive security events, which helps to detect potential attacks. It also profiles AD data for users, groups, and resources.
- **User behavior analytics –** Defender for Identity analyzes user behavior to identify what is anomalous.

- **Cloud-based detection –** Defender for Identity uses the cloud to enrich the data collected with additional information (such as IP details). This helps to detect attacks in real time.
- **AD FS support –** Defender for Identity now supports running identity sensors in AD FS servers and collecting information to detect attacks such as Nobelium.

A typical identity attack has different stages. First of all, the attacker needs to uncover details about the environment they are attacking. We call this the **reconnaissance** stage of the attack. Defender for Identity can detect the following types of reconnaissance events:

- LDAP enumeration
- Group membership enumeration
- Users and IP enumeration
- DNS enumeration
- Suspicious resource-access activities
- Reconnaissance by targeted entity attributes

After reconnaissance, the attacker's next step is to gain some sort of access to the system. The attacker can use many different methods for this. Defender for Identity can detect the following types of credential access events:

- Brute force attacks (AD DS and AD FS)
- Login/failed suspicious activities
- Suspicious DC password change suing NetLogon
- Suspicious Kerberos SPN exposure
- Honeytoken account activities
- Suspicious VPN activities

If an attacker successfully achieves an initial breach, the next step will be to move laterally within the infrastructure and try to gain more privileges. Most of the time, the initial breach is a typical end user account. To do significant damage, attackers will need higher privileges, so they will continue compromising systems until they have access to the keys of the kingdom, Enterprise/Domain Admin accounts. Defender for Identity can detect the following types of events, helping to identify lateral movement attempts:

- Pass-the-ticket attacks
- Pass-the-hash attacks

- NTLM relay and NTLM tampering
- Overpass-the-hash
- Suspicious certificates
- Suspicious group membership changes
- Suspicious SID history injection

If the attacker has access to privileged accounts, they can go further and compromise the remaining systems to get full control over the infrastructure. If the defender uses a hybrid setup, the attacker can try to extend their attack to cloud resources. Defender for Identity can detect the following types of events, helping to detect an attack in progress:

- Golden ticket attack
- Data exfiltration
- Code execution/service creation on DC and AD FS servers
- SMP manipulation
- Skelton key
- DNS remote code execution attempt
- Golden ticket leveraging
- Suspicious print spooler registration

It is important to know whether an attack is progressing to the stage mentioned above, as this means attacker already has access to most of the system. In such a situation, we need to act fast to minimize the impact.

Investigate

When there is a security event, SecOps teams can get overwhelmed by the number of events and alerts they are receiving. Prioritization of these events mostly depends on the skills of the engineers. Also, if a large number of alerts have been produced, it's possible for the engineers to miss some critical events.

Defender for Identity not only alerts us to security incidents, but also helps SecOps teams to prioritise their investigation efforts to make them more effective. As an example, **User investigation priority score** for a user account will help to shortlist the accounts for which immediate attention is required. A higher score here means more attention is required. This value is calculated based on the alerts connected to the user and any abnormal activities detected from the user account.

Some attacks involve large numbers of users. In such a situation, we can use **impact ranking** (from Low to High) to easily identify the users requiring immediate attention. Each attack can have multiple incidents. Defender for Identity marks each incident with a severity score (from Low to High), which also helps SecOps teams to identify which incidents to focus on first.

In addition to this, Defender for Identity also allows us to create custom detection rules to help SecOps teams focus on environment-specific security events.

We also can use Microsoft 365 Defender advanced threat hunting to query cross-domain data to investigate security events further. For that, we need to use KQL queries.

Respond

So far, we have seen how Defender for Identity can increase the efficiency of detection and prioritization of incidents. This also helps to reduce the time it takes from the initial detection of an event to its final fix.

Not only that, but when it comes to incident response, we can use Microsoft 365 Defender to automate our incident response process. For more information about automated remediation, please visit `https://bit.ly/3l7kZkK`.

Microsoft Defender for Identity architecture

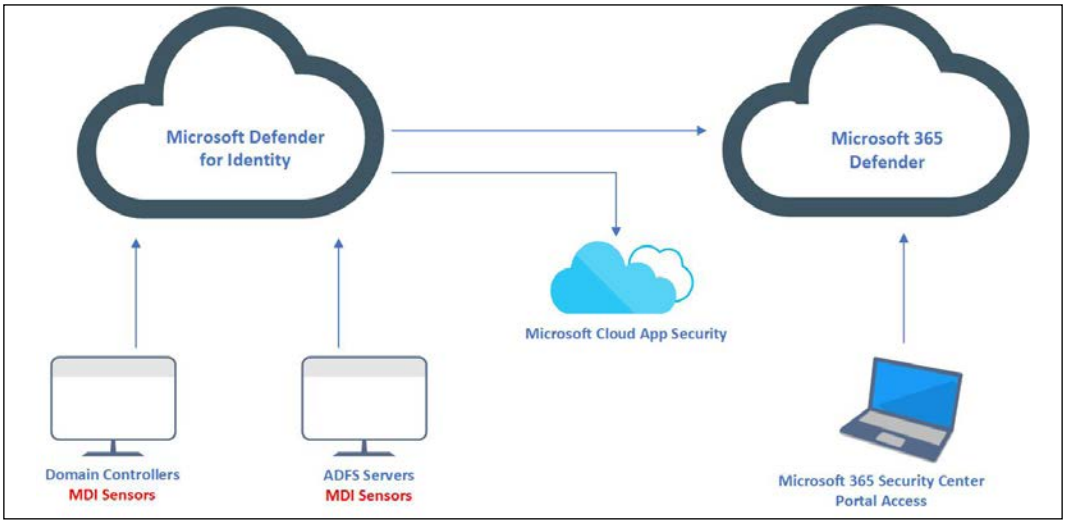


Figure 19.19: MDI architecture

The preceding diagram explains the components involved in the Microsoft Defender for Identity setup. Let's go ahead and see the roles of each of these components:

Microsoft Defender for Identity portal – This portal allows us to configure our Defender for Identity instance. Using this portal we can download MID sensors, check the status of our MID sensors, configure honeytoken accounts, configure email settings, and so on. We also can view and investigate security incidents in the environment using the Microsoft Defender for Identity portal.

MDI sensors – Microsoft Defender for Identity collects security events, analyzes network traffic, and monitors AD entities via MDI sensors. These sensors need to be installed in each domain controller and AD FS server in the environment for best results. Defender for Identity also has standalone sensors. Instead of installing a sensor in each domain controller, we can configure a dedicated server as a sensor. We then need to configure port mirroring in our domain controllers to pass traffic through the standalone sensor. However, this standalone sensor can't collect **Event Tracing for Windows** (**ETW**) log entries, which use for multiple detection. Microsoft's recommendation is to install sensors on domain controllers and AD FS servers for best results.

Microsoft 365 Defender portal – Defender for Identity is a product in the Microsoft 365 Defender suite. It uses one portal to collect data from different products and then analyzes the data to identify attacks spread across different domains. Using this portal, SecOps teams can also do advanced threat hunting. If we need to configure automated remediation or custom detection rules, we need to use this portal to do it. The Microsoft Defender for Identity portal also forwards activity data, alerts, and metadata to the Microsoft 365 Defender portal. It is recommended to use the Microsoft 365 Defender portal for investigation as it contains rich data from different sources.

Microsoft Cloud App Security – Prior to the release of the Microsoft 365 Defender portal, the data it now handles was forwarded to Microsoft Cloud App Security, and engineers were able to do their investigations using the Cloud App Security portal.

In the next section, we are going to look into the things we need to consider before deployment of Microsoft Defender for Identity.

Microsoft Defender for Identity prerequisites

Before we deploy Microsoft Defender for Identity, it is important to check for the following prerequisites.

Licenses

Microsoft Defender for Identity requires the **Enterprise Mobility + Security E5** (**EMS E5/A5**), **Microsoft 365 E5** (**M365 E5/A5/G5**), or standalone licenses. These licenses can purchase through Microsoft 365 portals or via your **Cloud Solution Partner** (**CSP**).

Connectivity to the Defender for Identity cloud service

It is a common best practice to block/limit internet access to domain controllers. Unless you are using standalone sensors, all domain controllers with MDI sensors need to be able to communicate with the Defender for Identity cloud service. More info about the links and proxy configuration is available on `https://bit.ly/3cM3KkB`.

Service accounts

Microsoft Defender for Identity requires a service account to connect to AD. This service account must have read permissions for all objects in the domain.

Microsoft Defender for Identity supports two types of service accounts:

1. Stand-alone service account
2. **Group Managed Service Account** (**gMSA**)

If the sensors' machines are running Windows Server 2012 or above, the recommended type of service account is gMSA. In *Chapter 8*, I explained gMSA and how we can set up this type of account. Please check *Chapter 8* for more info.

If you are using an older operating system than Windows Server 2012, you will have to use a standalone service account.

Honeytoken account

During the reconnaissance or lateral movement phases of an attack, the hackers will try to access different user accounts. A honeytoken account helps MDI to detect such activities quickly. This account should be set up as a standard company account, but should never be used to log in. If any activity is detected in this honeytoken account, it will immediately be flagged by Microsoft Defender for Identity.

Firewall ports

Certain ports need to be open in your firewalls for internal and external communication by Microsoft Defender for Identity, as follows:

Protocol	TCP/UDP	Port	To/From	Direction
SSL	TCP	443	Defender for Identity cloud services	Outbound
SSL	TCP	444	Sensor service	Both
DNS	TCP and UDP	53	Sensors to DNS Servers	Outbound
Netlogon	TCP/UDP	445	Sensors to all devices	Outbound
RADIUS	UDP	1813	RADIUS to sensors	Inbound
NTLM over RPC	TCP	135	Sensors to all devices	Outbound
NetBIOS	UDP	137	Sensors to all devices	Outbound
TLS to RDP	TCP	3389	Sensors to all devices	Outbound

For standalone sensors, the port requirements are different. More info about this is available at `https://bit.ly/3l6MVFr`.

Advanced audit policies

Defender for Identity detects 4726, 4728, 4729, 4730, 4732, 4733, 4753, 4756, 4757, 4758, 4763, 4776, 7045, and 8004 Windows event logs from domain controllers. Some of these event logs only appear if advanced audit policies are enabled in the domain controllers. I explained earlier in this chapter how we can enable advanced audit policies.

To capture the above events, we need to enable the following advanced audit policies:

1. **Account Logon | Audit Credential Validation**
2. **Account Management | Audit User Account Management**
3. **Account Management | Audit Distribution Group Management**
4. **Account Management | Audit Computer Account Management**
5. **Account Management | Audit Security Group Management**

We need to capture **Success** and **Failure** events for all the above policies.

NTLM auditing

Windows event 8004 contains NTLM authentication data. By enabling NTLM auditing, we can allow Microsoft Defender for Identity to enrich event data by displaying the **source user**, **source device**, and **accessed resource**.

NTLM auditing should be enabled on domain controllers. We can enable this by using GPO settings under **Computer configuration | Policies | Windows Settings | Security Settings | Local Policies | Security Options**.

To collect the relevant data, we need to enable the following policy settings:

1. **Network Security | Restrict NTLM | Outgoing NTLM traffic to remote servers (Audit All)**
2. **Network Security | Restrict NTLM | Audit NTLM authentication in this domain (Enable all)**
3. **Network Security | Restrict NTLM | Audit incoming NTLM Traffic (Enable auditing for all accounts)**

Once the settings are in place, MDI will display the NTLM data in Resource Access over NTLM and Failed logon events.

SAM-R Permissions

Microsoft Defender for Identity can detect lateral movement paths. To do this, MDI needs to be able to query the local administrators of any computer. This query uses the SAM-R protocol and the Microsoft Defender for Identity service account. By default, the service account can't do this query, so we need to grant the relevant permissions by using GPO. This policy setting should apply to all computers except domain controllers. To update the settings, go to **Computer Configuration | Windows Settings | Security Settings | Local Policies | Security Options** and add the service account to the list under the **Network access –** Restrict clients allowed to make remote calls to SAM policy.

Sizing tool

The Microsoft Defender for Identity sensor requires a minimum of 2 cores and 6 GB of RAM. However, this can be changed based on network traffic, server roles, and the server's current resource usage. Therefore, it is best to run the sensor sizing tool before installation. This tool will automatically analyze the current system and make recommendations for resource upgrades. To download the tool, please go to `https://bit.ly/3xh7y6S`.

This completes the prerequisites section for Microsoft Defender for Identity. Please note that I didn't mention much here about standalone sensors because they have limitations and are not the recommended method.

Deployment

After the prerequisites are sorted, the next phase is to set up the Microsoft Defender for Identity service and sensors. This is a very straightforward process and Microsoft has well-structured documentation for it. You can access this at `https://bit.ly/30RUc4L`. After deployment, I highly recommend doing the testing as specified in the documentation, so we know the service is working as expected.

Azure AD Connect Health

In the previous chapter, we learned what Azure AD Connect is and how it works in a hybrid Azure AD environment. Azure AD Connect is responsible for synchronization between Azure AD and on-prem AD. Therefore, it is important to monitor the health of the Azure AD Connect service to make sure it is running as expected. In a given computer infrastructure, only one Azure AD Connect instance can be active at a given time, so this puts more pressure on the health of the service. The Azure AD Connect service is a Windows service, so there are many tools on the market that can monitor the status of the service. But even if the service is up and running, it doesn't mean synchronization is healthy.

Azure AD Connect Health is a service that comes with Azure AD Premium to monitor the health of Azure AD Connect. Azure AD Connect Health can monitor the following types of sync errors:

- Duplicate attributes
- Data mismatches
- Data validation failures
- Large attributes
- Federated domain changes
- Existing admin role conflicts
- Other errors (those not categorized here)

Azure AD Connect Health insights are gathered via health agents.

There are three types of agents used by Azure AD Connect Health:

- **Azure AD Connect** (**sync**): This is installed as part of Azure AD Connect. This agent will gather information related to Azure AD Connect, such as service status, synchronization rules, service name, last sync time, sync errors, and alerts.
- **Azure AD Connect Health Agent for Active Directory Federation Services** (**AD FS**): This agent can gather additional data to monitor the health of Azure AD Connect in a federated environment. This agent will gather information such as the total number of requests processed by AD FS, requests based on relaying party trust, authentication methods used by requests, alerts, failed requests, and so on.
- **Azure AD Connect Health Agent for Active Directory Domain Services** (**AD DS**): This agent can gather additional data from an on-prem AD environment, which will provide additional insights to detect any underlying directory issues in a hybrid environment. This agent gathers information, such as the AD topology, forest and domain functional levels, **Flexible Single Master Operation** (**FSMO**) role holders, replication status, and number of processed authentication requests.

Prerequisites

We need to meet the following prerequisites to use Azure AD Connect Health:

- An Azure AD Premium subscription.
- Relevant Azure AD Connect Health agents installed on target computers.
- Allow outgoing TCP `443 & 5671` traffic to Azure endpoints from the target servers (for a complete list of URLs, see `https://bit.ly/3FK3BdP`)
- PowerShell 5.0 or above installed on the target computers.
- **Federal Information Processing Standards** (**FIPS**) should be disabled.

Configuration

In this section, we are going to look at Azure AD Connect Health in action. In my demonstration environment, I have the latest version of Azure AD Connect installed.

Azure AD Connect Health (sync) comes as a part of it, as shown in the following screenshot:

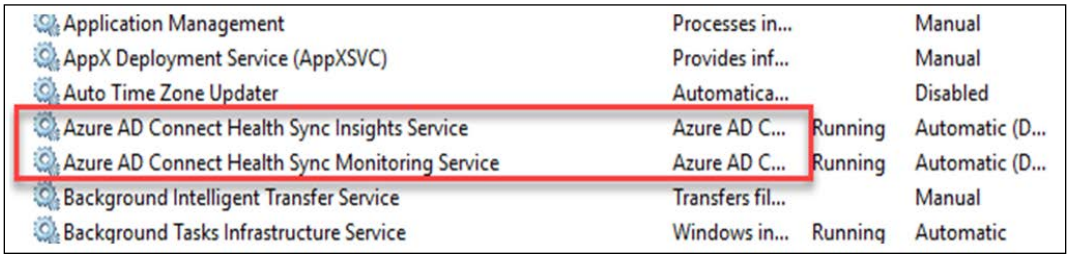


Figure 19.20: Azure AD Connect health services

But I'd like to install **Azure AD Connect Health Agent for AD DS** to gather additional insights from my on-prem AD environment. In order to do that, perform the following steps:

1. Log in to the target computer as the Domain Admin/Enterprise Admin.
2. Go to the Azure portal at `https://portal.azure.com` and log in as a global administrator.
3. Then go to **Azure Active Directory | Azure AD Connect** and click on **Azure AD Connect Health**, as shown in the following screenshot:

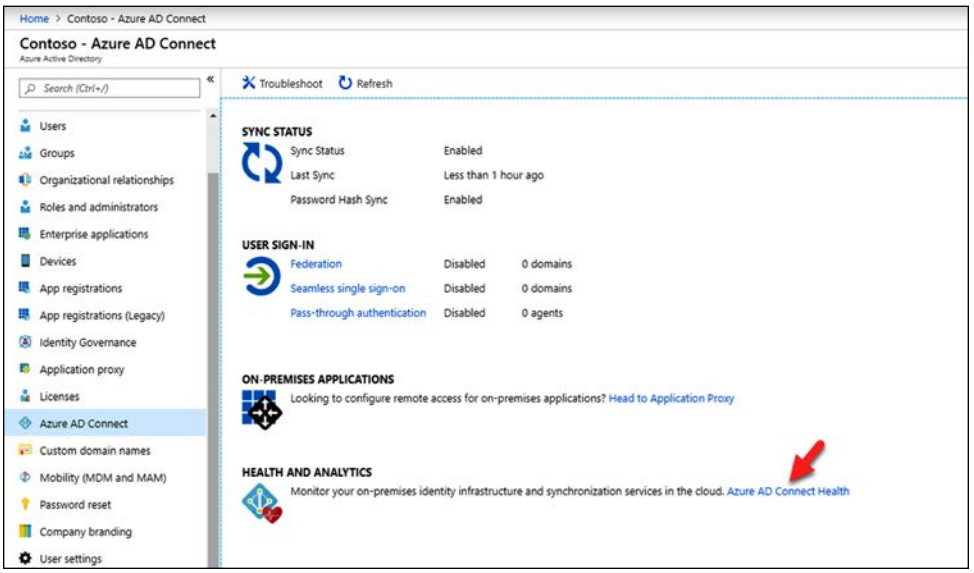


Figure 19.21: Azure AD Connect Health access

4. In the new window, click on **Download Azure AD Connect Health Agent for AD DS**, as shown in the following screenshot:

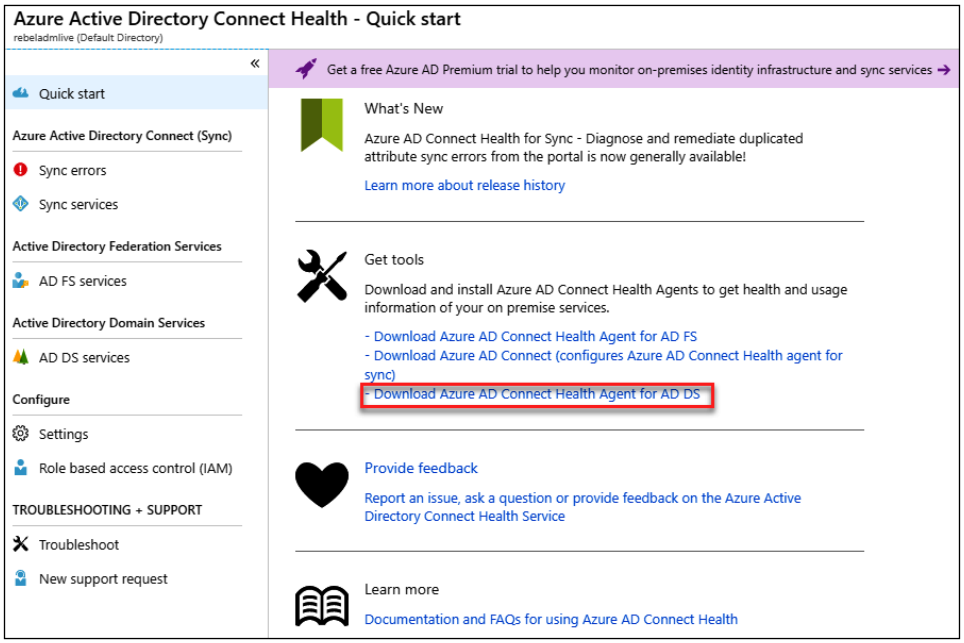


Figure 19.22: Download Azure AD Connect Health Agent for AD DS

5. Once the download is completed, run the `ADHealthAddsAgentSetup.exe` as administrator.
6. Complete the installation, and then sign in using the global administrator account in the login window, as shown in the following screenshot:

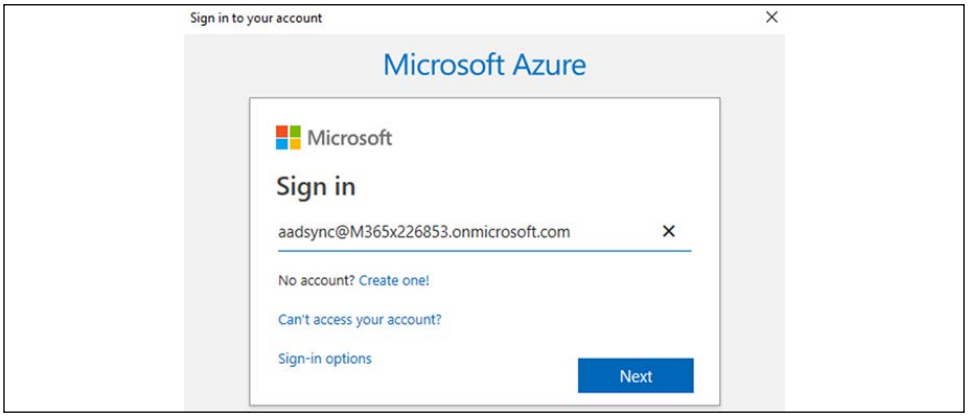


Figure 19.23: Azure AD login

7. Let the system complete the agent registration, as shown in the following screenshot:

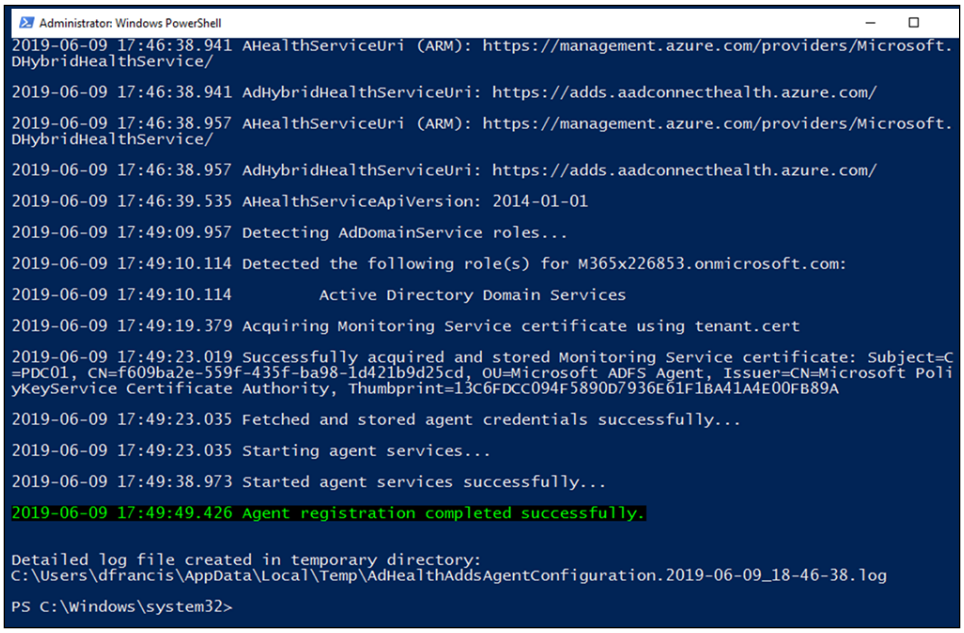


Figure 19.24: Azure AD Connect Health service registration

If a proxy is in place, we have to import the proxy settings from Internet Explorer by running `Set-AzureAdConnectHealthProxySettings -ImportFromInternetSettings`.

Once registration is completed, it can take a few hours to display the collected data.

8. Once agents start reporting, we can view **Azure Active Directory Connect** (**Sync**) data under **Sync errors** and **Sync services**, as shown in the following screenshot:

Figure 19.25: Sync errors and Sync services

The landing page shows the high-level health status of Azure AD Connect:

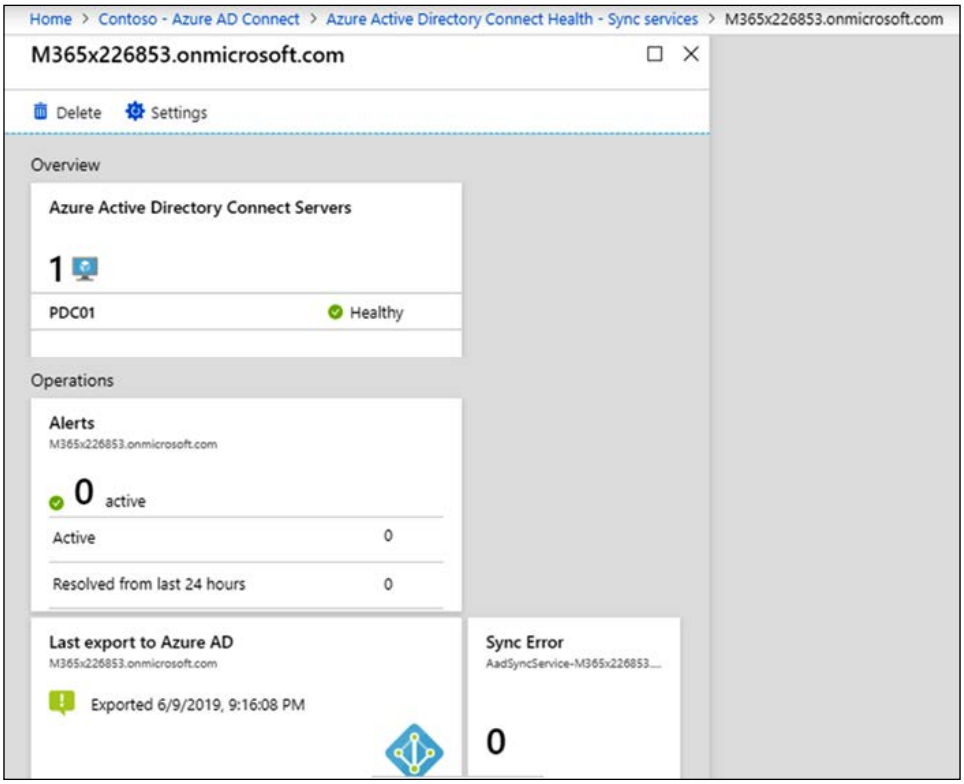


Figure 19.26: Overview of Azure AD Connect Health

On the error pages, we can see detailed descriptions of events:

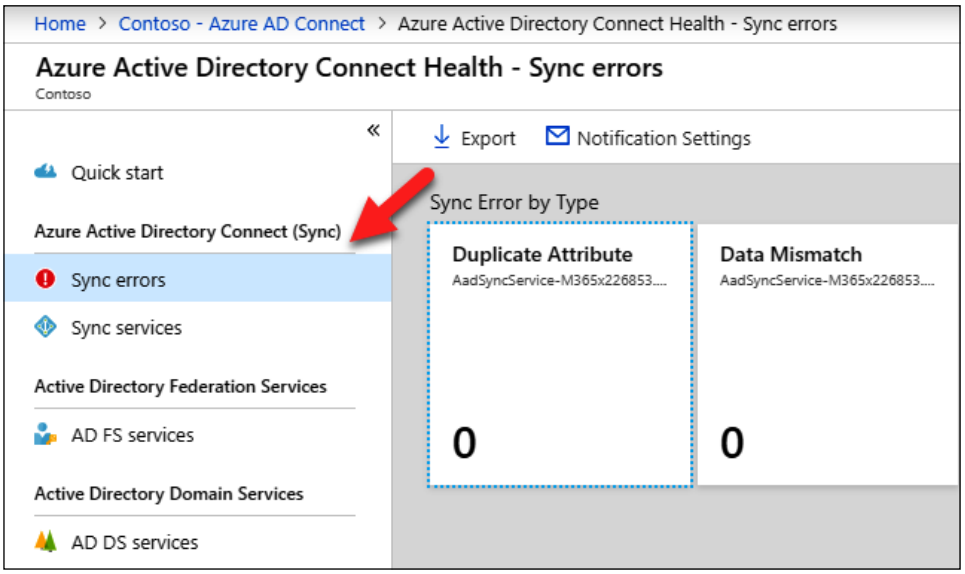


Figure 19.27: Sync errors

9. Similarly, data collected from the **Azure AD Connect Health Agent for AD DS** can be accessed under **Active Directory Domain Services**, as shown in the following screenshots:

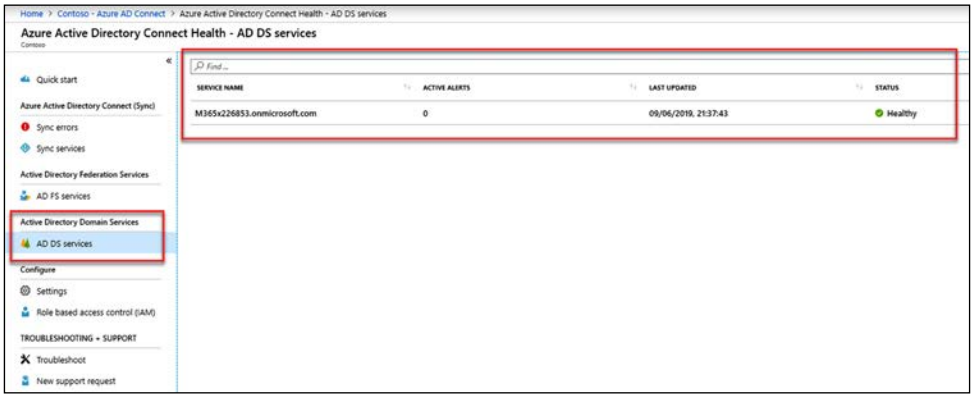


Figure 19.28: AD DS services

The landing page shows the overall health of replication:

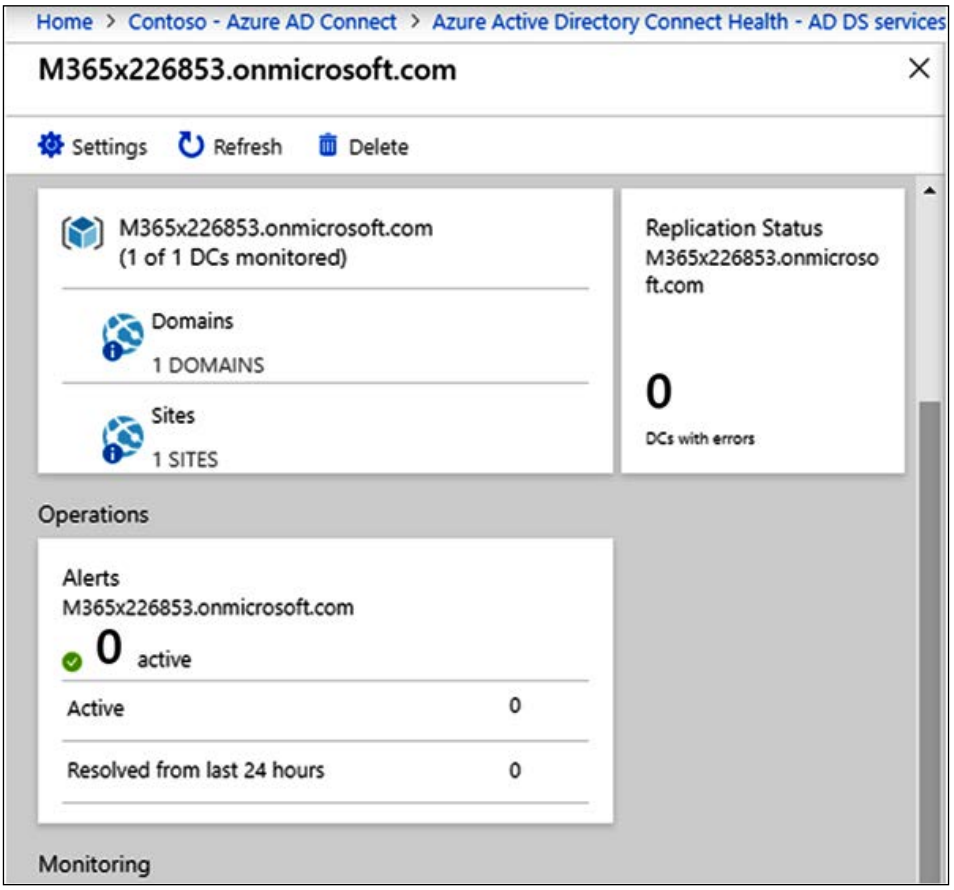


Figure 19.29: Replication health

Azure AD Connect Health is mainly focused on monitoring the health of directory synchronization. Additional insights collected from AD FS and AD DS agents primarily help to troubleshoot directory synchronization health issues. However, healthy synchronization doesn't mean we are running a healthy Azure AD hybrid environment. This can only be ensured by monitoring identity infrastructure security threats. Microsoft has different services that can monitor identity infrastructure threats. Azure Security Center and Azure Sentinel are good examples of this. I highly recommend that you look into these solutions to further improve the overall health of your identity infrastructure.

Summary

Continuous monitoring and auditing are a must for identity infrastructures to identify potential security threats and maintain a healthy environment. There are a lot of tools and methods out there to do this, but the success of these solutions depends on the accuracy of detection, the way it presents data, and how it helps in identifying the root cause.

In this chapter, we started by looking at Windows' built-in tools and methods that can be used to monitor and audit AD environments. First, we started with GUI tools and then moved to PowerShell-based auditing. Then we looked at Microsoft Defender for Identity and how it can help to identify security threats in the infrastructure that cannot be detected using traditional tools and methods. Last but not least, we also learned how Azure AD Connect Health can help to monitor the synchronization health of the Azure AD hybrid environment.

After a long journey, we are now reaching the end of this book. In the final chapter, we will look at the most common AD-related issues and how we can fix them.

packt.com

Subscribe to our online digital library for full access to over 7,000 books and videos, as well as industry leading tools to help you plan your personal development and advance your career. For more information, please visit our website.

Why subscribe?

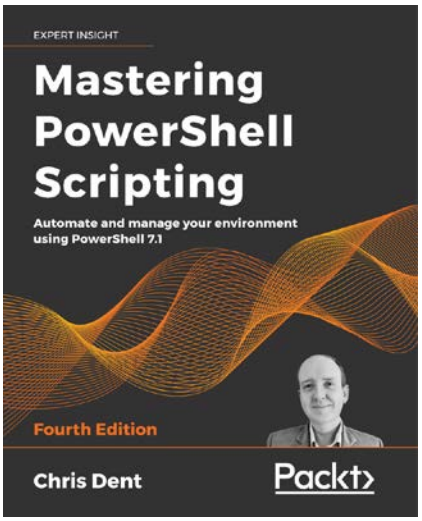
- Spend less time learning and more time coding with practical eBooks and Videos from over 4,000 industry professionals
- Learn better with Skill Plans built especially for you
- Get a free eBook or video every month
- Fully searchable for easy access to vital information
- Copy and paste, print, and bookmark content

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at www.Packt.com and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at customercare@packtpub.com for more details.

At www.Packt.com, you can also read a collection of free technical articles, sign up for a range of free newsletters, and receive exclusive discounts and offers on Packt books and eBooks.

Other Books You May Enjoy

If you enjoyed this book, you may be interested in these other books by Packt:



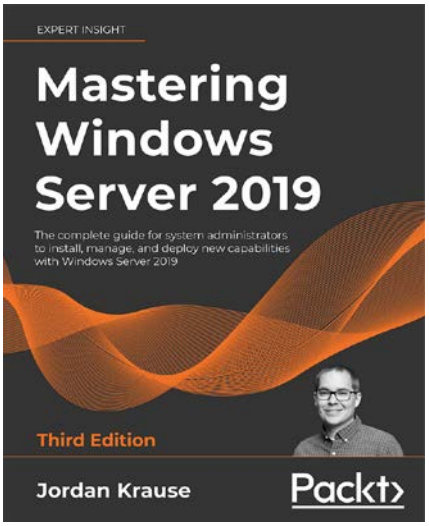
Mastering PowerShell Scripting – Fourth Edition

Chris Dent

ISBN: 978-1-80020-654-0

- Optimize code with functions, switches, and looping structures
- Test and debug your scripts as well as raising and catching errors

- Work with objects and operators to test and manipulate data
- Parse and manipulate different data types
- Use jobs, runspaces, and runspace pools to run code asynchronously
- Write .NET classes with ease within PowerShell
- Create and implement regular expressions in PowerShell scripts
- Make use of advanced techniques to define and restrict the behavior of parameters



Mastering Windows Server 2019 – Third Edition

Jordan Krause

ISBN: 978-1-80107-831-3

- Work with Server Core and Windows Admin Center
- Secure your network and data with modern technologies in Windows Server 2019
- Understand containers and understand when to use Nano Server
- Discover new ways to integrate your datacenter with Microsoft Azure
- Reinforce and secure your Windows Server
- Virtualize your datacenter with Hyper-V
- Explore Server Manager, PowerShell, and Windows Admin Center
- Centralize your information and services using Active Directory and Group Policy

Packt is searching for authors like you

If you're interested in becoming an author for Packt, please visit `authors.packtpub.com` and apply today. We have worked with thousands of developers and tech professionals, just like you, to help them share their insight with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

Share your thoughts

Now you've finished *Mastering Active Directory, Third Edition*, we'd love to hear your thoughts! If you purchased the book from Amazon, please `click here to go straight to the Amazon review page` for this book and share your feedback or leave a review on the site that you purchased it from.

Your review is important to us and the tech community and will help us make sure we're delivering excellent quality content.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

Z