Networks and Systems Security II - Winter 2025 Exercise 3

April 10, 2025

Exercise 3 (total points: 90)

Due date: April 15 , Time: 23:59 Hrs. (Hard deadline! No extensions

1 Windows Active Directory Exploit (Total points: 60)

This exercise simulates a common attack vector in Windows Active Directory environments: capturing and replaying NTLM hashes to escalate privileges. You will use a controlled lab (VM) environment with a Windows Server 2019 AD domain controller and two Windows 10 clients to explore vulnerabilities in SMB authentication and practice ethical hacking techniques.

You are being provided with three VMs. One is a Windows Server 2019 with AD configured. The others are two clients that run Windows 10 enterprise clients. The AD domain name is MYROOT.local. Both client VMs are configured to connect and rely on the AD server for user authentication. One such domain user is `'elvisp'` and uses the password `'@pr11f00l'`. To login to the domain you could log in as `elvisp@myroot.local` using the above password.

Start the provided VMs (Windows Server 2019 and two Windows 10 clients) in your virtualization software. Ensure all VMs are on the same network and can ping each other. Log in to one Windows 10 client using the credentials `elvisp@myroot.local` with password `'@pr11f00l'` to confirm AD authentication works.

You can use either of the windows client VMs to log in with these credentials. Once logged with the said machine, your task would be to discover other domain users and hosts and replay their NTLM hashes for authenticating on their behalf.

It must be noted that all the VMs have SMB service enabled, however, only Windows client machines have SMB signing disabled/unenforced. These could be your *targets* (described below).

You have all the tools at your disposal: `arpscan` to discover other hosts of the domain, `PowerView` for knowing the domain controller IP address and other domain user names, `crackmapexec` for discovering SMB shares and replaying hashes, wherever applicable.

Your main tasks would be run a `Linux` VM that runs `responder.py` and `ntlmreplayx.py` (*target* set to the other windows client) to capture the hashed passwords. Use a Kali Linux/Blackarch tools on your Linux VM as your attack machine. Ensure it is on the same network as the Windows VMs.

To trigger the hash transmission, you require to access the SMB share on the domain controller. This can be achieved through the address bar of Windows Explorer on one of the client machines (on which you logged on through the destination URL `'\\<SMB server IP>'`). More specifically, on the Windows 10 client where you logged in as `elvisp`, open File Explorer and enter <Domain Controller IP>\share in the address bar. This will trigger an NTLM authentication attempt that Responder can capture.

To respond to hash transmissions you could use `responder.py` (*e.g.*, responder.py -I <interface>).

To replay hashes of other users you could use `ntlmreplayx.py` (*e.g.*, `ntlmreplayx.py` -t <target IP>).

If the replay succeeds with `ntlmreplayx.py`, you may be able to discover other users' hashed passwords. Thereafter, you can use these credentials (username and NTLM hashes) to log in as the other users. You can use `crackmapexec` along with `psexec` (for getting a powershell on the SMB share) for this.

Example: `crackmapexec smb` <target IP> -u <username> -H <NTLM hash> to test authentication, followed by psexec.py <username>@<target IP> -H <NTLM hash> to gain a shell."

To summarise, the following are your tasks:

1. Setup and run the windows clients and servers. Validate the credentials for user `elvisp`.

2. On a separate `Linux` VM, use tools described above to run `responder.py` and `ntlmreplayx.py` (with appropriate arguments) to login and capture the credentials of other users.

3. Use the credentials (username and hash) to login to the other windows host and read/write into the SMB share. You can use `crackmapexec` and `psexec` for this. Alternatively, you could also use metasploit's `auxiliary/scanner/smb/smb_login` for the same.

4. Use `hashcat` to crack the discovered hash and print them.

What to submit and rubrics.

1. A write up describing all that you did with the outcomes. Specifically,

   - Command-line with arguments used for running `responder.py` and `ntlmreplayx.py` with their semantics, with appropriate screenshots (10 points if arguments are correctly supplied with their correct semantics described; 5 points if only some of the commands and arguments are correctly supplied; 0 if commands are arguments are not explained correctly.).

   - Command-line with arguments used for running `crackmapexec` and `psexec` or `metaspolit auxiliary/scanner/smb/smb_login` to use the obtained credentials from the previous step, with their semantics and appropriate screenshots (20 points if arguments are correctly supplied with their correct semantics described; 10 points if only some of the commands and arguments are correctly supplied; 0 if commands are arguments are not explained correctly.).

- Comamnd-line with arguments used for cracking the hash using `hashcat` with appropriate semantics and screenshots (20 points if arguments are correctly supplied with their correct semantics described; 10 points if only some of the commands and arguments are correctly supplied; 0 if commands are arguments are not explained correctly.).
- Correct password for the discovered user(s) (10 points if correct password is discovered; 5 points if only attempt is correct with the right commands and arguments but the correct password is not identified).

LibreSwan IPSec/IKE

The objective of this part is to familiarize you with using LibreSwan IPSec/IKEv2 protocol. For this you need to create a set-up involving four VMs, as shown in figure 1. The VMs 2 and 3 are supposed to VPN Gateways. By default the VM1 and VM4 are should not know about one another and should not be able to ping one another. The VM2 and VM3 are however enabled to forward IP traffic.

You need to install LibreSwan on VM2 and VM3. Configure LibreSwan on both the VM2 and VM3. They should be configured to establish mutual authenticated connection through X.509 public key certificates (self signed).

Once established, the tunnel should allow the VM1 to ping VM4 WITHOUT changing the underlying routing table entries. Capture the traffic between VM2 and VM3 showing the IKE tunnel setup and the encrypted ICMP echo (ping) messages being transported as ESP packets.

It must be noted that the traffic egressing the gateway VM3 and arriving VM4 must bear the source address of the VM3 when arriving to VM4.

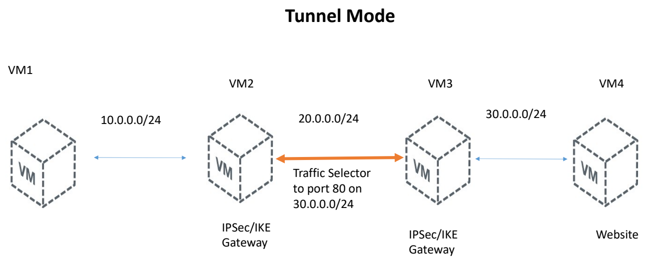

Figure 1: IPSec/IKE tunnel which allows access to port 80 on the webserver VM

1.1 What to submit:

1. A write up describing all that you did with the outcomes. Specifically,

   - Description of the commands that were ran for configuring the IPSec/IKE tunnel for both tunnel and transport modes, including details of the commands used to setup the tunnel (15 points if arguments are correctly supplied with their correct semantics described; 7.5 points if only some of the commands and arguments are correctly supplied; 0 if commands are arguments are not explained correctly.) .
   - `Wireshark` screenshot showing the traffic between the client and server and at their respective VPN gateways and between gateways (15 points if `wireshark` output shows the appropriate IP addresses of the packets between gateways and at the end points; 7.5 points if correct commands were used to configure but for some reason the correct IP addresses of packets is not visible).