SPECTEROPS

NTLM

The Legacy Protocol That Won’t Die

Elad Shamir – Director of Research

NTLM 101

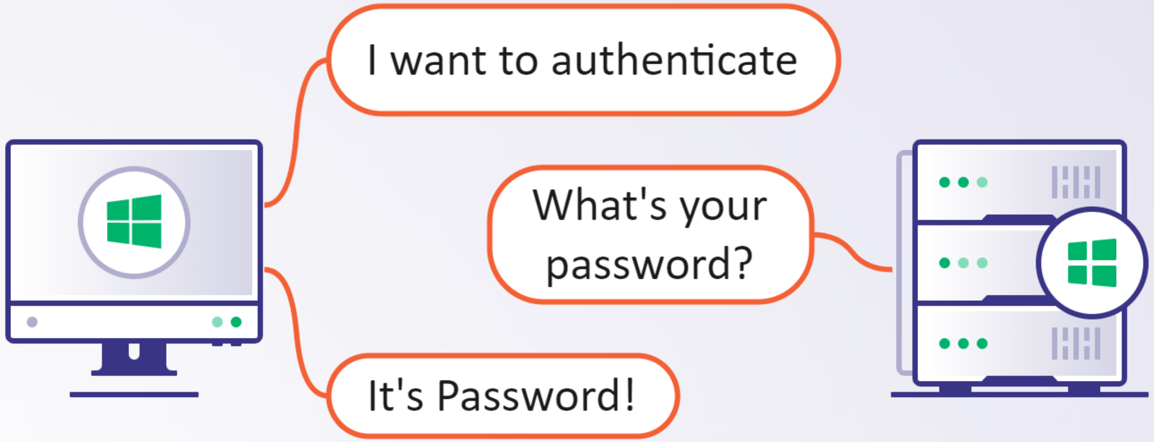
Background

- A legacy authentication protocol for Windows environments
- Introduced in **1993** as the successor to “LAN Manager”
  - NTLM = New Technology LAN Manager
- Kerberos is the primary authentication protocol in AD, but NTLM is still alive and kicking
- Numerous vulnerabilities and attacks affected NTLM over the years
- As of 2010, Microsoft recommends avoiding NTLM

NTLM 101

Designing an Authentication Protocol

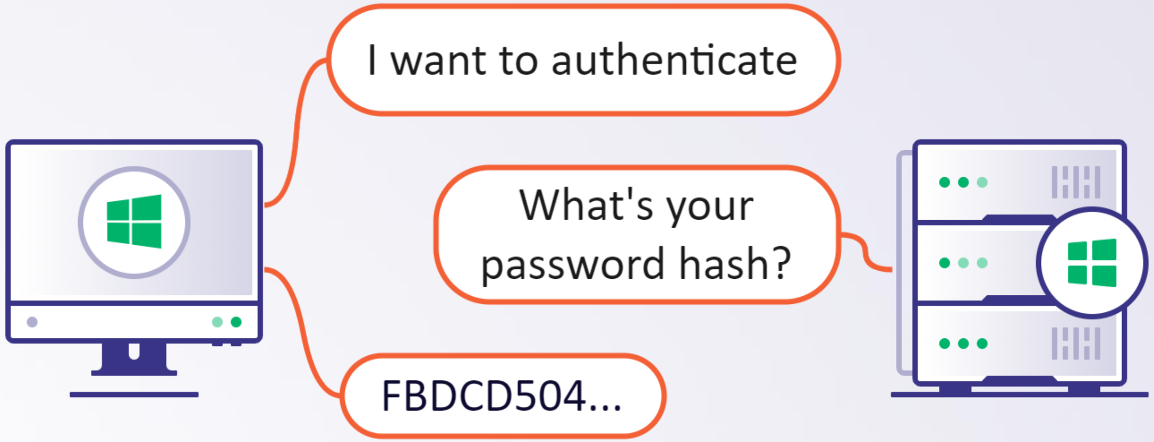
- **Solution: Send the password to the server**
- Problems:
  - Can be intercepted by MitM attackers
  - The password is disclosed to the server



NTLM 101

Designing an Authentication Protocol

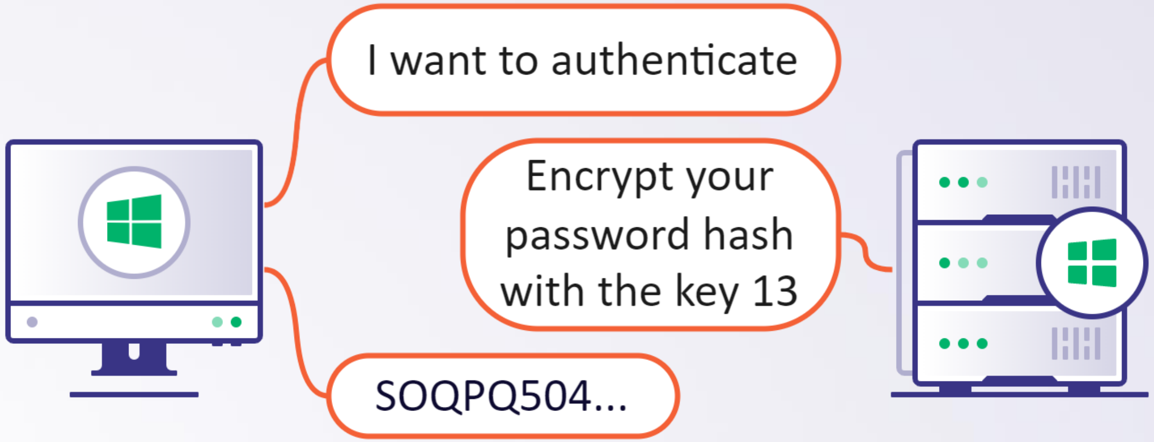
- **Solution: Hash the password**
- Problems:
  - Can be intercepted by MitM attackers
    - May be cracked
    - No need to crack
      - Replay attacks



NTLM 101

Designing an Authentication Protocol

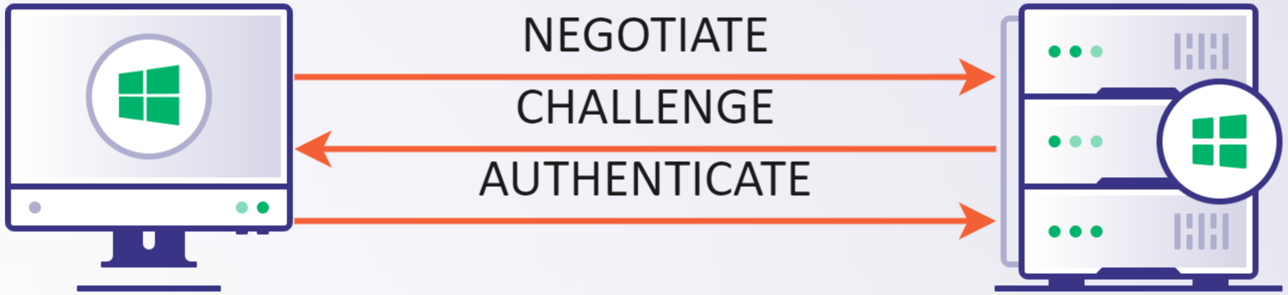
- **Solution: Challenge-Response**
- Problems:
  - Can be intercepted by MitM attackers
    - May be cracked
    - No need to crack
      - Pass the Hash (PtH) – if you have it
      - Relay attacks



NTLM 101

The Basics

- **NEGOTIATE:** Initiation, client security flags
- **CHALLENGE:** 8-byte nonce, server security flags
- **AUTHENTICATE:** Client security flags, **cryptographically generated response**


NEGOTIATE

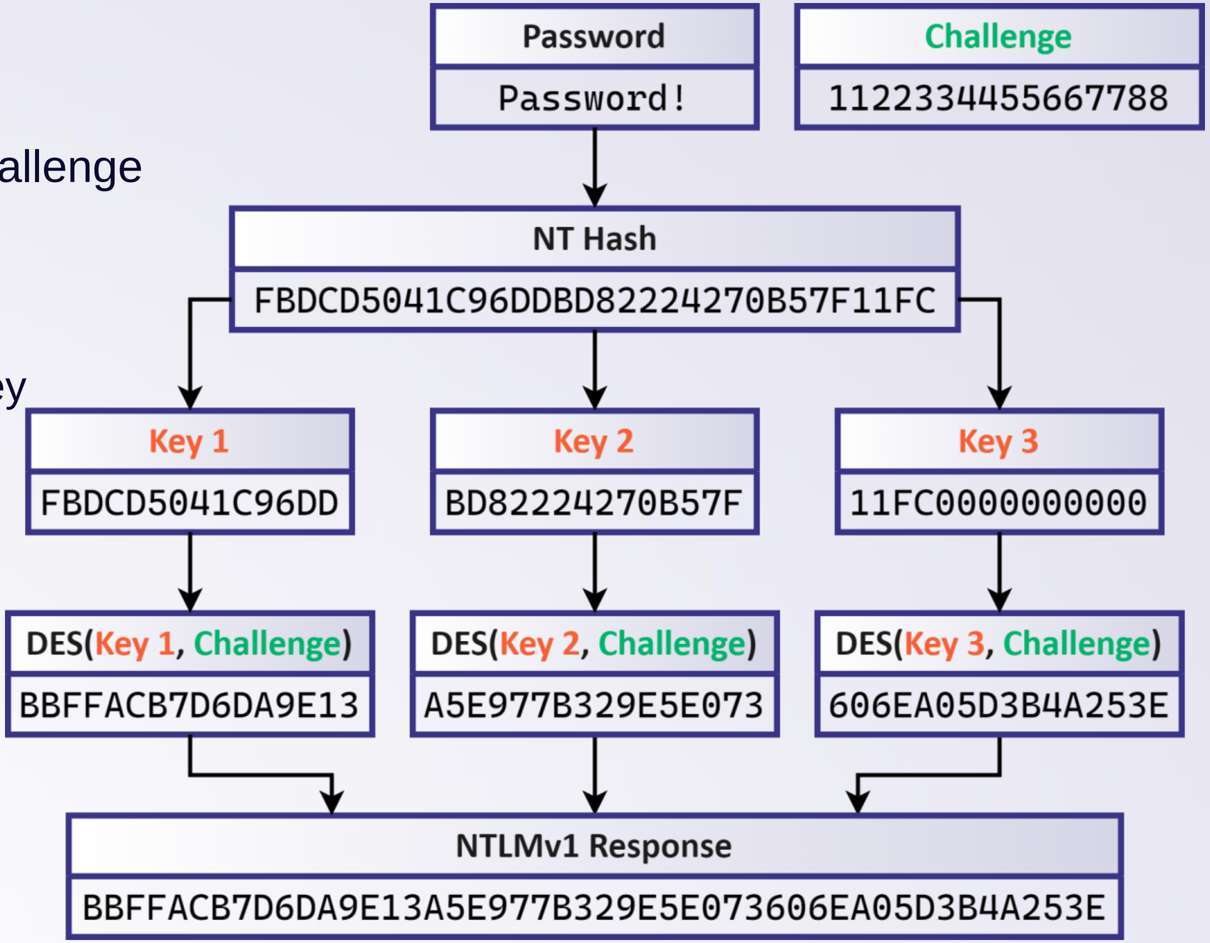
CHALLENGE

AUTHENTICATE

NTLM 101

NTLMv1 Response

- The NT hash of the password is 16-byte long
- The hash is used as the key to DES encrypt the challenge
- The challenge is encrypted three times:
  - With the first 7 bytes of the NT hash as the key
  - With the second 7 bytes of the NT hash as the key
  - With the last 2 bytes of the NT hash as the key
- The response is a concatenation of the three
- Cracking 7 bytes twice is exponentially easier than cracking 16 bytes at once
- Password reuse results in the same response to the same challenge
  - Rainbow tables are feasible



Password

Password!

Challenge

1122334455667788

NT Hash

FBDCD5041C96DDBD82224270B57F11FC

Key 1

FBDCD5041C96DD

Key 2

BD82224270B57F

Key 3

11FC0000000000

DES(Key 1, Challenge)

BBFFACB7D6DA9E13

DES(Key 2, Challenge)

A5E977B329E5E073

DES(Key 3, Challenge)

606EA05D3B4A253E

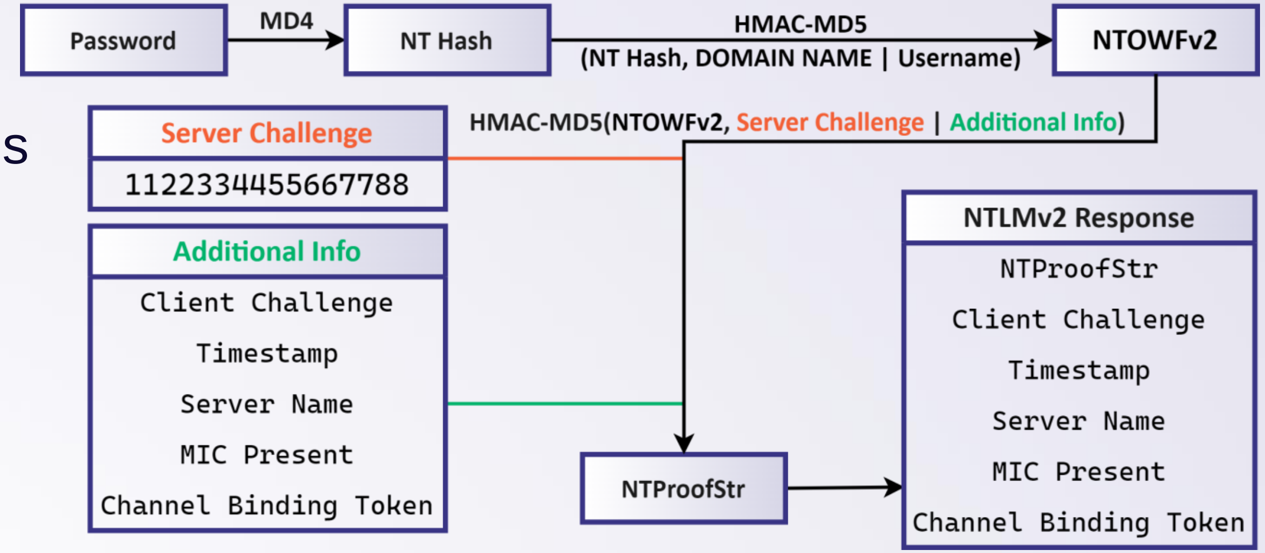
NTLMv1 Response

BBFFACB7D6DA9E13A5E977B329E5E073606EA05D3B4A253E

NTLM 101

NTLMv2 Response

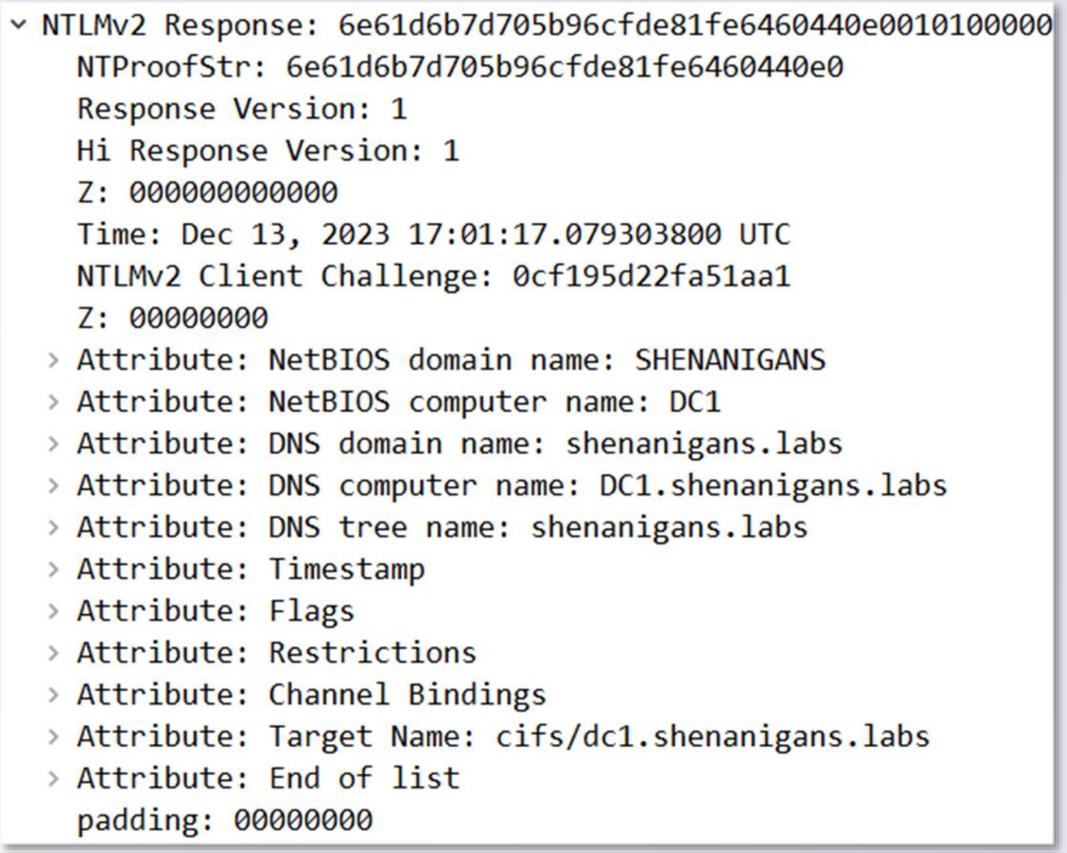
- Uses HMAC-MD5 instead of DES
- First, hashes the user and domain names with the NT hash
- The client generates an 8-byte nonce
- That hash is used to hash both challenges and additional session information



NTLM 101

NTLMv2 Response

- Uses HMAC-MD5 instead of DES
- First, hashes the user and domain names with the NT hash
- The client generates an 8-byte nonce
- That hash is used to hash both challenges and additional session information



```
˅ NTLMv2 Response: 6e61d6b7d705b96cfde81fe6460440e0010100000
    NTProofStr: 6e61d6b7d705b96cfde81fe6460440e0
    Response Version: 1
    Hi Response Version: 1
    Z: 000000000000
    Time: Dec 13, 2023 17:01:17.079303800 UTC
    NTLMv2 Client Challenge: 0cf195d22fa51aa1
    Z: 00000000
  › Attribute: NetBIOS domain name: SHENANIGANS
  › Attribute: NetBIOS computer name: DC1
  › Attribute: DNS domain name: shenanigans.labs
  › Attribute: DNS computer name: DC1.shenanigans.labs
  › Attribute: DNS tree name: shenanigans.labs
  › Attribute: Timestamp
  › Attribute: Flags
  › Attribute: Restrictions
  › Attribute: Channel Bindings
  › Attribute: Target Name: cifs/dc1.shenanigans.labs
  › Attribute: End of list
    padding: 00000000
```

NTLM 101

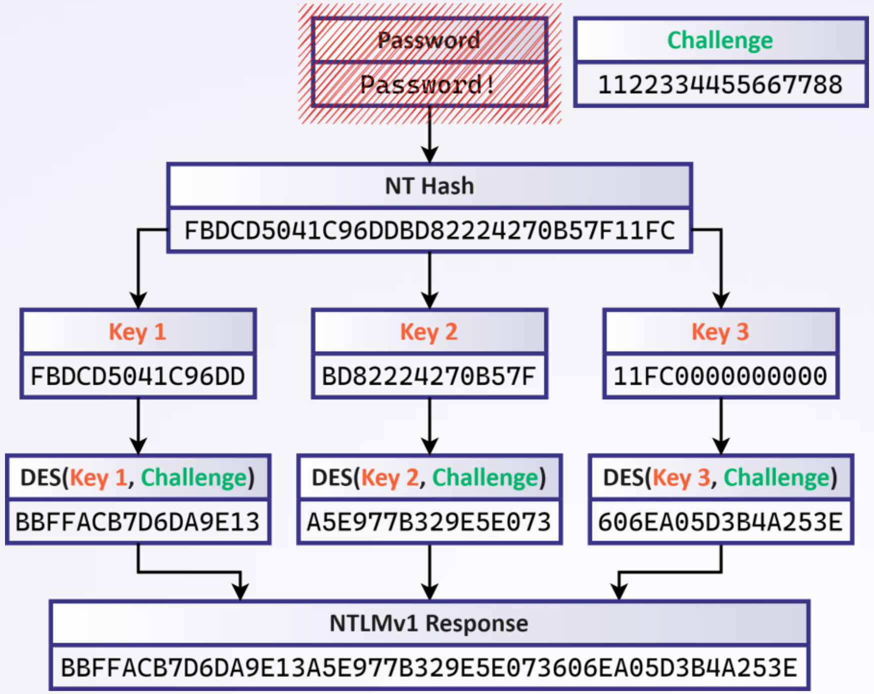
Password Attacks

- Both NTLMv1 and NTLMv2 challenge-responses can be used in offline password attacks
- The cleartext password *may* be cracked if it is insufficiently strong
    - NTLMv1 is easier to crack than NTLMv2
    - NT Hashes are easier to crack than NTLMv1
- The NT Hash ***<u>can</u>*** be recovered from NTLMv1 (e.g., using crack.sh)
    - The NT Hash is equivalent to the password due to Pass the Hash attacks

NTLM 101

Pass the Hash

- Both NTLMv1 and NTLMv2 allow generating a response using the NT Hash by skipping the first step in the process



Password

Password!

Challenge

1122334455667788

NT Hash

FBDCD5041C96DDBD82224270B57F11FC

Key 1

FBDCD5041C96DD

Key 2

BD82224270B57F

Key 3

11FC0000000000

DES(Key 1, Challenge)

BBFFACB7D6DA9E13

DES(Key 2, Challenge)

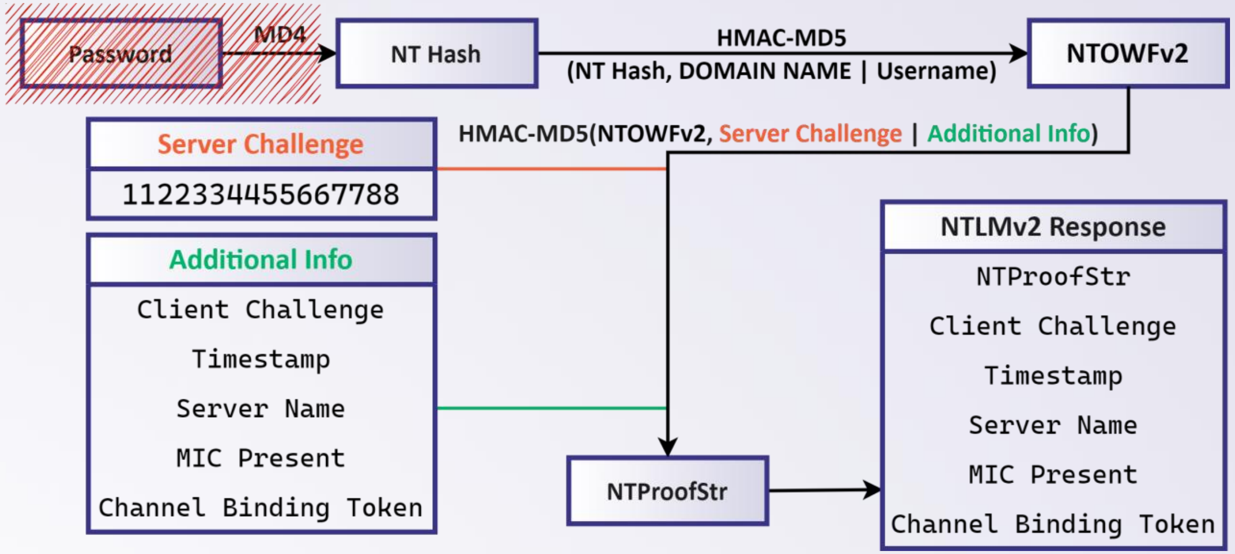
A5E977B329E5E073

DES(Key 3, Challenge)

606EA05D3B4A253E

NTLMv1 Response

BBFFACB7D6DA9E13A5E977B329E5E073606EA05D3B4A253E



Password

MD4

NT Hash

HMAC-MD5

(NT Hash, DOMAIN NAME | Username)

NTOWFv2

HMAC-MD5(NTOWFv2, Server Challenge | Additional Info)

Server Challenge

1122334455667788

Additional Info

Client Challenge

Timestamp

Server Name

MIC Present

Channel Binding Token

NTProofStr

NTLMv2 Response

NTProofStr

Client Challenge

Timestamp

Server Name

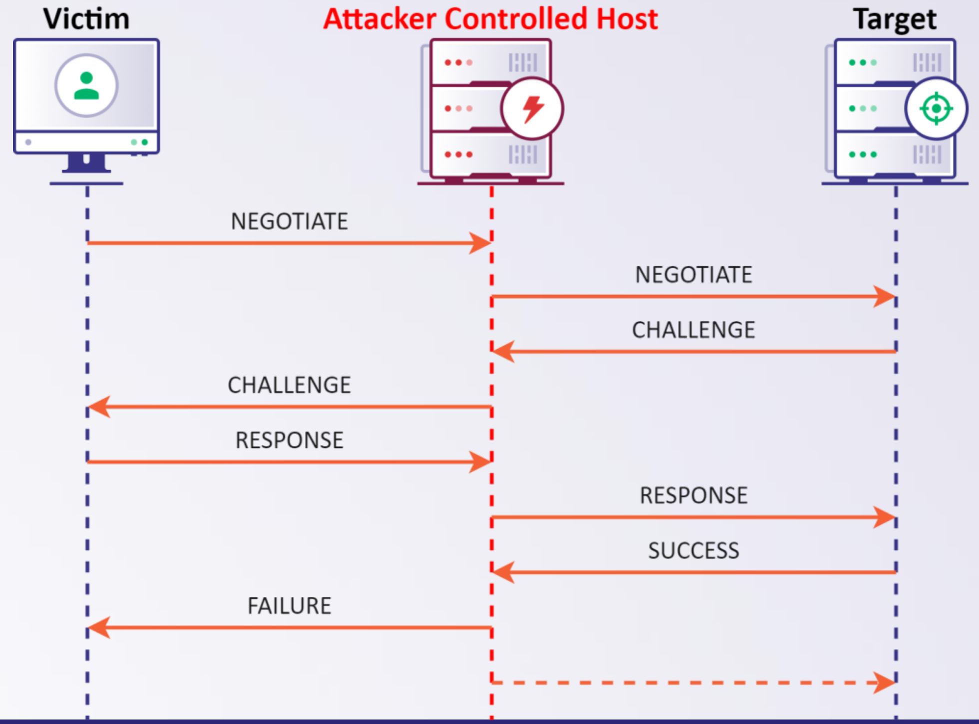
MIC Present

Channel Binding Token

The Elephant in the Room: NTLM Relay Attacks

Who Needs to Crack Passwords Anyway?

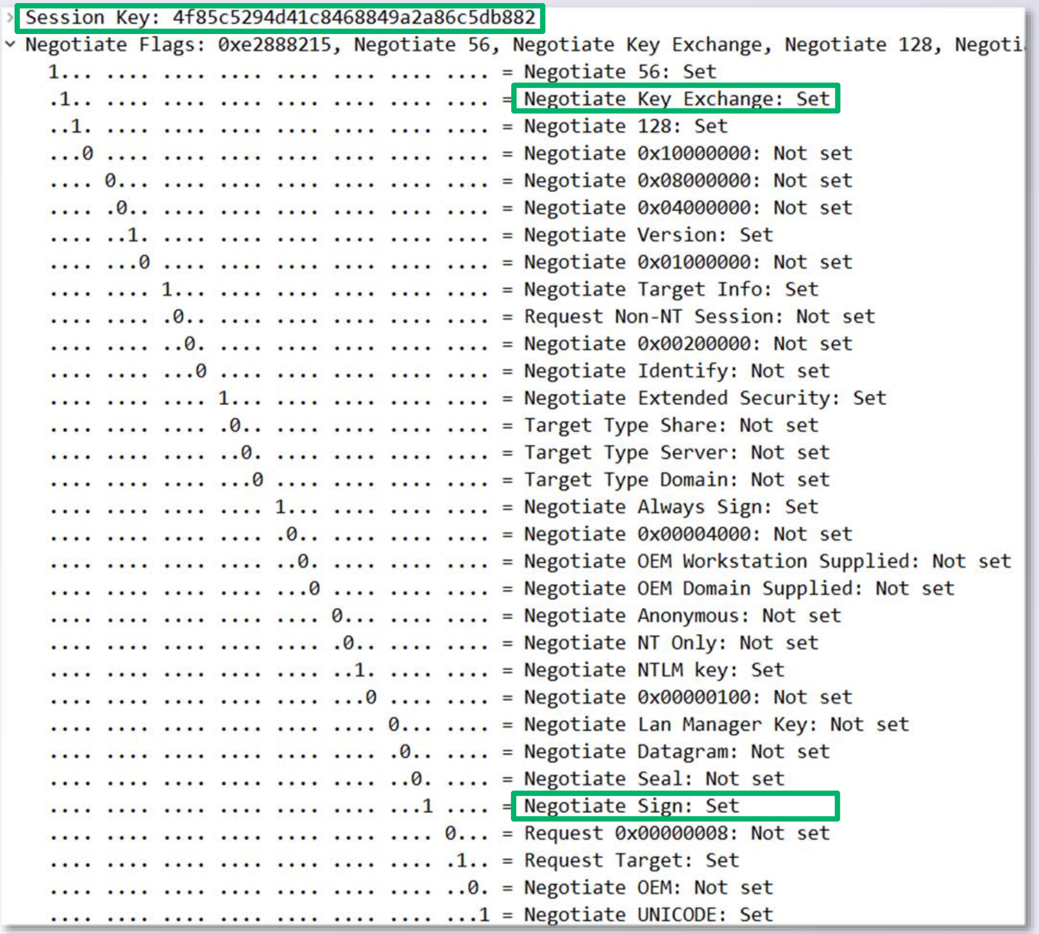
- Attackers in MitM position can relay the NTLM messages between the client and the server and establish a session
- No need to recover the NT hash or the password
- The root cause is that there’s no server authentication



The Elephant in the Room: NTLM Relay Attacks

Mitigations: Session Key Exchange

- A session key may be exchanged in the AUTHENTICATE message
- The key is RC4 encrypted
- A MitM attacker can’t obtain the session key without recovering the NT hash or the password
- A session is established, but the attacker can’t use it
  - Assuming the server enforces signing
- Can’t we remove that?

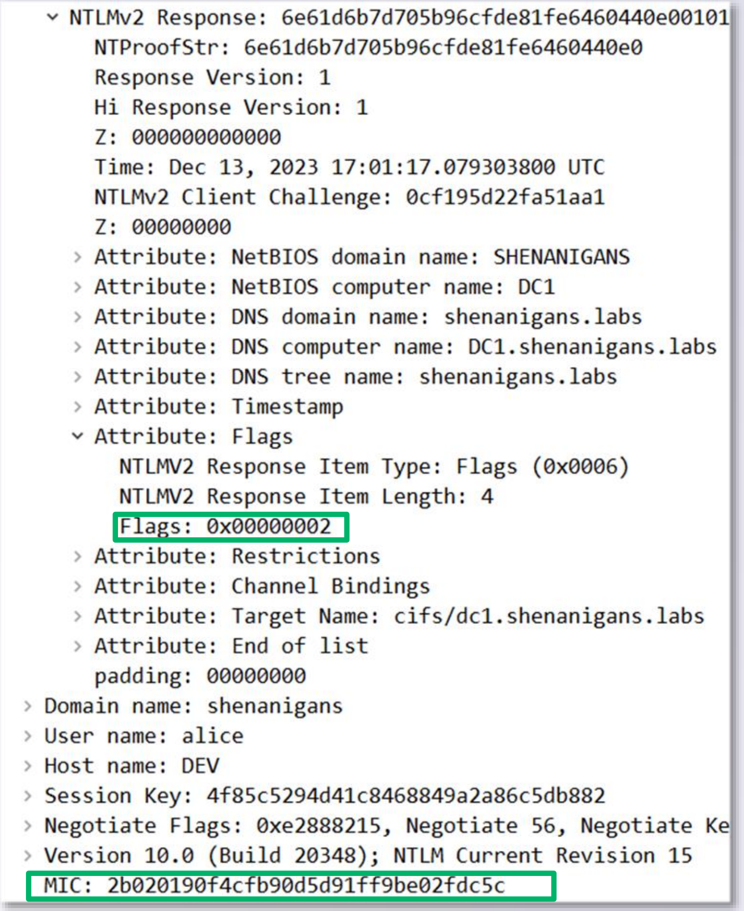


```
> Session Key: 4f85c5294d41c8468849a2a86c5db882
v Negotiate Flags: 0xe2888215, Negotiate 56, Negotiate Key Exchange, Negotiate 128, Negoti
    1... .... .... .... .... .... .... .... = Negotiate 56: Set
    .1.. .... .... .... .... .... .... .... = Negotiate Key Exchange: Set
    ..1. .... .... .... .... .... .... .... = Negotiate 128: Set
    ...0 .... .... .... .... .... .... .... = Negotiate 0x10000000: Not set
    .... 0... .... .... .... .... .... .... = Negotiate 0x08000000: Not set
    .... .0.. .... .... .... .... .... .... = Negotiate 0x04000000: Not set
    .... ..1. .... .... .... .... .... .... = Negotiate Version: Set
    .... ...0 .... .... .... .... .... .... = Negotiate 0x01000000: Not set
    .... .... 1... .... .... .... .... .... = Negotiate Target Info: Set
    .... .... .0.. .... .... .... .... .... = Request Non-NT Session: Not set
    .... .... ..0. .... .... .... .... .... = Negotiate 0x00200000: Not set
    .... .... ...0 .... .... .... .... .... = Negotiate Identify: Not set
    .... .... .... 1... .... .... .... .... = Negotiate Extended Security: Set
    .... .... .... .0.. .... .... .... .... = Target Type Share: Not set
    .... .... .... ..0. .... .... .... .... = Target Type Server: Not set
    .... .... .... ...0 .... .... .... .... = Target Type Domain: Not set
    .... .... .... .... 1... .... .... .... = Negotiate Always Sign: Set
    .... .... .... .... .0.. .... .... .... = Negotiate 0x00004000: Not set
    .... .... .... .... ..0. .... .... .... = Negotiate OEM Workstation Supplied: Not set
    .... .... .... .... ...0 .... .... .... = Negotiate OEM Domain Supplied: Not set
    .... .... .... .... .... 0... .... .... = Negotiate Anonymous: Not set
    .... .... .... .... .... .0.. .... .... = Negotiate NT Only: Not set
    .... .... .... .... .... ..1. .... .... = Negotiate NTLM key: Set
    .... .... .... .... .... ...0 .... .... = Negotiate 0x00000100: Not set
    .... .... .... .... .... .... 0... .... = Negotiate Lan Manager Key: Not set
    .... .... .... .... .... .... .0.. .... = Negotiate Datagram: Not set
    .... .... .... .... .... .... ..0. .... = Negotiate Seal: Not set
    .... .... .... .... .... .... ...1 .... = Negotiate Sign: Set
    .... .... .... .... .... .... .... 0... = Request 0x00000008: Not set
    .... .... .... .... .... .... .... .1.. = Request Target: Set
    .... .... .... .... .... .... .... ..0. = Negotiate OEM: Not set
    .... .... .... .... .... .... .... ...1 = Negotiate UNICODE: Set
```

The Elephant in the Room: NTLM Relay Attacks

Mitigations: MIC

- If the session key isn’t mandatory, why can’t a MitM attacker remove it?
- The Message Integrity Code (MIC) was introduced *later* to protect all three messages
  - All three messages are signed using the session key
  - Authentication fails if a single bit changes
- If the MIC isn’t mandatory, why can’t a MitM attacker remove it?
  - In NTLMv1, it is indeed possible!
  - In NTLMv2, an element indicating that the MIC is present is hashed into the response
    - If that element is removed, the response is no longer valid

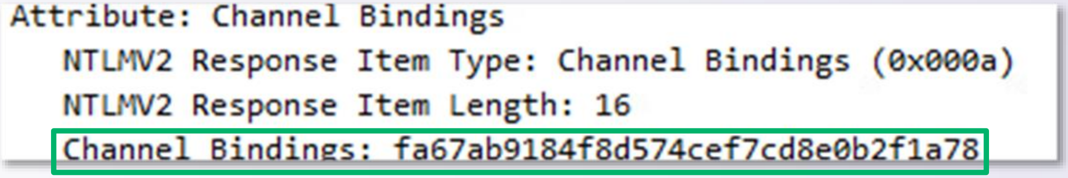


```
⌄ NTLMv2 Response: 6e61d6b7d705b96cfde81fe6460440e00101
    NTProofStr: 6e61d6b7d705b96cfde81fe6460440e0
    Response Version: 1
    Hi Response Version: 1
    Z: 000000000000
    Time: Dec 13, 2023 17:01:17.079303800 UTC
    NTLMv2 Client Challenge: 0cf195d22fa51aa1
    Z: 00000000
  › Attribute: NetBIOS domain name: SHENANIGANS
  › Attribute: NetBIOS computer name: DC1
  › Attribute: DNS domain name: shenanigans.labs
  › Attribute: DNS computer name: DC1.shenanigans.labs
  › Attribute: DNS tree name: shenanigans.labs
  › Attribute: Timestamp
  ⌄ Attribute: Flags
      NTLMV2 Response Item Type: Flags (0x0006)
      NTLMV2 Response Item Length: 4
      Flags: 0x00000002
  › Attribute: Restrictions
  › Attribute: Channel Bindings
  › Attribute: Target Name: cifs/dc1.shenanigans.labs
  › Attribute: End of list
    padding: 00000000
› Domain name: shenanigans
› User name: alice
› Host name: DEV
› Session Key: 4f85c5294d41c8468849a2a86c5db882
› Negotiate Flags: 0xe2888215, Negotiate 56, Negotiate Ke
› Version 10.0 (Build 20348); NTLM Current Revision 15
  MIC: 2b020190f4cfb90d5d91ff9be02fdc5c
```

The Elephant in the Room: NTLM Relay Attacks

Mitigations: Channel Binding

- A mechanism that binds a transport layer protocol (TLS) with an application layer protocol
- Helps ensure that the secure channel established in a lower layer is indeed the one used by the application
- A token from the server certificate hash is used to establish the TLS is hashed into the client’s NTLM response



```
Attribute: Channel Bindings
    NTLMV2 Response Item Type: Channel Bindings (0x000a)
    NTLMV2 Response Item Length: 16
    Channel Bindings: fa67ab9184f8d574cef7cd8e0b2f1a78
```

The Elephant in the Room: NTLM Relay Attacks

The Devil is in the Details

- NTLM Relay attacks are still viable in some configurations/scenarios
- Signing and Channel Binding can be enabled, required/enforced, or disabled
  - Separate per-service settings for clients and servers
  - **The server configuration and *implementation* are ultimately what matters**
- The mitigations aren’t always present
  - Some clients don’t support MIC (Win XP/2003, 3rd party applications/platforms)
  - Some clients are still using NTLMv1
  - Some clients don’t negotiate signing (WebClient/WebDAV)
- Implementation flaws introduce vulnerabilities, such as “Drop the MIC”, which allow bypassing mitigations if they are enabled but not enforced by the server

The Elephant in the Room: NTLM Relay Attacks

What are We Looking For?

Client Side

- Signing is not negotiated
  - WebClient (WebDAV) – Affects all Windows workstations by default
- NTLMv1 is enabled
  - Not enabled by default
  - Less than 1% of all NTLM traffic
- Older Windows versions and 3rd-party software that doesn’t implement MIC

Server Side

- Signing/Channel Binding is disabled (configuration), not enforced (configuration), or ignored when not negotiated (implementation)
- Organizations often enforce it only when supported by the client (where applicable)
  - The MIC prevents manipulation

Authentication Coercion

The Idea

- Why would a victim authenticate to us anyway?
    - Opportunistic approach – just sit and wait
    - Intentional approach – make it happen
        - “Bring Your Own Victim” (BYOV)
- Abuse mechanisms that allow coercing a client to connect to an arbitrary path
    - Make them connect to an attacker-controlled service
    - Require authentication
    - Tell them you support only NTLM

Authentication Coercion

Coercing User Account Authentication

- Trick the client to load resources from a path on an attacker-controlled host
  - Embed an image in an email or on a webpage
  - Create a file that attempts to load an embedded resource (Word, Excel, PDF, etc.)
  - Create a file that attempts to load an icon from the attacker host upon directory browsing (Search Connectors, Shortcuts, etc.)
  - Social engineering
- Coercing WebClient (HTTP) traffic is preferred (no signing is negotiated), but targeting WebDAV requires the WebClient service to be installed and running
  - It is installed by default on all Windows workstations
  - Some of the above coercion techniques trigger the service to start automatically

Authentication Coercion

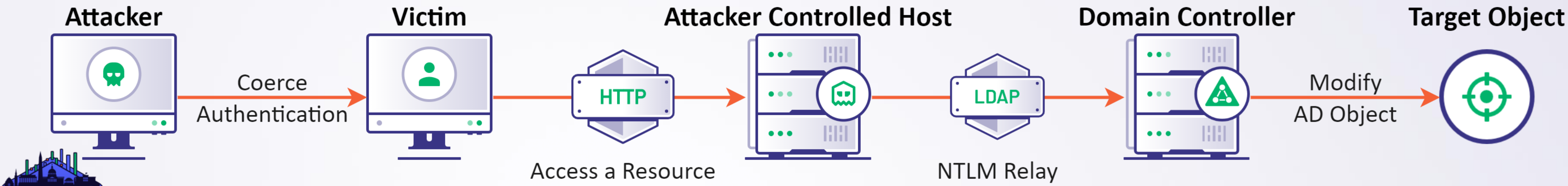
Coercing Computer Account Authentication

- Domain-joined hosts have “computer accounts”
- When a process running as SYSTEM/Network Service attempts to access a remote resource, it uses this account for authentication
- The “Printer Bug”
  - The first publicly disclosed remote authentication coercion technique
  - Request the target to send Print Spooler notifications to an attacker-controlled host
- PetitPotam
  - Ask the Encrypting File System to access a path on an attacker-controlled host
- These techniques support WebDAV UNC paths to get WebClient traffic, but they will not trigger starting the service if it is not already running

Advanced Real-World Scenarios

HTTP to LDAP(S)

- HTTP clients don’t normally negotiate signing
- Unless enforced, LDAP servers don’t care if the client didn’t negotiate signing
- What can attackers do with an LDAP session?
  - Privileged accounts can take over AD objects (add users to groups, grant access, etc.)
  - Computer accounts can modify *some* attributes of their own account
    - Configure Resource-Based Constrained Delegation, “Shadow Credentials”, SPN-jacking, etc.



Live Demo!

Advanced Real-World Scenarios

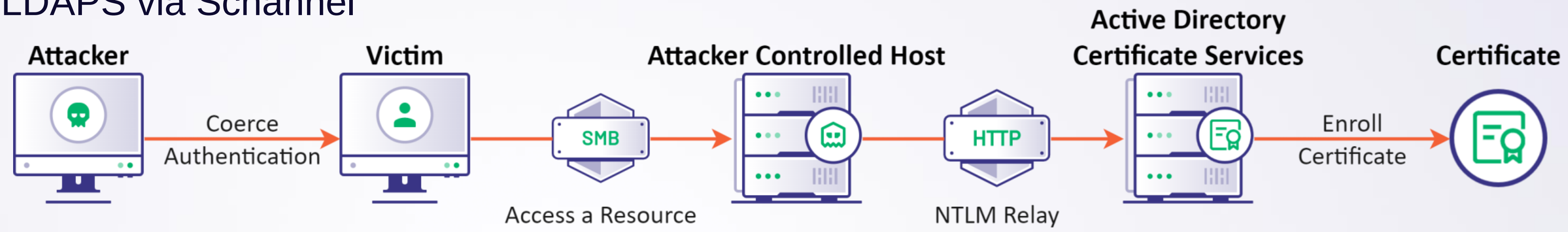
HTTP to LDAP(S): Bypassing Mitigations

- LDAP can enforce Signing and Channel Binding
- Signing requires all messages to be signed
  - LDAPS traffic is considered signed due to TLS
- Channel Binding binds the NTLM exchange to the server certificate
  - Establishing a session over LDAP with StartTLS bypasses channel binding
    - Authentication happens before the secure channel is created – nothing to bind to
- If both are enforced, this scenario is not viable

Advanced Real-World Scenarios

Certified Pre-Owned: ESC8

- Active Directory Certificate Services may have HTTP-based web enrollment endpoints
  - Allows clients to enroll certificates
  - By default, there are two interesting certificate templates published: “Machine” and a “User”
- By default, IIS doesn’t enforce Extended Protection for Authentication
- Relay for certificate enrollment is possible even from SMB clients
  - By default, affects servers too – including Domain Controllers!
- An attacker can use the certificate to obtain a Kerberos TGT with PKINIT or authenticate to LDAPS via Schannel



Attacker

Coerce Authentication

Victim

Access a Resource

Attacker Controlled Host

NTLM Relay

Active Directory Certificate Services

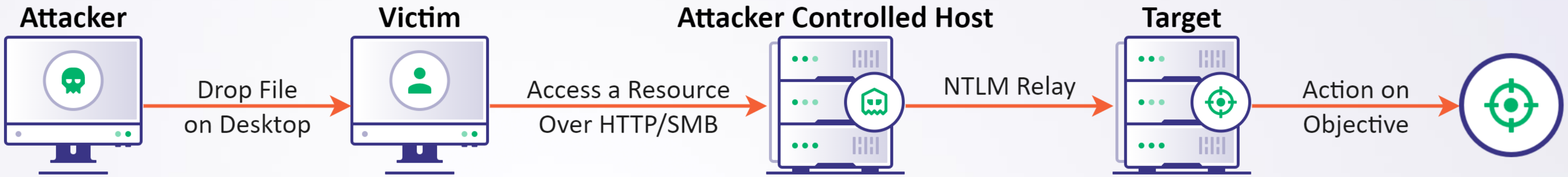
Enroll Certificate

Certificate

Advanced Real-World Scenarios

Remote Credential Abuse Without Code Execution

- Credential abuse of currently logged-on users typically involves executing code on the target host
  - Scraping LSASS (Mimikatz), token theft, dumping Kerberos tickets, “Internal Monologue”
- Achieving code execution via lateral movement is an “expensive” action for attackers
  - Remote file system access is more affordable
- Targeted user coercion with NTLM relay can achieve objectives without code execution
  - Drop a hidden authentication coercion file on the target’s desktop for a split second
  - Works even against locked workstations and disconnected RDP sessions



Live Demo!

Advanced Real-World Scenarios

SCCM Site Takeover

- System Center Configuration Manager allows deploying code and configuration to managed hosts
  - Can be abused for enumeration, privilege escalation, and lateral movement
- Servers running SCCM components have privileged access to SCCM components
- The Primary Site server has privileged access to the Site Database
  - Can coerce authentication from a Management Point server and relay it to the Site Database to elevate privileges and take over the entire site
- Site Servers are members of the SMS Admins group on each SMS Provider
  - Coerce authentication from a Site Server and relay it to an SMS Provider to elevate privileges and take over the entire site

Mitigations

Tactical Solution (I): Signing and Channel Binding Enforcement

- When signing and channel binding are both enforced, it’s effective
- But the configuration must be deployed on every single affected service
  - Attackers are likely to discover new affected services as needed
- It’s ***evidently*** a losing game

Mitigations

Tactical Solution (II): Protected Users

- The Protected Users Active Directory group offers a set of protections for its members both on the domain controller side and the device side
- On the device, the user’s NT hash is not stored in LSASS and NTLM authentication is blocked
- Opt-in: Only members are protected
- Computers and service accounts can’t/shouldn’t be added to the group

Mitigations

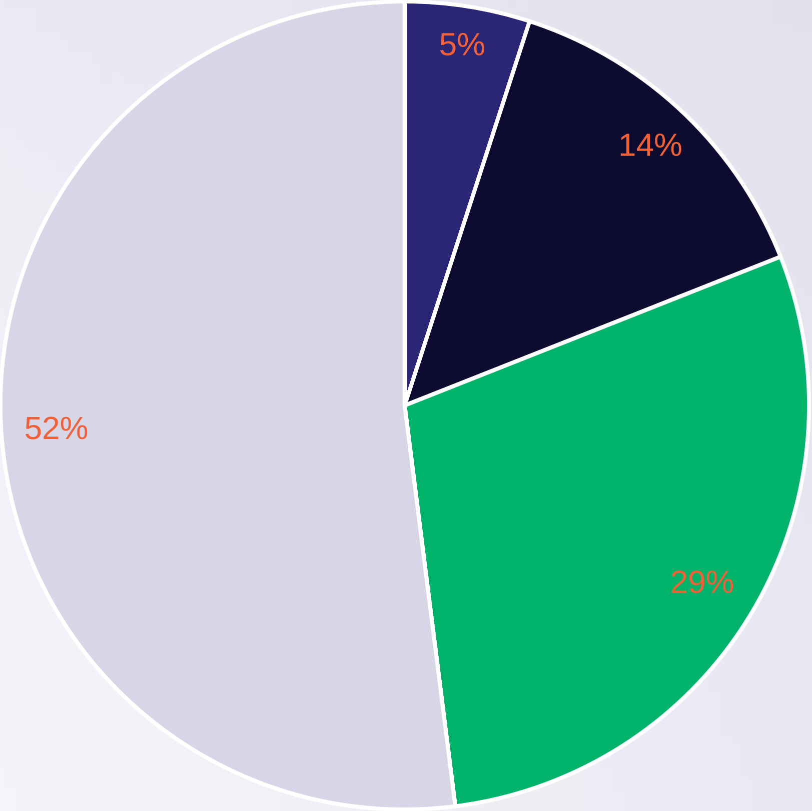
Strategic Solution: Audit NTLM Usage and Eliminate It

- There’s a configuration allowing the audit of all NTLM usage on a host
- It can help identify the culprits and address them
- Once all NTLM authentication is eliminated, NTLM can be disabled to mitigate these attacks altogether
  - I have never heard of an organization that disabled NTLM – and there’s a reason!
- If it were that simple, someone would have already done it
  - **Microsoft is working on it!**

Microsoft's Roadmap

Why is NTLM Still in Use Anyway?

- Microsoft's analysis of their telemetry shows the following statistics for the reasons for NTLM usage:
  - 5% due to no line of sight to a Domain Controller
    - Unable to request Kerberos tickets
  - 14% due to trying to authenticate to "unknown servers"
    - No SPN, IP address, etc.
  - 29% local user authentication
  - 52% hard-coded NTLM usage
    - Mostly by Windows components
      - Print Spooler is the main culprit
    - 10% by 3rd party software



Microsoft’s Roadmap

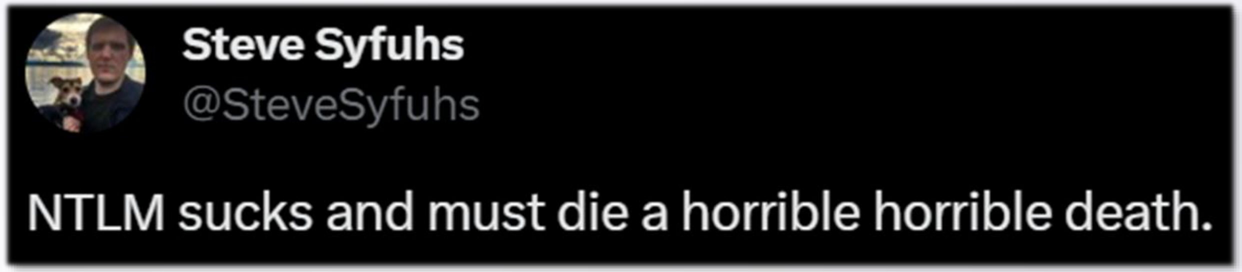
Facilitate NTLM Alternatives

- IAKERB (KDC Proxy) *should* address the line-of-sight issues
- A local KDC *should* solve local auth issues
- Configuring clients to allow IPs in SPN *should* solve some of the unknown server issues
- Microsoft is actively working towards solving hard-coded NTLM usage
  - Including reaching out to 3rd parties to work on solutions

Microsoft’s Roadmap

Ultimate Goal: NTLM Deprecation

- Microsoft’s Authentication Platform team is hopeful that NTLM will be disabled by default by 2028
  - Disabled, not deprecated
  - History tells us it will take longer than that
- That goal is at least 4 years away
  - In the meantime, AD/Windows environments are exposed (90% of organizations)



Steve Syfuhs
@SteveSyfuhs

NTLM sucks and must die a horrible horrible death.

Microsoft’s Roadmap

Is it Practical?

- Microsoft will likely avoid “pulling the plug” while 3rd party software still depends on it
  - What’s the threshold?
- Sysadmins are likely to reenable NTLM as a first step in troubleshooting issues
- Can attackers apply the same/similar attacks to NTLM’s replacement?
  - Plethora of attacks against Kerberos
  - Kerberoasting and AS-REP Roasting for password attacks
  - Kerberos Relay is a thing but with limited applicability

Questions?

SPECTEROPS

Thank You!

SPECTEROPS

SO-CON

2024