From Man-in-the-Middle to Privesc and RCE: Exploiting the Netlogon Protocol

Tom Tervoort

Outline

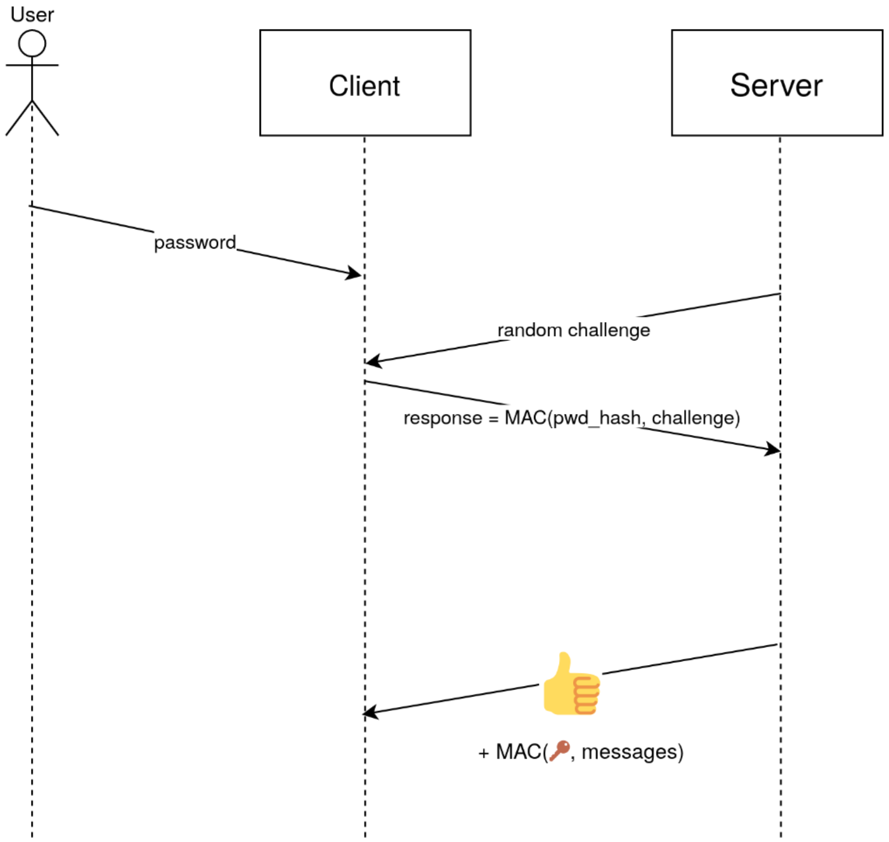
1. Introduction to NTLM and Netlogon
2. Netlogon vulnerabilities
3. New exploit

History of Windows AD authentication

- 1980’s, early 90’s: LM protocol
  - Password as DES key; very broken
- 1993: NTLMv1
  - MD4 + DES; divide and conquer attack
- 1998: NTLMv2
  - MD4 + HMAC-MD5; relay and offline brute-force issues
  - **Enabled by default; hard to get rid off**
- 2000: Kerberos (preferred option)

Secura

How NTLM works



User

Client

Server

password

random challenge

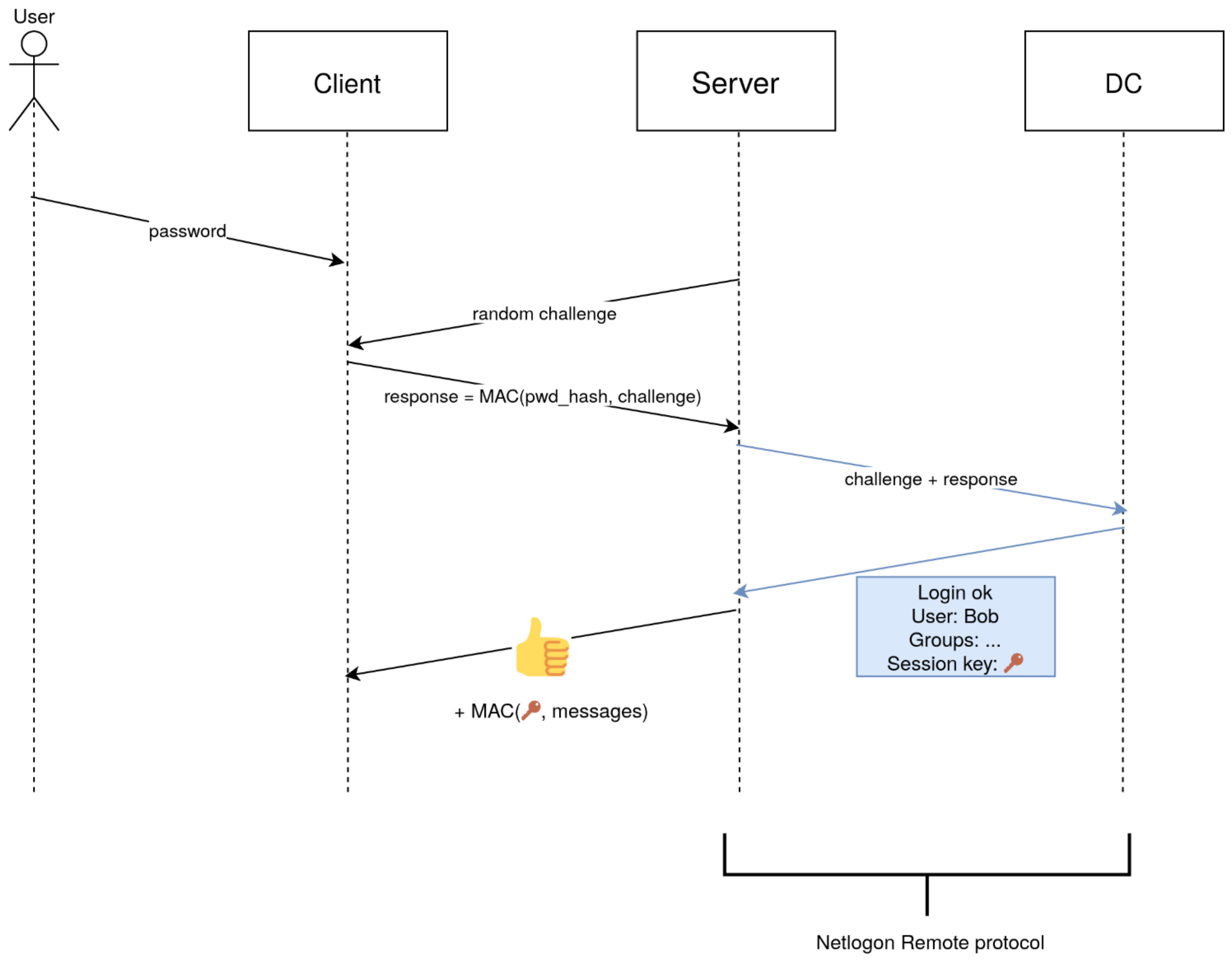
response = MAC(pwd_hash, challenge)

👍

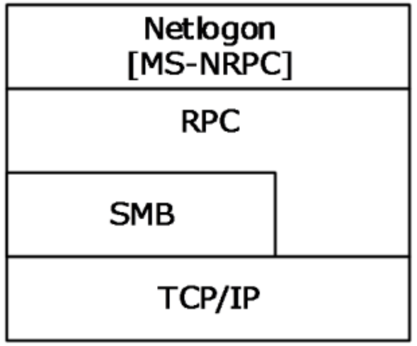
+ MAC(🔑, messages)

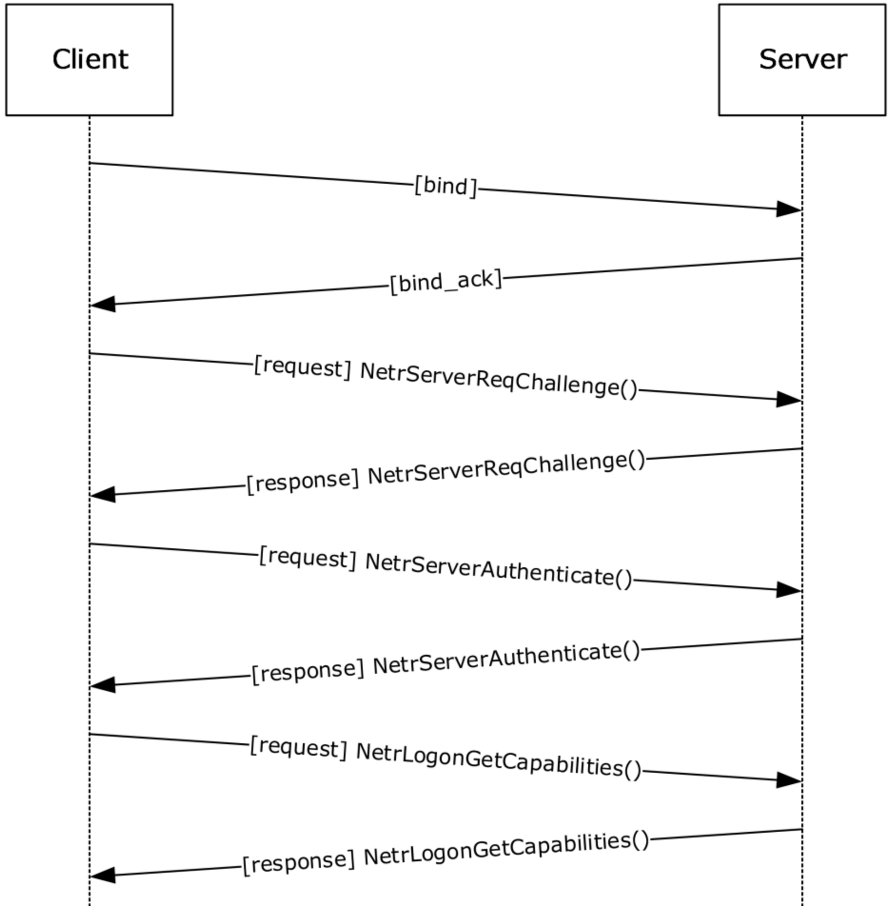
Session key 🔑 is derived from password hash

How NTLM works



The Netlogon Remote protocol




Client
Server
[bind]
[bind_ack]
[request] NetrServerReqChallenge()
[response] NetrServerReqChallenge()
[request] NetrServerAuthenticate()
[response] NetrServerAuthenticate()
[request] NetrLogonGetCapabilities()
[response] NetrLogonGetCapabilities()

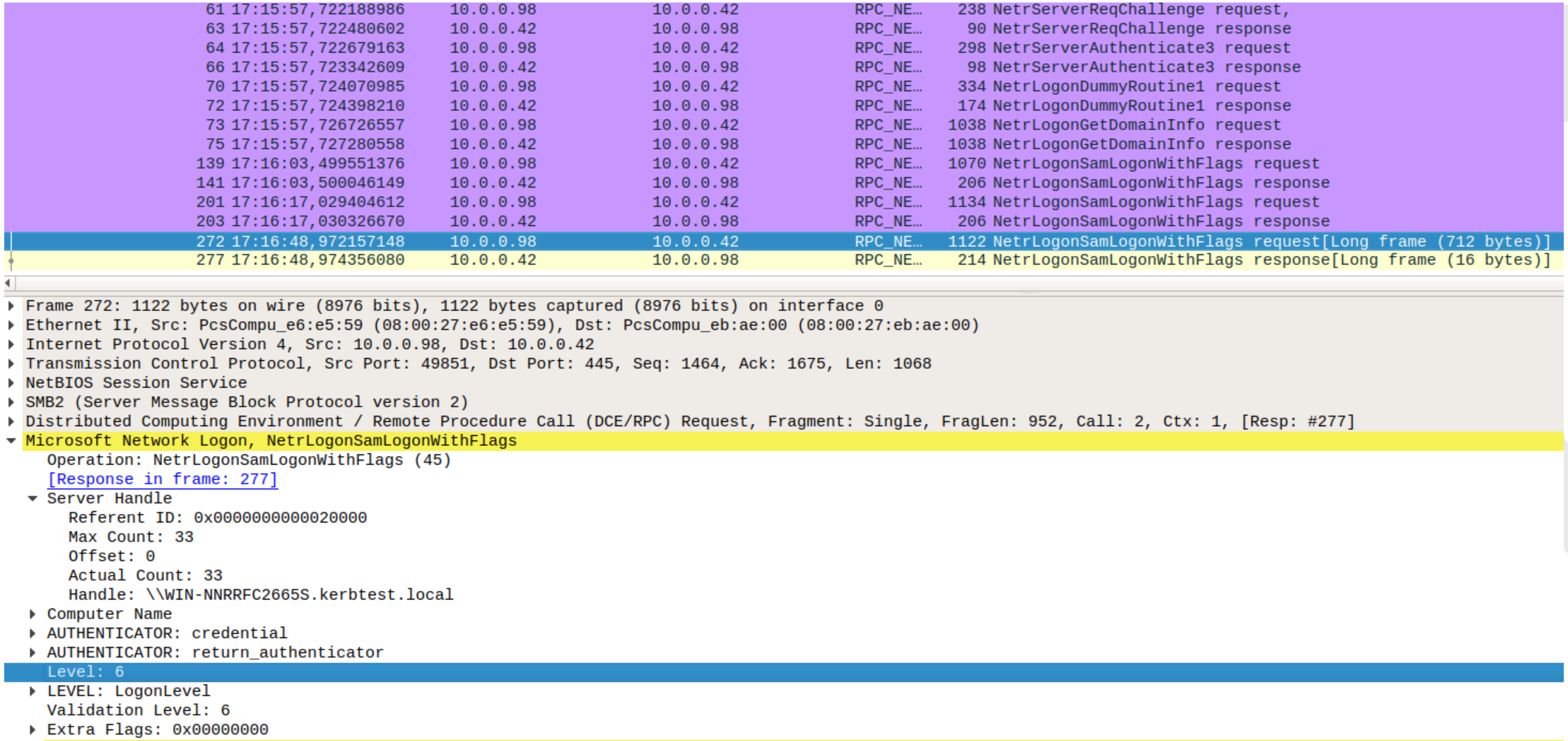
- MSRPC protocol
- Dynamic TCP port (portmapper)
- **Fallback: tunnel over SMB pipe**
- Computer password as shared secret for authentication and crypto
- Message signing/sealing: HMAC-SHA2 + encrypt with AES-CFB8
- Modern client will reject connection if server says it doesn’t support signing/sealing

Prior Netlogon vulnerabilities

- CVE-2015-0005
  - Computer A can submit NTLM handshake intended for computer B
  - Result: steal session key; relay attacks; SMB forgery/decryption…
- CVE-2019-1019
  - Strip computer name from NTLM challenge; client doesn't mind
  - DC sees no computer name; skips check
  - Same result

Secura

Huh? Why can I read this?



```
 61 17:15:57,722188986   10.0.0.98    10.0.0.42    RPC_NE…   238 NetrServerReqChallenge request,
 63 17:15:57,722480602   10.0.0.42    10.0.0.98    RPC_NE…    90 NetrServerReqChallenge response
 64 17:15:57,722679163   10.0.0.98    10.0.0.42    RPC_NE…   298 NetrServerAuthenticate3 request
 66 17:15:57,723342609   10.0.0.42    10.0.0.98    RPC_NE…    98 NetrServerAuthenticate3 response
 70 17:15:57,724070985   10.0.0.98    10.0.0.42    RPC_NE…   334 NetrLogonDummyRoutine1 request
 72 17:15:57,724398210   10.0.0.42    10.0.0.98    RPC_NE…   174 NetrLogonDummyRoutine1 response
 73 17:15:57,726726557   10.0.0.98    10.0.0.42    RPC_NE…  1038 NetrLogonGetDomainInfo request
 75 17:15:57,727280558   10.0.0.42    10.0.0.98    RPC_NE…  1038 NetrLogonGetDomainInfo response
139 17:16:03,499551376   10.0.0.98    10.0.0.42    RPC_NE…  1070 NetrLogonSamLogonWithFlags request
141 17:16:03,500046149   10.0.0.42    10.0.0.98    RPC_NE…   206 NetrLogonSamLogonWithFlags response
201 17:16:17,029404612   10.0.0.98    10.0.0.42    RPC_NE…  1134 NetrLogonSamLogonWithFlags request
203 17:16:17,030326670   10.0.0.42    10.0.0.98    RPC_NE…   206 NetrLogonSamLogonWithFlags response
272 17:16:48,972157148   10.0.0.98    10.0.0.42    RPC_NE…  1122 NetrLogonSamLogonWithFlags request[Long frame (712 bytes)]
277 17:16:48,974356080   10.0.0.42    10.0.0.98    RPC_NE…   214 NetrLogonSamLogonWithFlags response[Long frame (16 bytes)]
```

```
▸ Frame 272: 1122 bytes on wire (8976 bits), 1122 bytes captured (8976 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_e6:e5:59 (08:00:27:e6:e5:59), Dst: PcsCompu_eb:ae:00 (08:00:27:eb:ae:00)
▸ Internet Protocol Version 4, Src: 10.0.0.98, Dst: 10.0.0.42
▸ Transmission Control Protocol, Src Port: 49851, Dst Port: 445, Seq: 1464, Ack: 1675, Len: 1068
▸ NetBIOS Session Service
▸ SMB2 (Server Message Block Protocol version 2)
▸ Distributed Computing Environment / Remote Procedure Call (DCE/RPC) Request, Fragment: Single, FragLen: 952, Call: 2, Ctx: 1, [Resp: #277]
▾ Microsoft Network Logon, NetrLogonSamLogonWithFlags
    Operation: NetrLogonSamLogonWithFlags (45)
    [Response in frame: 277]
  ▾ Server Handle
      Referent ID: 0x0000000000020000
      Max Count: 33
      Offset: 0
      Actual Count: 33
      Handle: \\WIN-NNRRFC2665S.kerbtest.local
  ▸ Computer Name
  ▸ AUTHENTICATOR: credential
  ▸ AUTHENTICATOR: return_authenticator
    Level: 6
  ▸ LEVEL: LogonLevel
    Validation Level: 6
  ▸ Extra Flags: 0x00000000
```

Secura

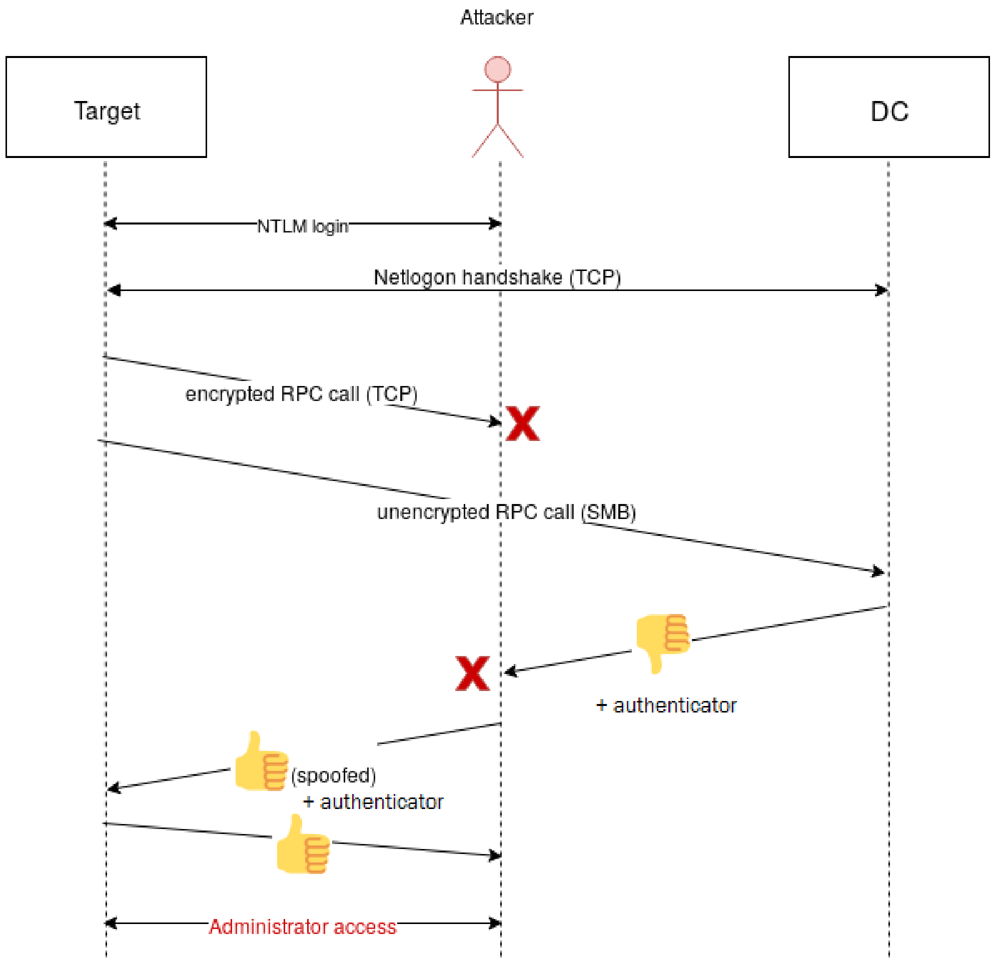
The vulnerability

- When client initiates session over TCP and falls back to SMB during a session, it “forgets” about the negotiated encryption method; server accepts this
- Calls still contain “authenticators”, but these do not depend on message content

- Result: **MitM can read/change messages!**

Secura

NTLM exploit



When you have a MitM position between a domain-joined computer and a DC, and this computer offers some service that accepts NTLM, **you can log in as any user, with any password, with any domain privileges**.

Made PoC with iptables and Impacket that logs anyone in as Domain Administrator (and thus **local admin** on the target) with the password “letmein”.

Secura

Who is vulnerable

- Typical corporate laptop has SMB service; can get **RCE as local admin** through e.g. the PsExec method
- MitM through e.g. ARP/NDP spoofing, fake Wi-Fi access point, physical access
- Stolen laptop scenario: Bitlocker (TPM Only) bypass

Secura

CVE-2019-1424

- Reported to Microsoft
- CVSS score: 8.1
- Recommendation:
    1. Clients should not stop encrypting on SMB fallback
    2. DC’s should not accept unencrypted calls after encryption is negotiated
- Patch released in November 2019

Q&A

Secura

Thanks for watching!

Tom Tervoort
tom.tervoort@secura.com

Secura