Diving into MSSQL madness

Cyberdefense

whoami /all

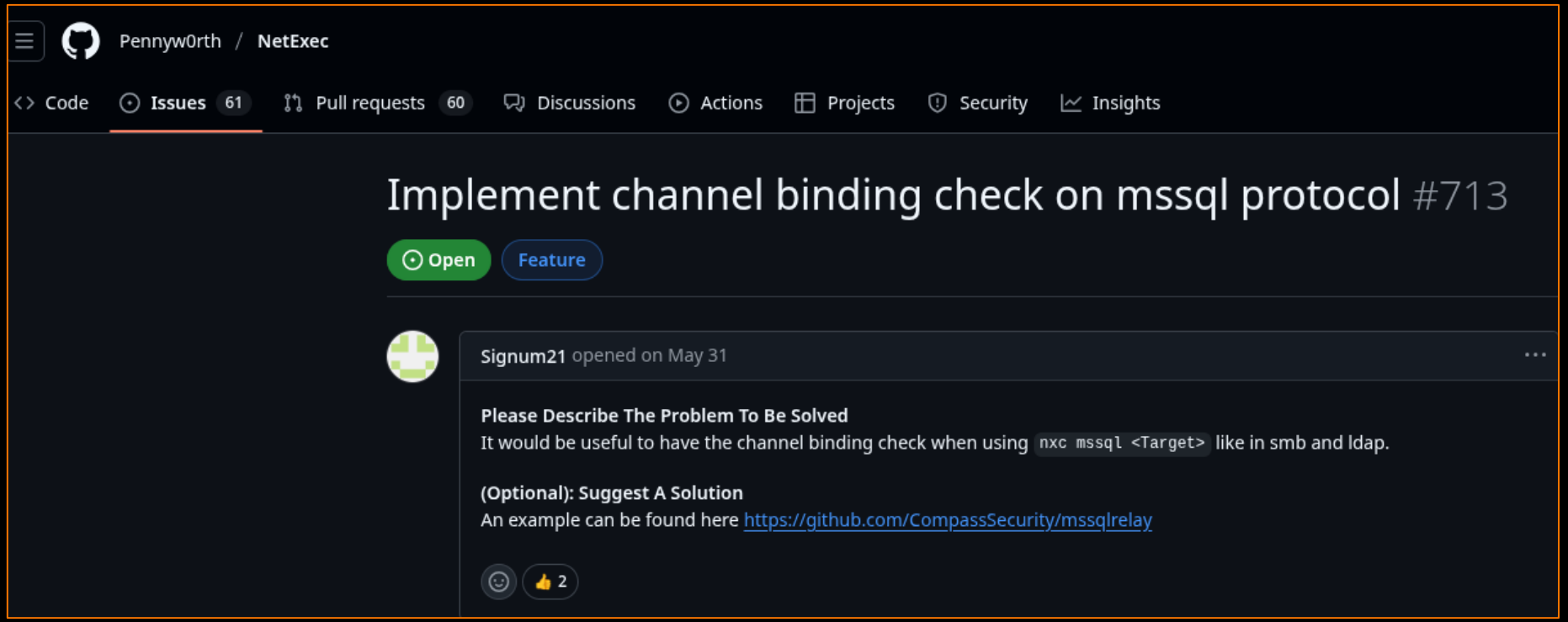
Éditer le profil

Aurélien Chalot

@Defte_

Hacker, sysadmin and security researcher @OrangeCyberdef 💻
Calisthenic enthousiast 💪 and wannabe philosopher 📖
🔥 Hide&Sec 🔥

1 / How it started ?



https://github.com/Pennyw0rth/NetExec/issues/713

cannatag / ldap3

Code Issues 178 Pull re

nentication #1087

owers:add_ntlm_channel_binding on

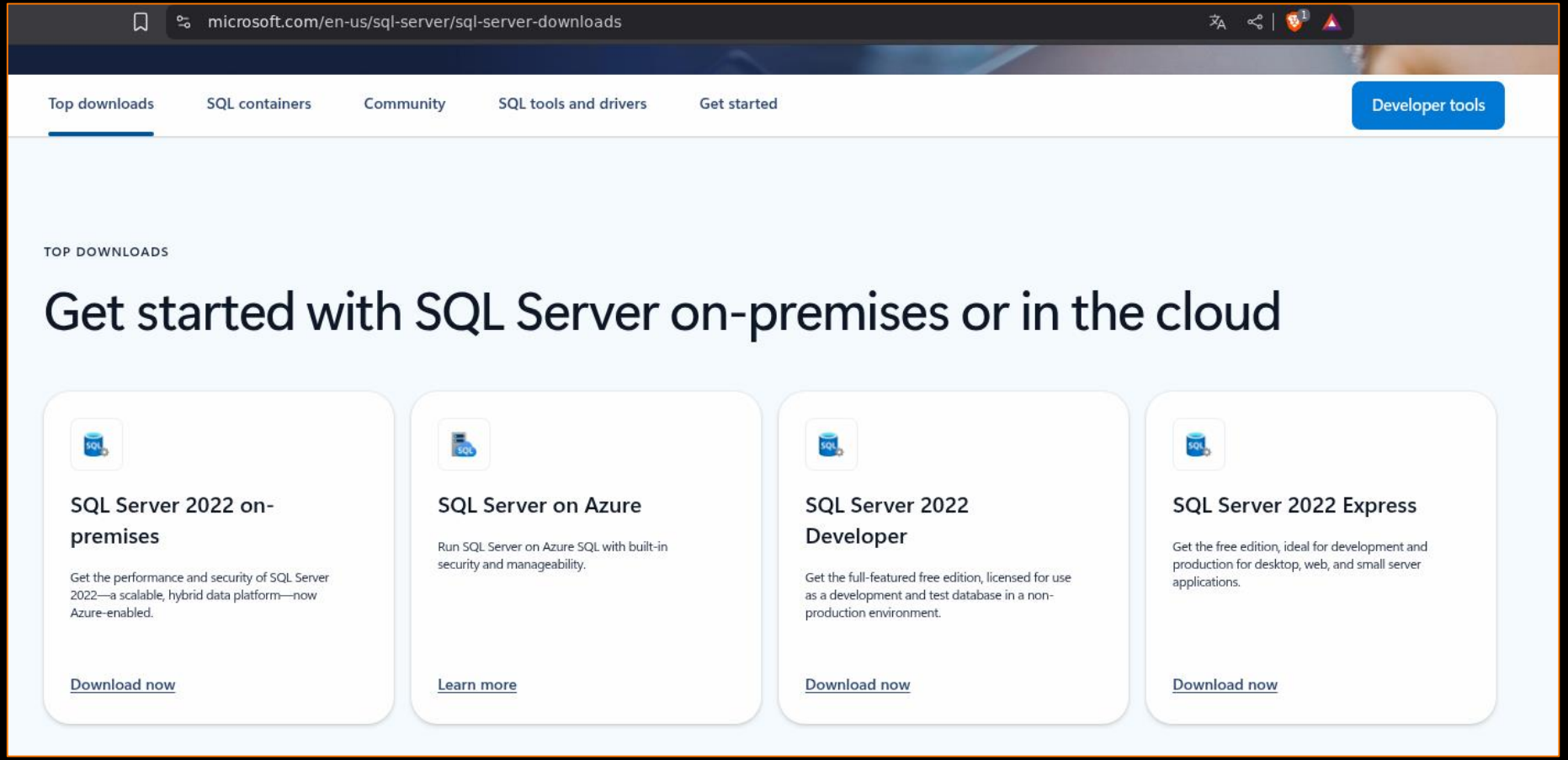
Contributor ...

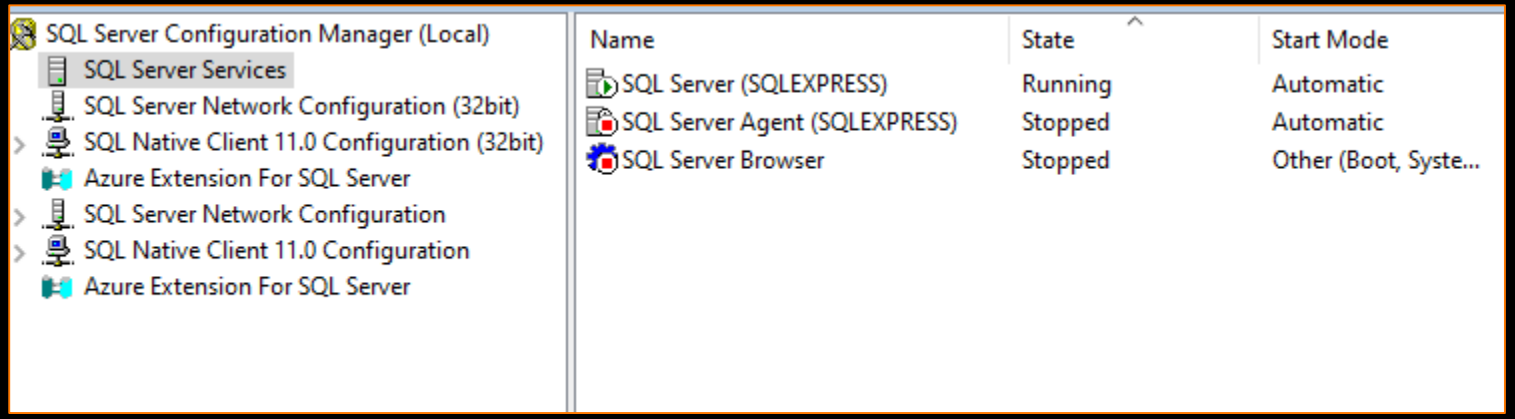
lea is to bind the outer secure
nel (NTLM here). This kind of
LM authentication.

R MS-NLMP 2.2.2.1 within the
ChannelBindings). The Value field
e sha256 of the server's certificate

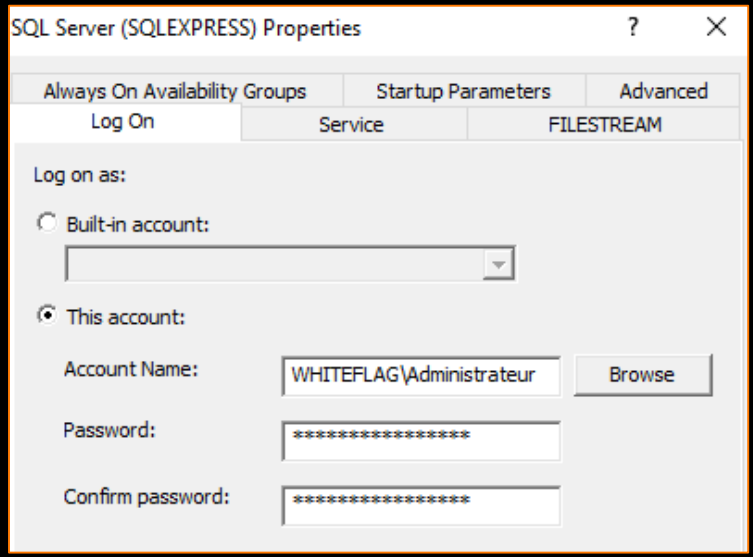
cts by @skelsec.

2 / First we'll need a setup


microsoft.com/en-us/sql-server/sql-server-downloads
Top downloads
SQL containers
Community
SQL tools and drivers
Get started
Developer tools
TOP DOWNLOADS
Get started with SQL Server on-premises or in the cloud
SQL Server 2022 on-premises
Get the performance and security of SQL Server 2022—a scalable, hybrid data platform—now Azure-enabled.
Download now
SQL Server on Azure
Run SQL Server on Azure SQL with built-in security and manageability.
Learn more
SQL Server 2022 Developer
Get the full-featured free edition, licensed for use as a development and test database in a non-production environment.
Download now
SQL Server 2022 Express
Get the free edition, ideal for development and production for desktop, web, and small server applications.
Download now



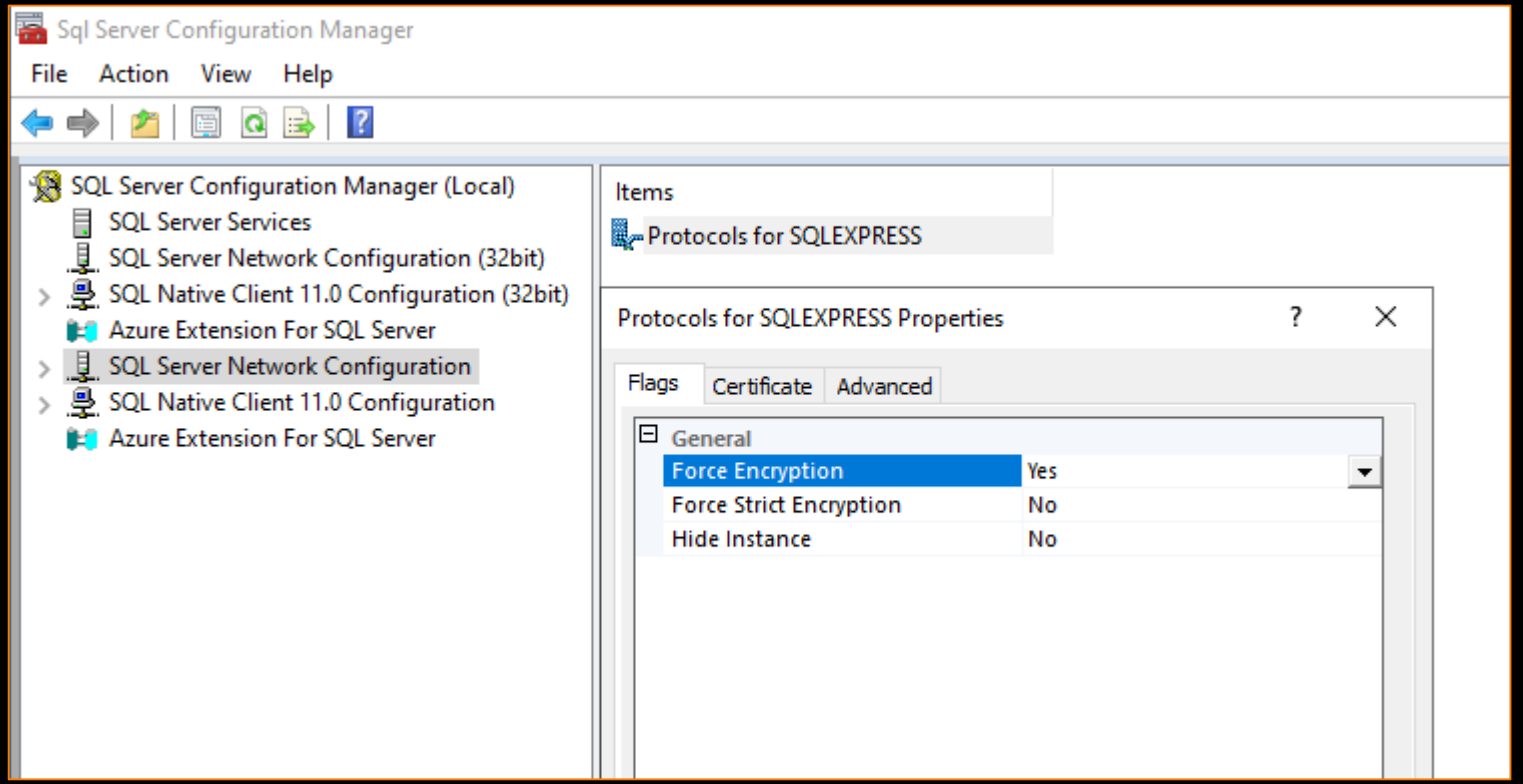
A simple database server running as the Administrator (RID500) domain account because who cares right ?



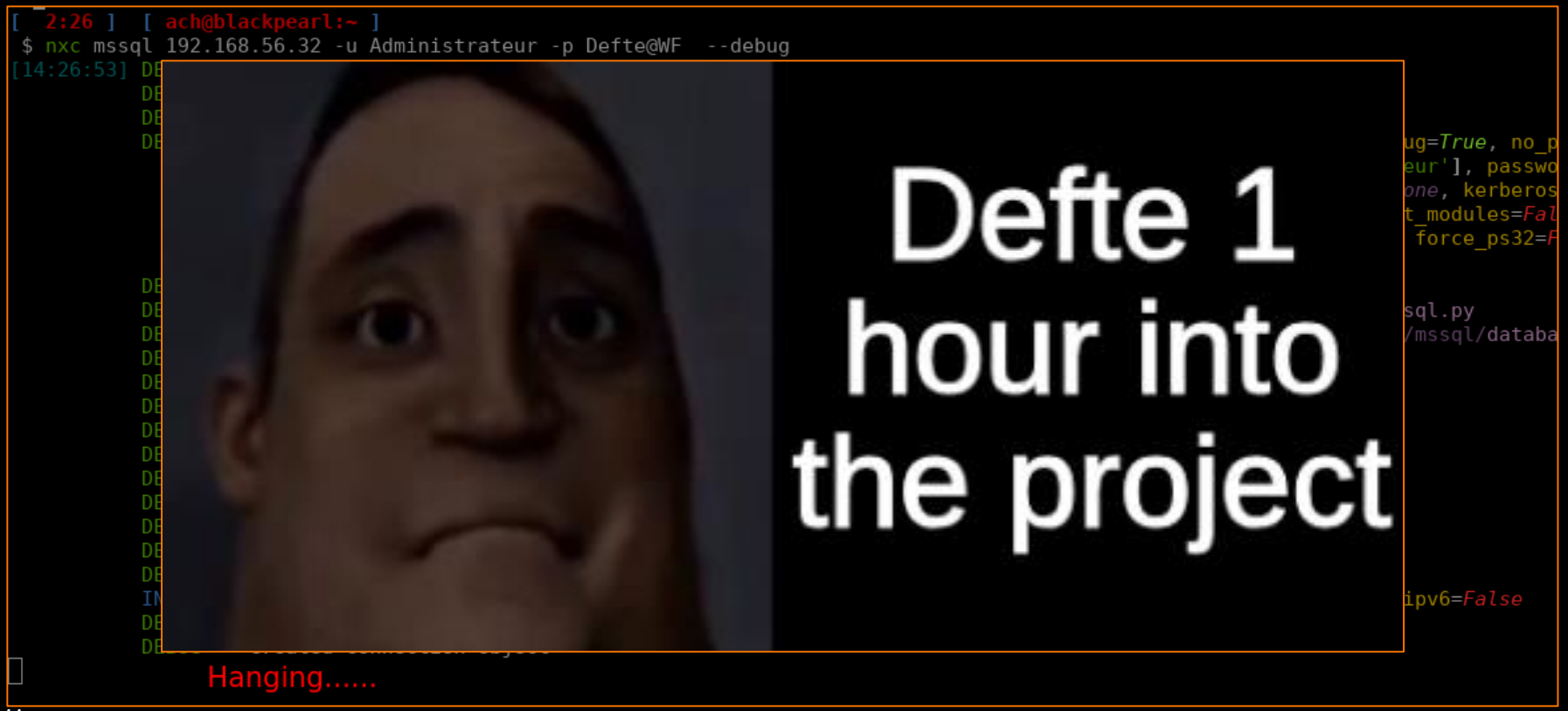
Let’s try NXC

```
  2:04 ]  [ ach@blackpearl:~ ]
$ nxc mssql 192.168.56.32 -u Administrateur -p Defte@WF
MSSQL       192.168.56.32   1433   SERVER           [*] Windows Server 2022 Build 20348 (name:SERVER) (dom
MSSQL       192.168.56.32   1433   SERVER           [+] whiteflag.local\Administrateur:Defte@WF (Pwn3d!)
  2:09 ]  [ ach@blackpearl:~ ]
$
```

Let’s enable TLS over MSSQL



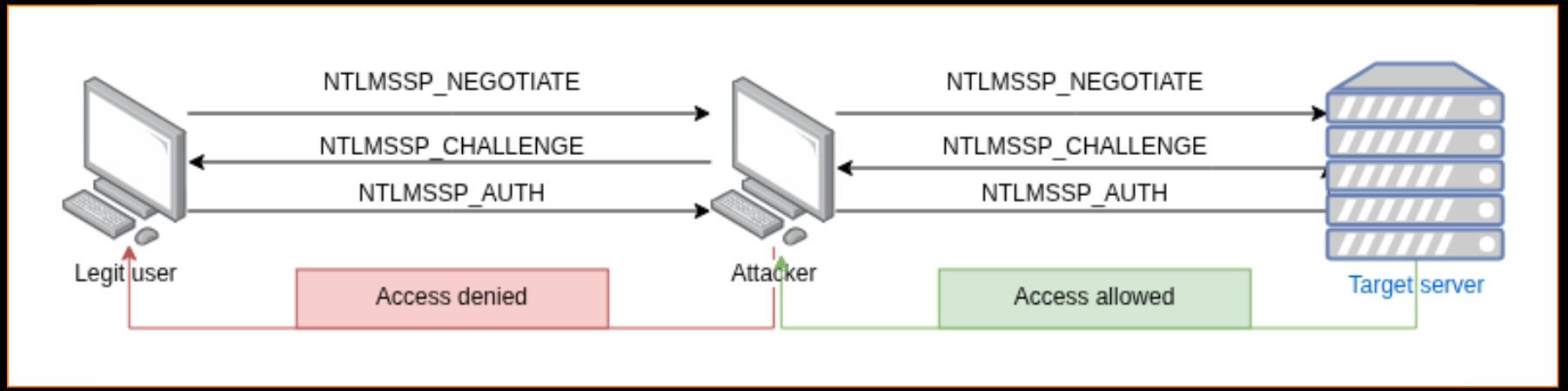
And test NXC again

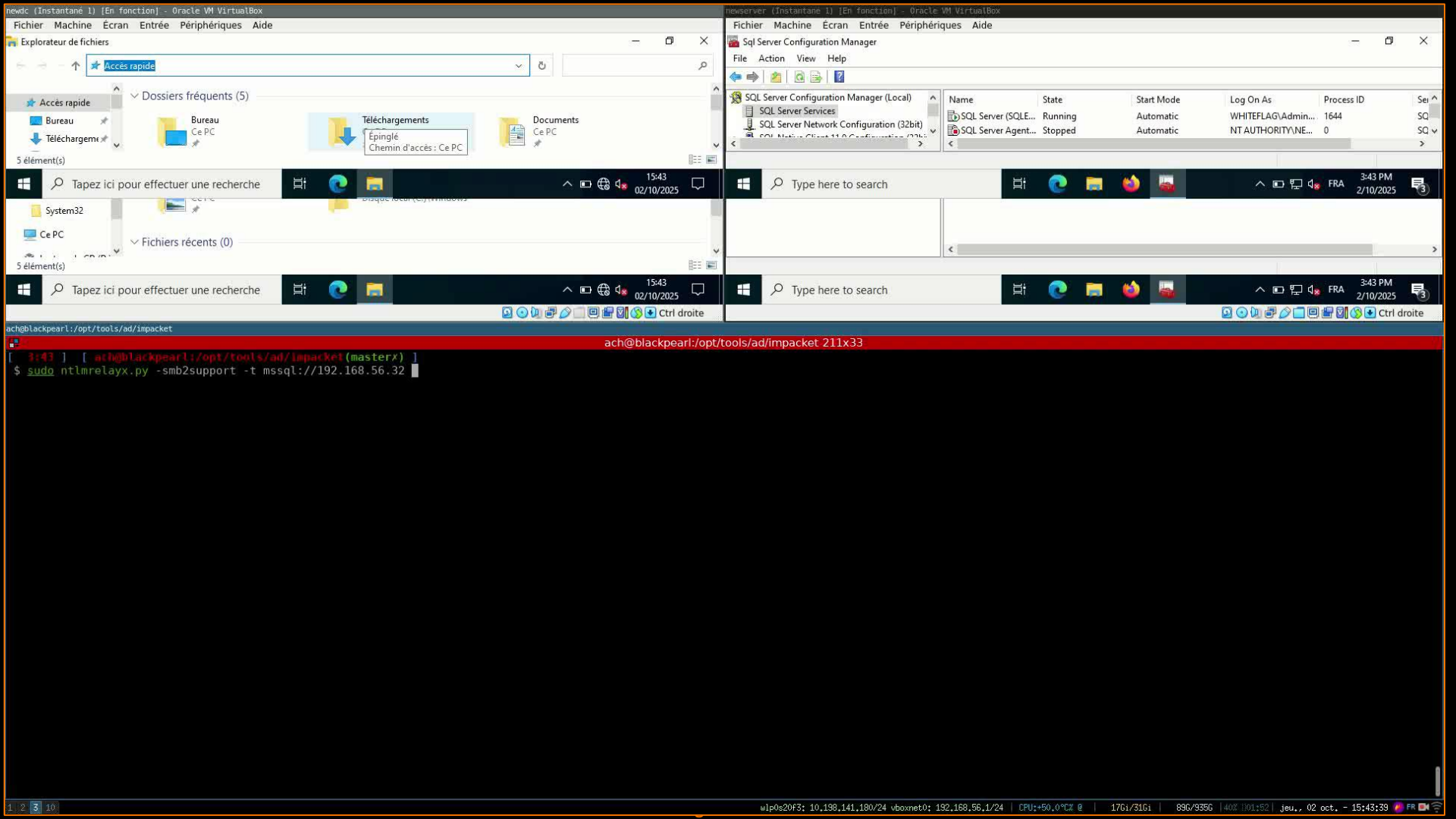


3 / Back to defaults

Default MSSQL configuration is vulnerable to NTLM relay

- NTLMSSP_NEGOTIATE: Hey man, I'm Deft, can I authenticate to you ?
- NTLMSSP_CHALLENGE: Sure, can you encrypt that <string> with your password pls ?
- NTLMSSP_AUTH: Here is the <encrypted_string>




newdc (Instantané 1) [En fonction] - Oracle VM VirtualBox
Fichier Machine Écran Entrée Périphériques Aide
Explorateur de fichiers
Accès rapide
Dossiers fréquents (5)
Bureau
Ce PC
Téléchargements
Épinglé
Chemin d'accès : Ce PC
Documents
Ce PC
5 élément(s)
Tapez ici pour effectuer une recherche
15:43
02/10/2025
System32
Ce PC
Fichiers récents (0)
5 élément(s)
Tapez ici pour effectuer une recherche
15:43
02/10/2025
Ctrl droite
newserver (Instantané 1) [En fonction] - Oracle VM VirtualBox
Fichier Machine Écran Entrée Périphériques Aide
Sql Server Configuration Manager
File Action View Help
SQL Server Configuration Manager (Local)
SQL Server Services
SQL Server Network Configuration (32bit)
Name
State
Start Mode
Log On As
Process ID
SQL Server (SQLE...
Running
Automatic
WHITEFLAG\Admin...
1644
SQL Server Agent...
Stopped
Automatic
NT AUTHORITY\NE...
0
Type here to search
FRA
3:43 PM
2/10/2025
Type here to search
FRA
3:43 PM
2/10/2025
Ctrl droite
ach@blackpearl:/opt/tools/ad/impacket
ach@blackpearl:/opt/tools/ad/impacket 211x33
[3:43] [ach@blackpearl:/opt/tools/ad/impacket(masterx)]
$ sudo ntlmrelayx.py -smb2support -t mssql://192.168.56.32
1 2 3 10
wlp0s20f3: 10.198.141.180/24 vboxnet0: 192.168.56.1/24
CPU:+50.0°C%
17Gi/31Gi
89G/935G
jeu., 02 oct. - 15:43:39

The only way to prevent the relay is to enable Channel Binding:

```
[  4:39 ]  [ ach@blackpearl:~ ]
 $ sudo ntlmrelayx.py -t mssql://192.168.56.32  -smb2support
Impacket v0.13.0.dev0+20250528.4535.5b338613 - Copyright Fortra, LLC and its affiliated companies

[*] Protocol Client SMB loaded..
[*] Protocol Client WINRMS loaded..
[*] Protocol Client SMTP loaded..
[*] Protocol Client RPC loaded..
[*] Protocol Client MSSQL loaded..
[*] Protocol Client IMAP loaded..
[*] Protocol Client IMAPS loaded..
[*] Protocol Client DCSYNC loaded..
[*] Protocol Client LDAP loaded..
[*] Protocol Client LDAPS loaded..
[*] Protocol Client HTTPS loaded..
[*] Protocol Client HTTP 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
[*] Multirelay disabled

[*] Servers started, waiting for connections
[*] SMBD-Thread-5 (process_request_thread): Received connection from 192.168.56.31, attacking target mssql://192.168.56.32
[-] ERROR(server\SQLEXPRESS): Line 1: Échec de la connexion. La connexion provient d'un domaine non approuvé et ne peut pas
[-] Authenticating against mssql://192.168.56.32 as WHITEFLAG/ADMINISTRATEUR FAILED
[*] All targets processed!
```

Looking at the logs

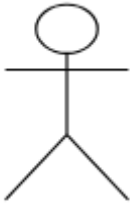
```
The Service Broker endpoint is in disabled or stopped state.
The Database Mirroring endpoint is in disabled or stopped state.
Service Broker manager has started.
Recovery is complete. This is an informational message only. No user action is required.
External governance manager initialized
Attribute synchronization manager initialized
Erreur : 17806, Gravité : 20, État : 46.
SSPI handshake failed with error code 0x80090346, state 46 while establishing a connection with integrated security;
the connection has been closed.
Reason: Les liaisons de canaux de ce client sont manquantes ou ne correspondent pas au canal TLS (Transport Layer Security) établi.
Il est possible que le service fasse l'objet d'une attaque, ou que le fournisseur de données ou système d'exploitation client nécessite
une mise à niveau pour prendre en charge la protection étendue.
Fermeture de la connexion. Les liaisons de chaînes SSPI fournies au client étaient incorrectes.   [CLIENT : 192.168.56.1]
Erreur : 18452, Gravité : 14, État : 1.
Login failed. The login is from an untrusted domain and cannot be used with Integrated authentication. [CLIENT : 192.168.56.1]
Erreur : 17806, Gravité : 20, État : 46.
```

The binding token for that client is missing or does not match the established TLS link.
It is possible that the MSSQL service is under attack or that the client does not support extended protection.
Closing the connection. The SSPI binding token sent by the client does not match.

4 / About TDS and STARTTLS

About TDS workflow

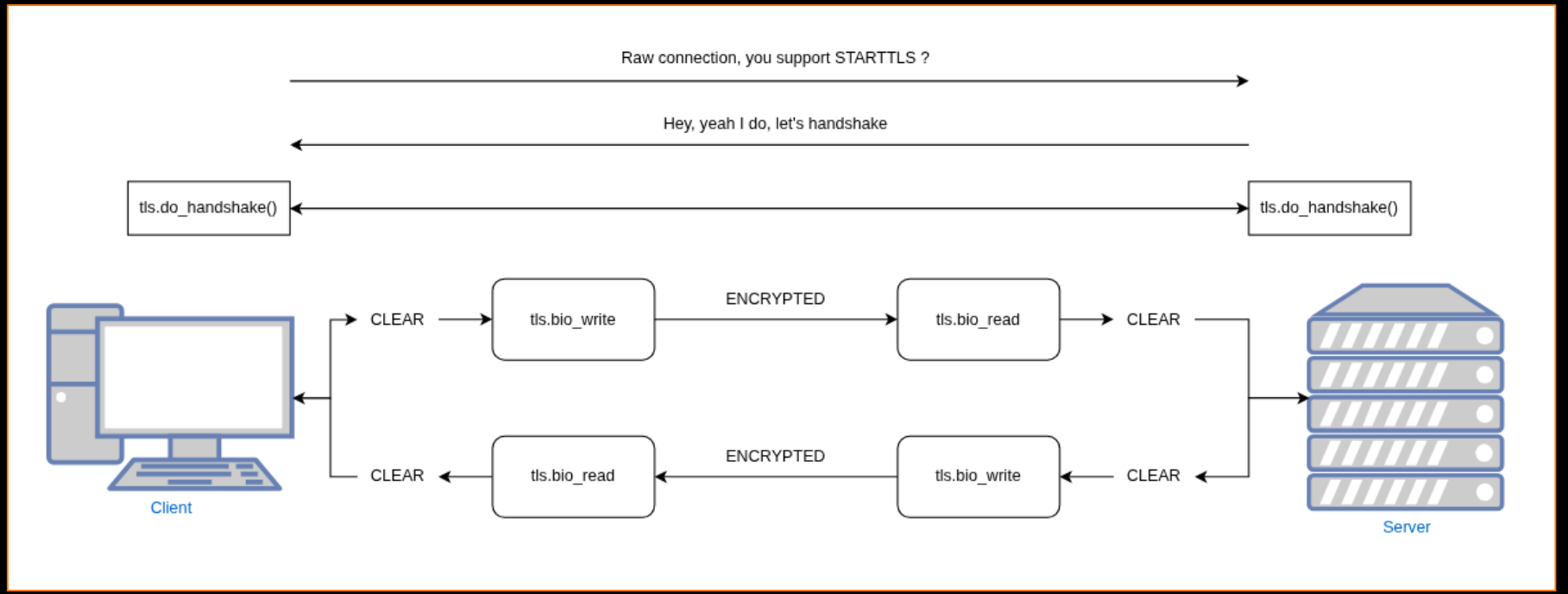
```
class TDS_LOGIN(Structure):
    structure = (
        ('Length','<L=0'),
        ('TDSVersion','>L=0x71'),
        ('PacketSize','<L=32764'),
        ('ClientProgVer','>L=7'),
        ('ClientPID','<L=0'),
        ('ConnectionID','<L=0'),
        ('OptionFlags1','<B=0xe0'),
        ('OptionFlags2','<B'),
        ('TypeFlags','<B=0'),
        ('OptionFlags3','<B=0'),
        ('ClientTimeZone','<L=0'),
        ('ClientLCID','<L=0'),
        ('HostNameOffset','<H'),
        ('HostNameLength','<H=len(self["HostName"])//2'),
        ('UserNameOffset','<H=0'),
        ('UserNameLength','<H=len(self["UserName"])//2'),
        ('PasswordOffset','<H=0'),
        ('PasswordLength','<H=len(self["Password"])//2'),
        ('AppNameOffset','<H'),
        ('AppNameLength','<H=len(self["AppName"])//2'),
        ('ServerNameOffset','<H'),
        ('ServerNameLength','<H=len(self["ServerName"])//2'),
        ('UnusedOffset','<H=0'),
        ('UnusedLength','<H=0'),
        ('CltIntNameOffset','<H'),
        ('CltIntNameLength','<H=len(self["CltIntName"])//2'),
        ('LanguageOffset','<H=0'),
        ('LanguageLength','<H=0'),
        ('DatabaseOffset','<H=0'),
        ('DatabaseLength','<H=len(self["Database"])//2'),
        ('ClientID','6s=b"\x01\x02\x03\x04\x05\x06"'),
        ('SSPIOffset','<H'),
        ('SSPILength','<H=len(self["SSPI"])'),
        ('AtchDBFileOffset','<H'),
        ('AtchDBFileLength','<H=len(self["AtchDBFile"])//2'),
        ('HostName',':'),
        ('UserName',':'),
        ('Password',':'),
        ('AppName',':'),
        ('ServerName',':'),
        ('CltIntName',':'),
        ('Database',':'),
        ('SSPI',':'),
        ('AtchDBFile',':'),
    )
```



Authentication workflow and STARTTLS

```
 7  def login(self, database, username, password='', domain='', hashes = None, useWindowsAuth = False):
 8      if hashes is not None:
 9          lmhash, nthash = hashes.split(':')
10          lmhash = binascii.a2b_hex(lmhash)
11          nthash = binascii.a2b_hex(nthash)
12      else:
13          lmhash = ''
14          nthash = ''
15
16      resp = self.preLogin()
17      # Test this!
18      if resp['Encryption'] == TDS_ENCRYPT_REQ or resp['Encryption'] == TDS_ENCRYPT_OFF:
19          LOG.info("Encryption required, switching to TLS")
20
21          # Switching to TLS now
22          ctx = SSL.Context(SSL.TLS_METHOD)
23          ctx.set_cipher_list('ALL:@SECLEVEL=0'.encode('utf-8'))
24          tls = SSL.Connection(ctx,None)
25          tls.set_connect_state()
26          while True:
27              try:
28                  tls.do_handshake()
29              except SSL.WantReadError:
30                  data = tls.bio_read(4096)
31                  self.sendTDS(TDS_PRE_LOGIN, data,0)
32                  tds = self.recvTDS()
33                  tls.bio_write(tds['Data'])
34              else:
35                  break
```

STARTTLS schema

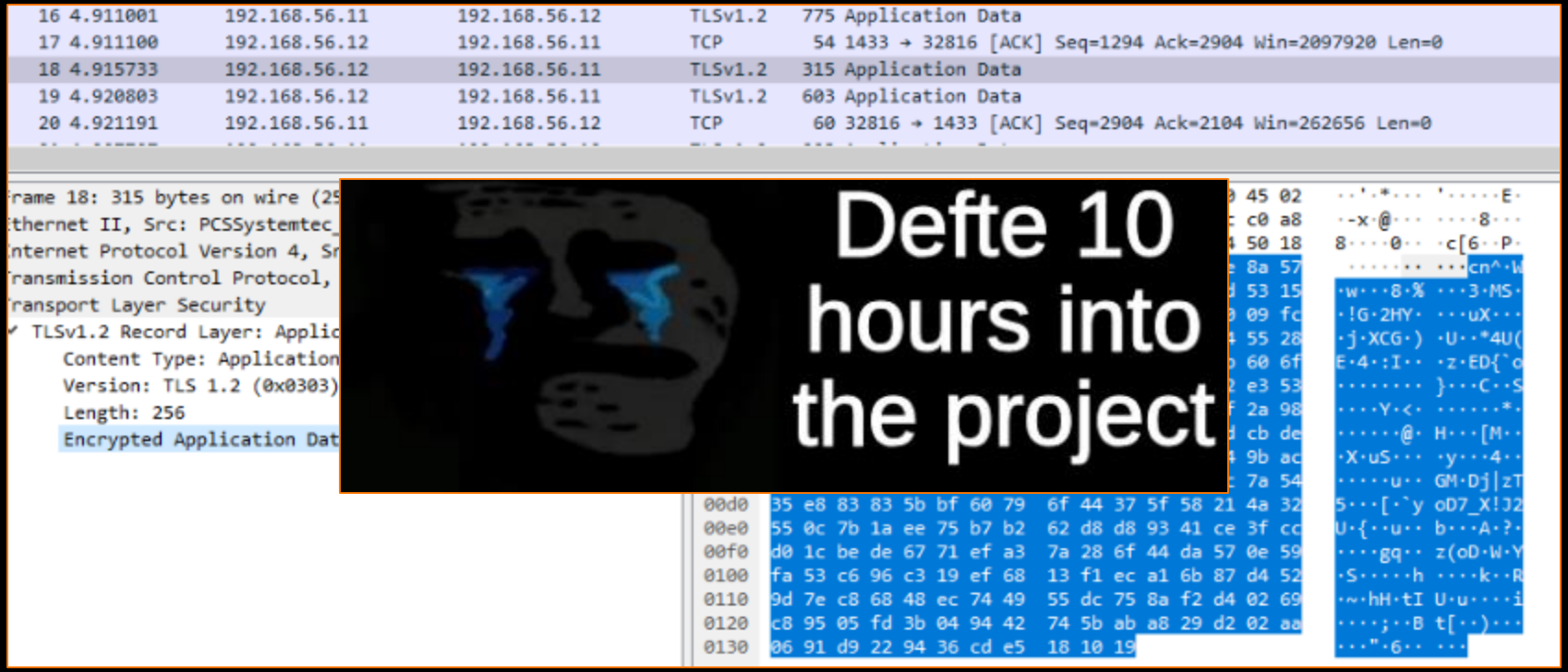


LDAP3 channel binding code

```python
if self.channel_binding == TLS_CHANNEL_BINDING:
    # To perform channel binding during NTLM authentication, we need to add a new AV_PAIR (MS-NLMP 2.2.2.1)
    # within the AUTHENTICATE_MESSAGE (MS-NLMP 2.2.1.3). This new AV_PAIR has AvId 0x000A (MsvAvChannelBindings).
    # The Value field contains an MD5 hash of a gss_channel_bindings_struct.
    # The logic here is heavly inspired by "msldap", "minikerberos" and "asysocks" projects by @skelsec.
    from hashlib import sha256, md5
    ntlm_client.tls_channel_binding = True
    peer_certificate_sha256 = sha256(self.server.tls.peer_certificate).digest()
    # https://datatracker.ietf.org/doc/html/rfc2744#section-3.11
    channel_binding_struct = bytes()
    initiator_address = b'\x00'*8
    acceptor_address = b'\x00'*8
    # https://datatracker.ietf.org/doc/html/rfc5929#section-4
    application_data_raw = b'tls-server-end-point:' + peer_certificate_sha256
    len_application_data = len(application_data_raw).to_bytes(4, byteorder='little', signed = False)
    application_data = len_application_data
    application_data += application_data_raw
    channel_binding_struct += initiator_address
    channel_binding_struct += acceptor_address
    channel_binding_struct += application_data
    # https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-nlmp/83f5e789-660d-4781-8491-5f8c6641f75e
    # "The Value field contains an MD5 hash of a gss_channel_bindings_struct"
    ntlm_client.client_av_channel_bindings = md5(channel_binding_struct).digest()
```

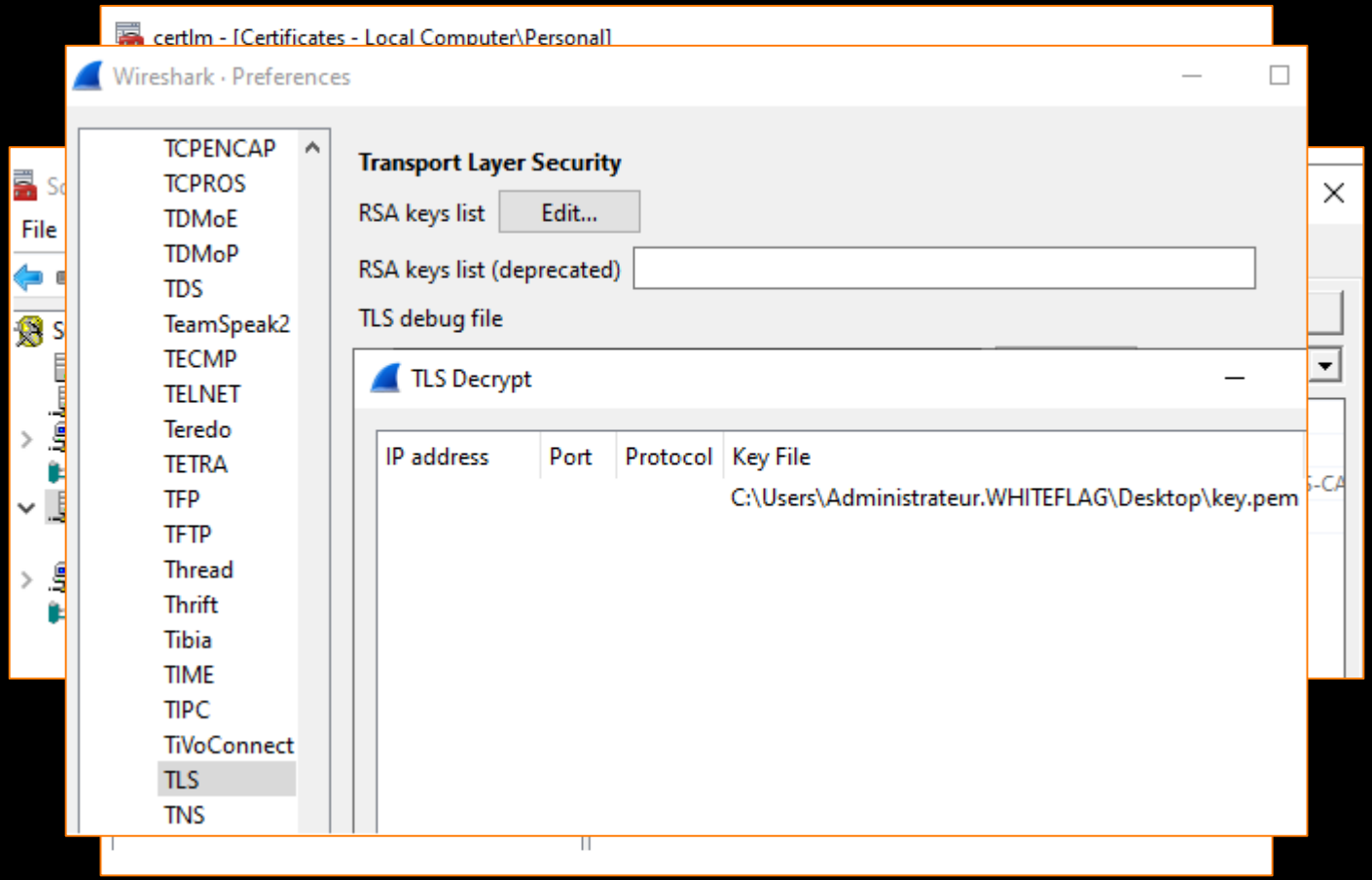
Copy paste and…

Looking at the authentication PCAP…

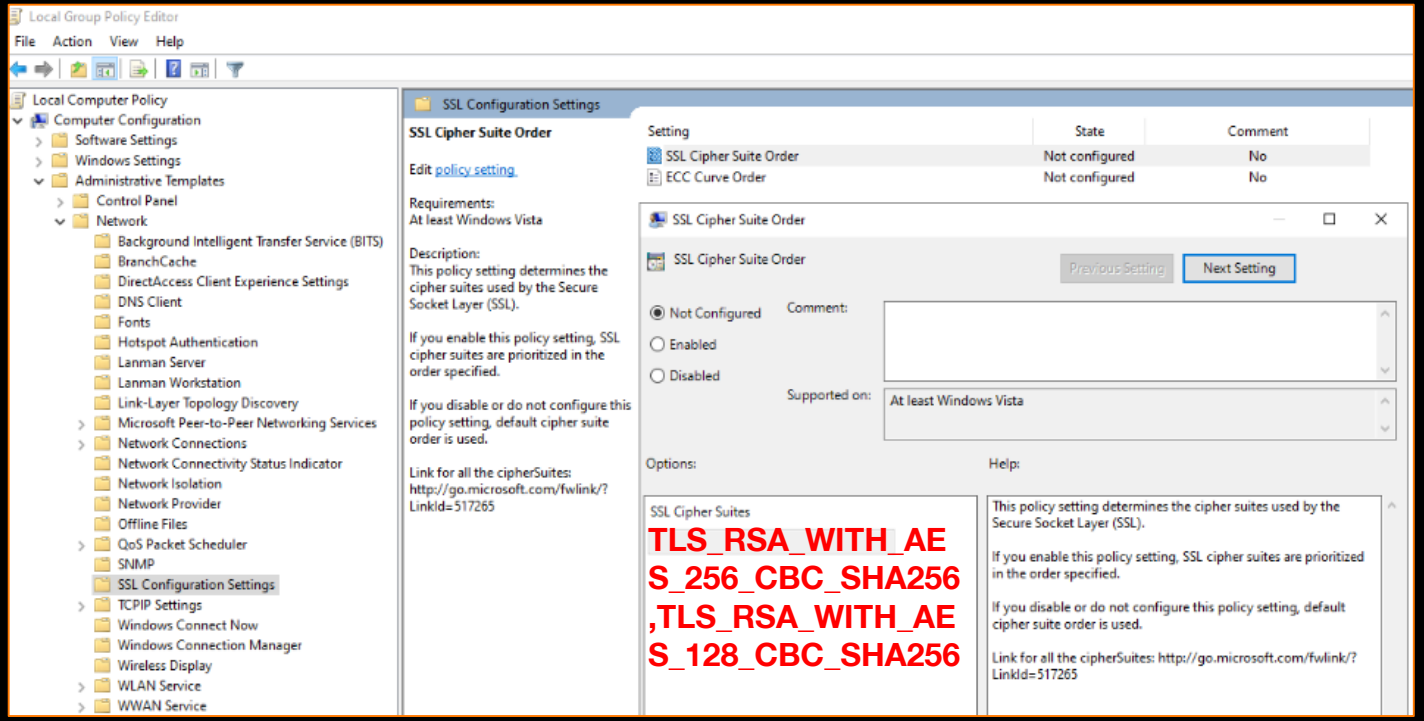


5 / Downgrading TLS

First we need a certificate and its private key

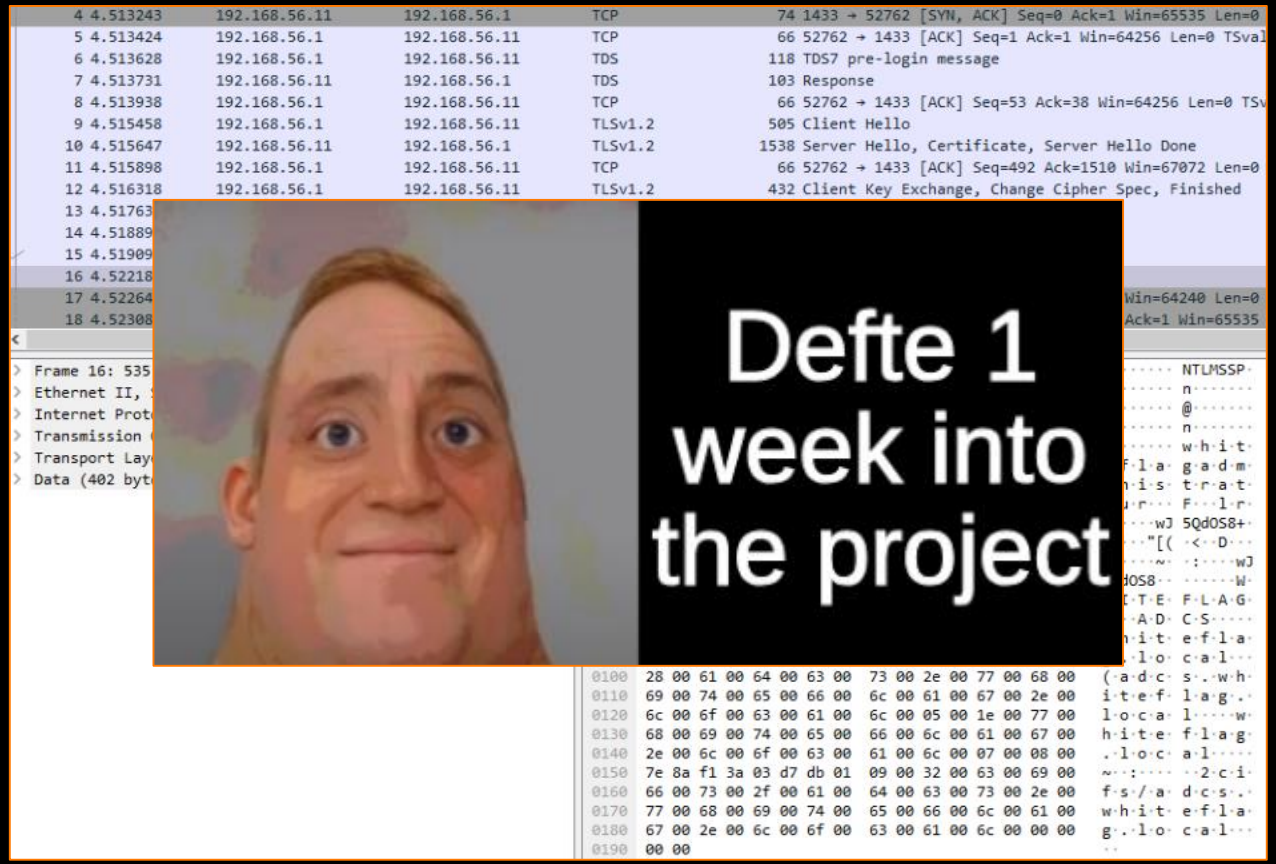


Then we need to downgrade TLS standards



RSA is mandatory because anything else uses Elliptic-Curves which is a pain to intercept1

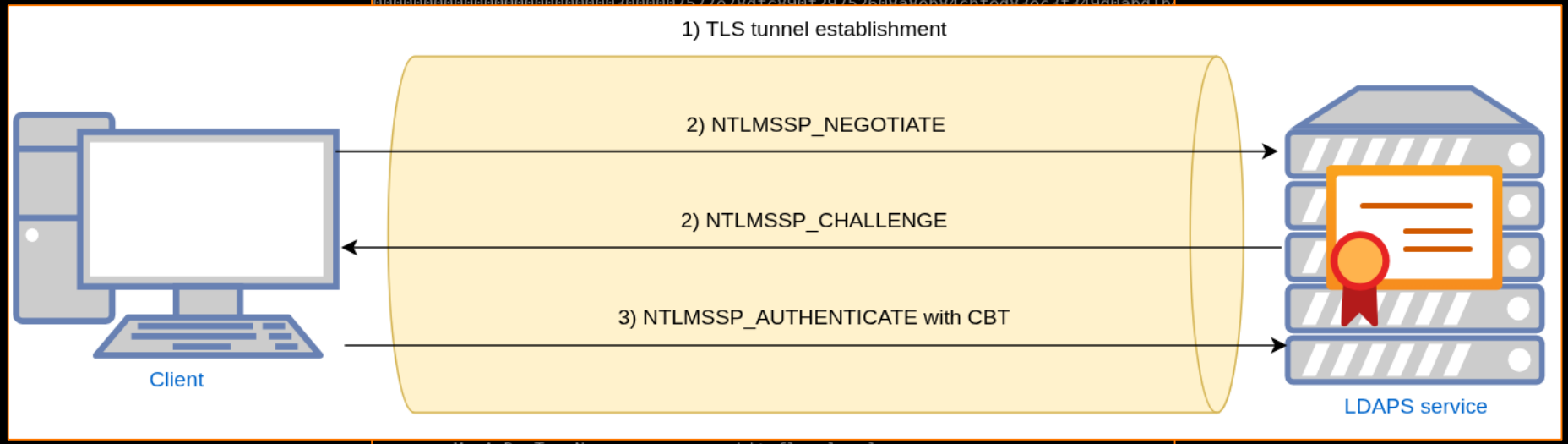
And finally….



6 / Implementing CBT

Dissecting NTLM authentication and why NTLMrelay doesn't work anymore

```
MIC:        : 415d34e4b0d9a57eb3a02b19c4bc47e4
Lanman      : 00000000000000000000000000000000000000000000000000
Ntlm        : 0ce8dd0704a0f9d163b8cfb01de66ba701010000000000030e8b367aceadb010
7004800490054004500460004c004100470001000800410044004300530004001e00770068006900
6f00630061006c00030028006100640063007300 2e00770068006900740065006600 6c006100670
07700680069007400650066006c00610067002e006c006f00630061006c000700080030e8b367ac
```




```
MsvAvDnsTreeName        : whiteflag.local
MsvAvTimestamp          : 2025-07-01 17:20:13.267972
MsvAvFlags              : 0x00000002 (TARGET_IS_UNTRUSTED)
MsvAvSingleHost         :
        Size            : 48
        Z4              : 00000000
        Custom Data     : 0000000000300000
        Machine ID      : 7577e78dfc890f29752608a8eb84cbfe
MsvAvChannelBindings    : 0c237bcdc07178a5c801f2446e7ce6b9
MsvAvTargetName         : MSSQLSvc/adcs.whiteflag.local:1433
MsvAvEOL
```

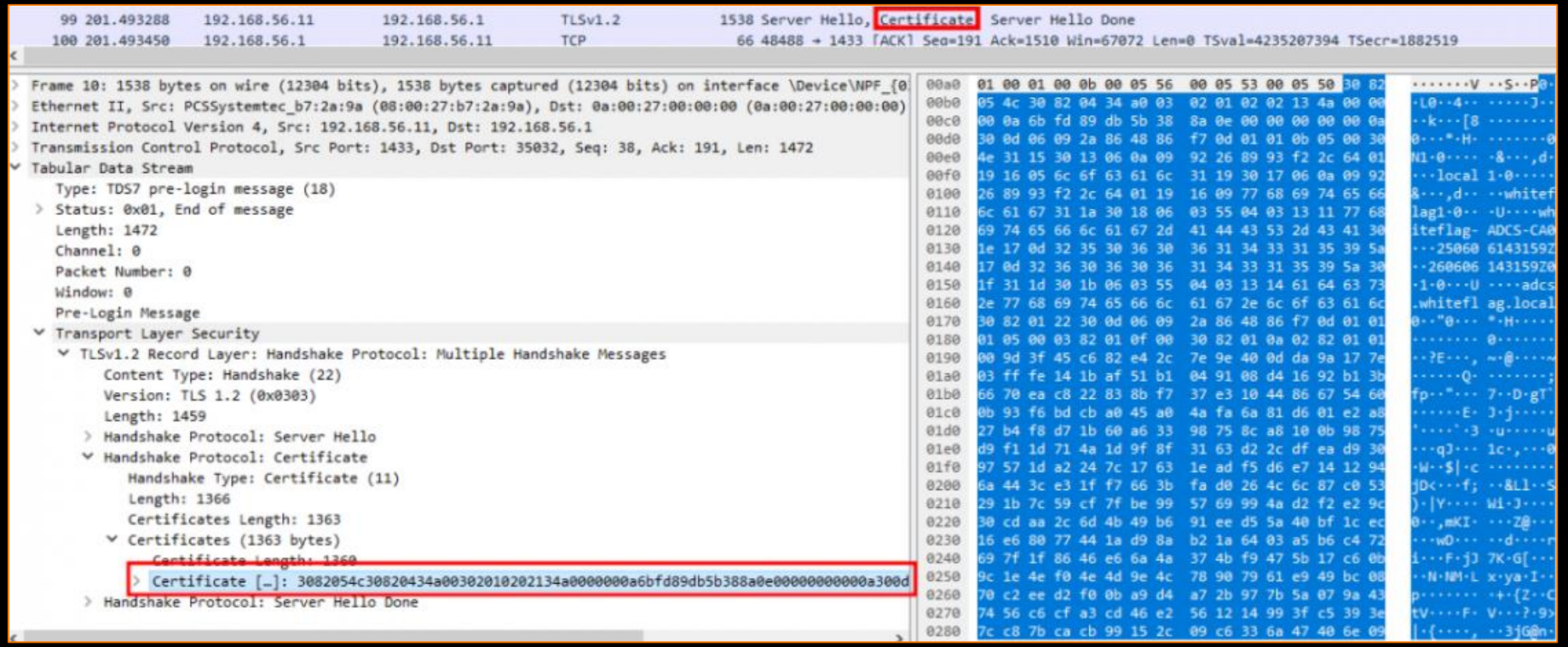
The fuck ??

```
rs from NTLMv2 response:
  MsvAvNbDomainName         : WHITEFLAG
  MsvAvNbComputerName       : ADCS
  MsvAvDnsDomainName        : whiteflag.local
  MsvAvDnsComputerName      : adcs.whiteflag.local
  MsvAvDnsTreeName          : whiteflag.local
  MsvAvTimestamp            : 2025-06-25 15:25:03.887978
  MsvAvFlags                : 0x00000002 (TARGET_IS_UNTRUSTED)
  MsvAvSingleHost           :
          Size              : 48
          Z4                : 00000000
          Custom Data       : 0000000000300000
          Machine ID        : fd97e810783133b6a0a9e9042bf74eed
  MsvAvChannelBindings      : 0d0aeb3d968741e55aa2529829bc5a10
  MsvAvTargetName           : MSSQLSvc/adcs.whiteflag.local:1433
  MsvAvEOL
```

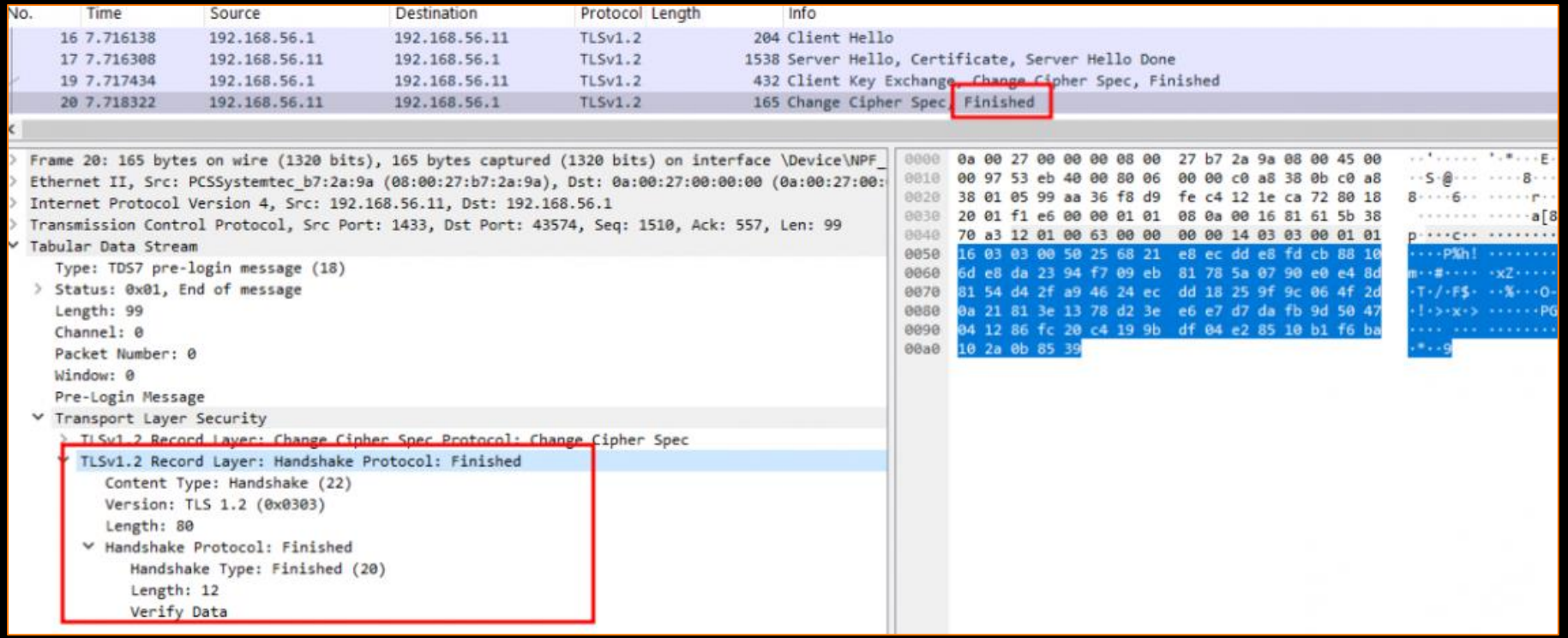
```
rs from NTLMv2 response:
  MsvAvNbDomainName         : WHITEFLAG
  MsvAvNbComputerName       : ADCS
  MsvAvDnsDomainName        : whiteflag.local
  MsvAvDnsComputerName      : adcs.whiteflag.local
  MsvAvDnsTreeName          : whiteflag.local
  MsvAvTimestamp            : 2025-06-25 15:26:25.035212
  MsvAvFlags                : 0x00000002 (TARGET_IS_UNTRUSTED)
  MsvAvSingleHost           :
          Size              : 48
          Z4                : 00000000
          Custom Data       : 0000000000300000
          Machine ID        : fd97e810783133b6a0a9e9042bf74eed
  MsvAvChannelBindings      : 61195c99f19cfae2647e719a15d40cf3
  MsvAvTargetName           : MSSQLSvc/adcs.whiteflag.local:1433
  MsvAvEOL
```

Did you know there are multiple ways you can compute a Channel Binding Token ?

TLS Server End Point (the one used by LDAPS)



TLS Unique (spoiler: the one used by MSSQL)



TLS Exporter

tls exporter computation

Demander | Tous | Images | Actualités | Vidéos | Recherche personnali

The computation of exported keying material in TLS is defined by RFC 5705 for TLS 1.2 and earlier, and by RFC 8446 for TLS 1.3. For TLS 1.2, the exporter computes a pseudorandom bit string using the TLS Pseudorandom Function (PRF) based on the master secret, a disambiguating label, and optionally a context value. The formula is:

$$\text{PRF}(\text{SecurityParameters.master\_secret}, \text{label}, \text{SecurityParameters.client\_random} + \text{SecurityParameters.server\_random} + \text{context\_value\_length} + \text{context\_value})[\text{length}]$$

If no context is provided, the computation simplifies to:

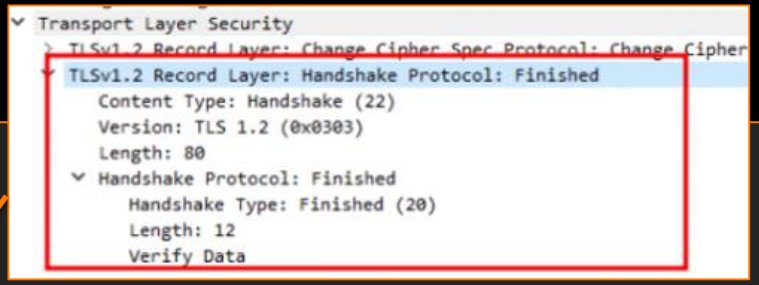
$$\text{PRF}(\text{SecurityParameters.master\_secret}, \text{label}, \text{SecurityParameters.client\_random} + \text{SecurityParameters.server\_random})[\text{length}]$$

For TLS 1.3, the PRF is replaced with HKDF, and the exporter computation is defined as:

$$\text{TLS-Exporter}(\text{label}, \text{context\_value}, \text{key\_length}) = \text{HKDF-Expand-Label}(\text{Derive-Secret}(\text{Secret}, \text{label}, ""), \text{"exporter"}, \text{Hash}(\text{context\_value}), \text{key\_length})$$

And MSSQL does not support it yet so….

Final computing

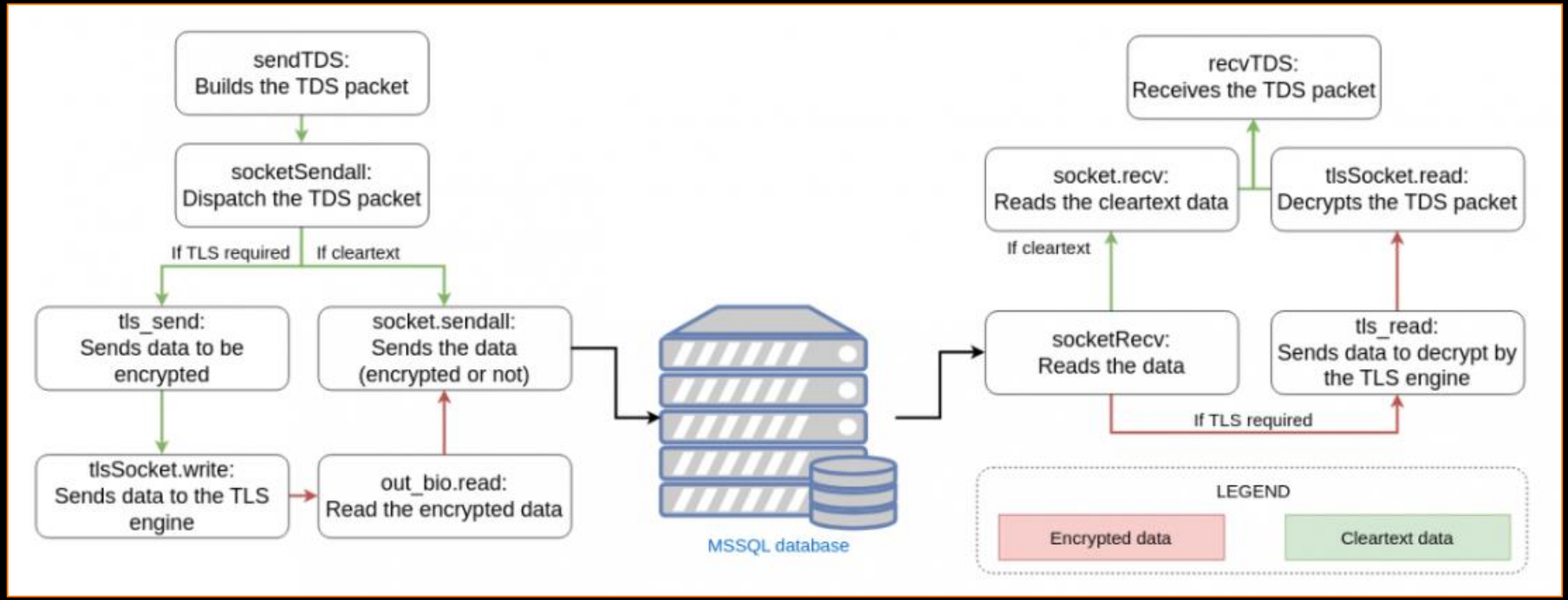



```python
def generate_cbt_from_tls_unique(self) -> bytes:
    channel_binding_struct = b""
    initiator_address = b"\x00" * 8
    acceptor_address = b"\x00" * 8
    application_data_raw = b"tls-unique:" + self.tls_unique # This was tls-server-end-point before
    len_application_data = len(application_data_raw).to_bytes(4, byteorder="little", signed=False)
    application_data = len_application_data
    application_data += application_data_raw
    channel_binding_struct += initiator_address
    channel_binding_struct += acceptor_address
    channel_binding_struct += application_data
    cbt_token = md5(channel_binding_struct).digest()
    LOG.debug(f"Computed tls-unique CBT token: {cbt_token.hex()}")
    return cbt_token
```

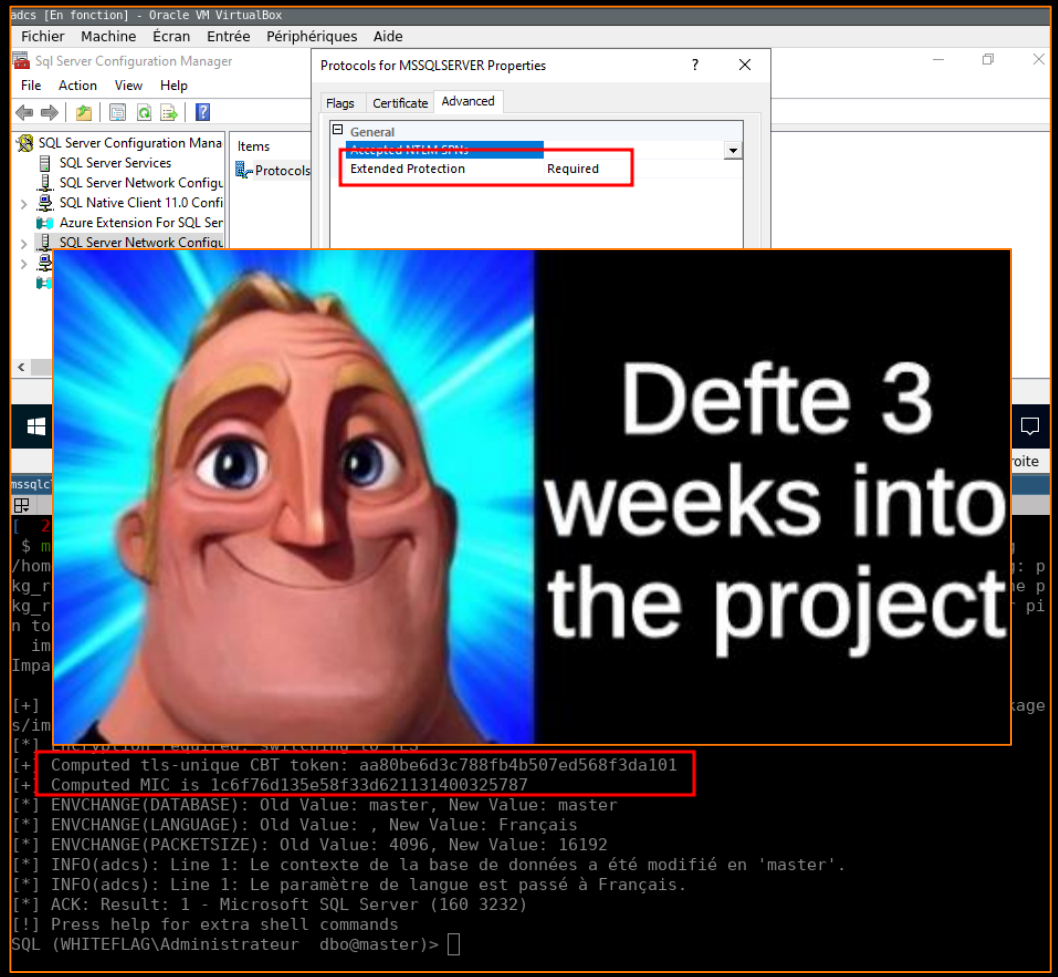
b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x17\x00\x00\x00
tls-unique:;\xbf\x87_\x85(\x94\xd562}['

BUT…

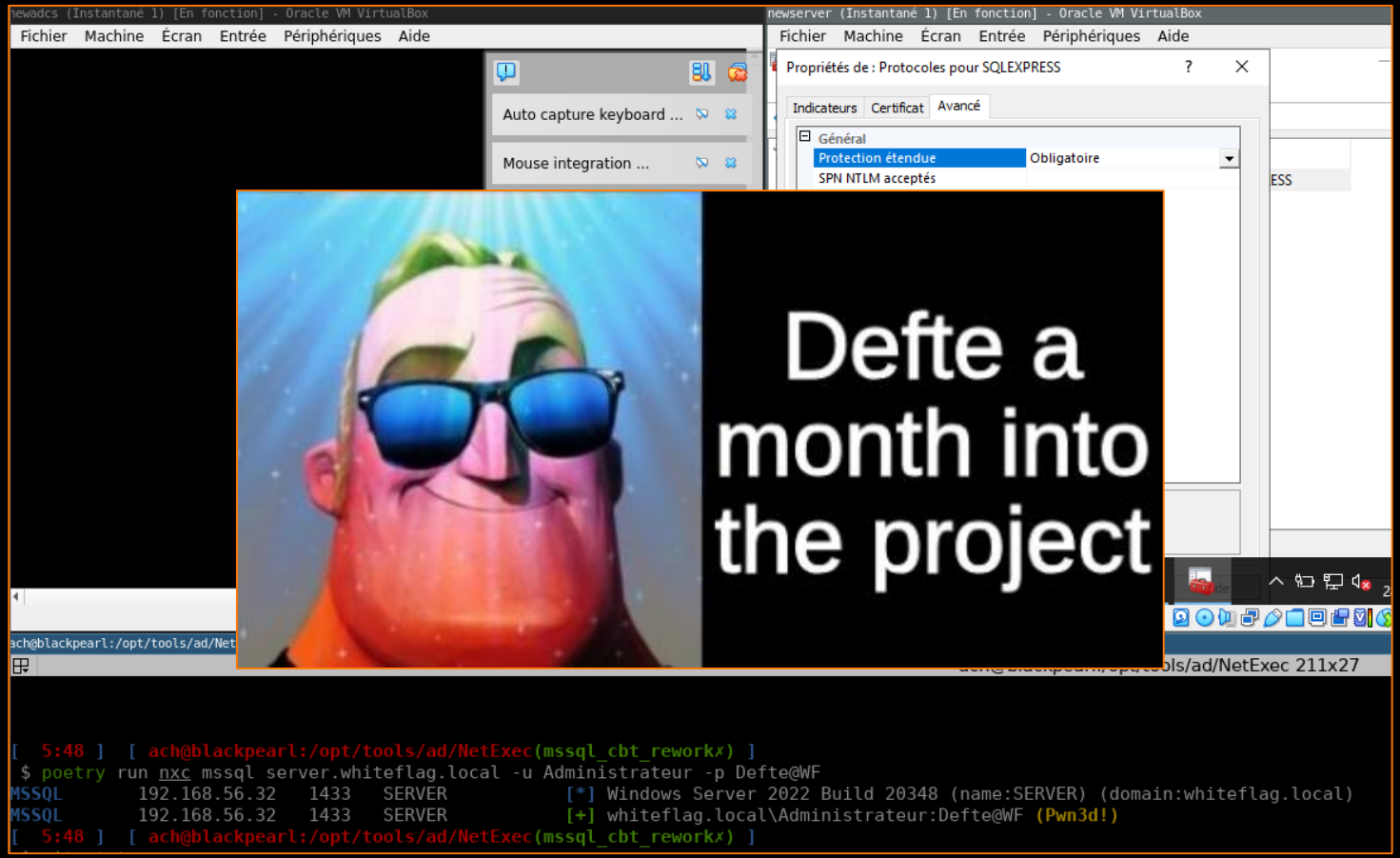
MSSQL supports both encryption via STARTTLS and plain text communications, so:



And...



Finally…



7 / Sum up & conclusion

Sum up

- MSSQL databases are vulnerable to NTLMrelay attacks by default
- To prevent that, M$ developed the Channel Binding token that is used to bind an authentication to a TLS tunnel
- To work, a TLS tunnel is mandatory (e.g CBT doesn't work on unencrypted channels)
- Channel Binding Tokens are buit using three methods:
  - TLS Server End Point (used by LDAPS) : computed via the sha256 hash of the certificate
  - TLS Unique (used by MSSQL) : computed via the sha256 of the TLS Finished message
  - TLS Exporter (used by ???) : computed via ……. ?
- Computing the CBT value and adding it to the NTLMSSP_AUTHENTICATE message grants us access to the most hardened MSSQL databases

Questions ?

Mail: aurelien.chalot@orangecyberdefense.com

Twitter: https://x.com/Defte_

Discord: deft_.

Related blogpost : https://blog.whiteflag.io/blog/a-journey-about-mssql/

Slides: https://github.com/Orange-Cyberdefense/conferences