Fixing NTLM decryption in Wireshark

Clément Notin

Staff Research Engineer

Pass the SALT 2023
Lille, France

NTLM decryption?

- NTLM is firstly an authentication protocol
  - Old one, obsolete, many security flaws ➡️ don't use it! (but you know, people still do...)

- But also:
  *"The process of authentication establishes a shared context between the two involved parties; this includes a **shared session key**, used for subsequent **signing** and **sealing** operations."*

- "Signing" is... signing 😉
- "Sealing" is a fancy word for "encryption" 🤷‍♀️

Source: https://davenport.sourceforge.net/ntlm.html

NTLM decryption?

👀 If you know the user's password...

➡️ you can obtain the shared session key...

💪 you can decrypt the traffic!

tenable

In Wireshark 🎉

create two tasks then enum RPC_C_AUTHN_LEVEL_PKT_PRIVACY.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Decrypted stub data	Sequence	Data
		49154 → 41722 [ACK] Seq=285 Ack=497 Win=64240 Len=0
28efdfe36ad36… 67150000130000000000000130000005c006600690072007300740074006100730060b00…	00000000	SchRpcRegisterTask request
		49154 → 41722 [ACK] Seq=285 Ack=1957 Win=64240 Len=0
		49154 → 41722 [ACK] Seq=285 Ack=3397 Win=64240 Len=0
376f4556d88a6… 00000200130000000000000130000005c006600690072007300740074006100730060b00…	00000000	SchRpcRegisterTask response
		41722 → 49154 [ACK] Seq=3397 Ack=397 Win=63956 Len=0
e33d4be166f4c… 23a660fe80c5b38f0ac74de1d56d065b98b791a169f29c05a3f17add35c2830055674b40…	01000000	SchRpcRegisterTask request

First call

```
> Frame 25: 2954 bytes on wire (23632 bits), 2954 bytes captured (23632 bits) on interface eth0, id 0
> Ethernet II, Src: VMware_08:66:01 (00:0c:29:08:66:01), Dst: VMware_f3:9b:97 (00:50:56:f3:9b:97)
> Internet Protocol Version 4, Src: 192.168.80.128 (192.168.80.128), Dst: 192.168.42.2 (192.168.42.2)
> Transmission Control Protocol, Src Port: 41722 (41722), Dst Port: 49154 (49154), Seq: 497, Ack: 285,
v Distributed Computing Environment / Remote Procedure Call (DCE/RPC) Request, Fragment: Single, FragLe
    Version: 5
    Version (minor): 0
    Packet type: Request (0)
  > Packet Flags: 0x03
  > Data Representation: 10000000 (Order: Little-endian, Char: ASCII, Float: IEEE)
    Frag Length: 2900
    Auth Length: 16
    Call ID: 2
    Alloc hint: 2852
    Context ID: 0
    Opnum: 1
    [Response in frame: 28]
    Encrypted stub data: db2614835d7c0be0e1f1462d61eef3a8d463f4c406758d15588ecbdcfcc28efdfe36ad36…
  v Auth Info: NTLMSSP, Packet privacy, AuthContextId(79231)
      Auth type: NTLMSSP (10)
      Auth level: Packet privacy (6)
      Auth pad len: 0
      Auth Rsrvd: 0
      Auth Context ID: 79231
    v NTLMSSP Verifier
```

```
0000  67 15 00 00 13 00 00 00  00 00 00 00 13 00 00 00   g····· ····· ····
0010  5c 00 66 00 69 00 72 00  73 00 74 00 74 00 61 00   \·f·i·r· s·t·t·a
0020  73 00 6b 00 6a 00 77 00  6c 00 70 00 6d 00 64 00   s·k·j·w· l·p·m·d
0030  49 00 66 00 00 00 ab ab  65 05 00 00 00 00 00 00   I·f····· e·····
0040  65 05 00 00 3c 00 3f 00  78 00 6d 00 6c 00 20 00   e···<·?· x·m·l· ·
0050  76 00 65 00 72 00 73 00  69 00 6f 00 6e 00 3d 00   v·e·r·s· i·o·n·=·
0060  22 00 31 00 2e 00 30 00  22 00 20 00 65 00 6e 00   "·1·.·0· "· ·e·n·
0070  63 00 6f 00 64 00 69 00  6e 00 67 00 3d 00 22 00   c·o·d·i· n·g·=·"·
0080  55 00 54 00 46 00 2d 00  31 00 36 00 22 00 3f 00   U·T·F·-· 1·6·"·?·
0090  3e 00 0a 00 3c 00 54 00  61 00 73 00 6b 00 20 00   >···<·T· a·s·k· ·
00a0  76 00 65 00 72 00 73 00  69 00 6f 00 6e 00 3d 00   v·e·r·s· i·o·n·=·
00b0  22 00 31 00 2e 00 32 00  22 00 20 00 78 00 6d 00   "·1·.·2· "· ·x·m·
00c0  6c 00 6e 00 73 00 3d 00  22 00 68 00 74 00 74 00   l·n·s·=· "·h·t·t·
00d0  70 00 3a 00 2f 00 2f 00  73 00 63 00 68 00 65 00   p·:·/·/· s·c·h·e·
00e0  6d 00 61 00 73 00 2e 00  6d 00 69 00 63 00 72 00   m·a·s·.· m·i·c·r·
00f0  6f 00 73 00 6f 00 66 00  74 00 2e 00 63 00 6f 00   o·s·o·f· t·.·c·o·
0100  6d 00 2f 00 77 00 69 00  6e 00 64 00 6f 00 77 00   m·/·w·i· n·d·o·w·
0110  73 00 2f 00 32 00 30 00  30 00 34 00 2f 00 30 00   s·/·2·0· 0·4·/·0·
0120  32 00 2f 00 6d 00 69 00  74 00 2f 00 74 00 61 00   2·/·m·i· t·/·t·a·
0130  73 00 6b 00 22 00 3e 00  0a 00 20 00 20 00 3c 00   s·k·"·>· ·· · ·<·
0140  54 00 72 00 69 00 67 00  67 00 65 00 72 00 73 00   T·r·i·g· g·e·r·s·
0150  3e 00 0a 00 20 00 20 00  20 00 20 00 3c 00 43 00   >··· · · · ·<·C·
0160  61 00 6c 00 65 00 6e 00  64 00 61 00 72 00 54 00   a·l·e·n· d·a·r·T·
0170  72 00 69 00 67 00 67 00  65 00 72 00 3e 00 0a 00   r·i·g·g· e·r·>···
0180  20 00 20 00 20 00 20 00  20 00 20 00 3c 00 53 00    · · · ·  · ·<·S·
0190  74 00 61 00 72 00 74 00  42 00 6f 00 75 00 6e 00   t·a·r·t· B·o·u·n·
01a0  64 00 61 00 72 00 79 00  3e 00 32 00 30 00 31 00   d·a·r·y· >·2·0·1·
01b0  35 00 2d 00 30 00 37 00  2d 00 31 00 35 00 54 00   5·-·0·7· -·1·5·T·
```

Decrypted data is correct (readable)

Frame (2954 bytes) | Decrypted NTLMSSP Verifier (12 bytes) | Decrypted data (2852 bytes) | Decrypted stub data (2852 bytes)

create two tasks then enum RPC_C_AUTHN_LEVEL_PKT_PRIVACY.pcapng | Packets: 41 · Displayed: 26 (63.4%) | Profile: Default

tenable

In Wireshark 😭

create two tasks then enum RPC_C_AUTHN_LEVEL_PKT_PRIVACY.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Decrypted stub data	Sequence	Data
		49154 → 41722 [ACK] Seq=285 Ack=497 Win=64240 Len=0
28efdfe36ad36… 671500001300000000000000130000005c006600690072007300740074006100730060b00…	00000000	SchRpcRegisterTask request
		49154 → 41722 [ACK] Seq=285 Ack=1957 Win=64240 Len=0
		49154 → 41722 [ACK] Seq=285 Ack=3397 Win=64240 Len=0
376f4556d88a6… 000002001300000000000000130000005c006600690072007300740074006100730060b00…	00000000	SchRpcRegisterTask response
		41722 → 49154 [ACK] Seq=3397 Ack=397 Win=63956 Len=0
e33d4be166f4c… 23a660fe80c5b38f0ac74de1d56d065b98b791a169f29c05a3f17add35c2830055674b40…	01000000	SchRpcRegisterTask request

Second call

```
> Frame 30: 2954 bytes on wire (23632 bits), 2954 bytes captured (23632 bits) on interface eth0, id 0
> Ethernet II, Src: VMware_08:66:01 (00:0c:29:08:66:01), Dst: VMware_f3:9b:97 (00:50:56:f3:9b:97)
> Internet Protocol Version 4, Src: 192.168.80.128 (192.168.80.128), Dst: 192.168.42.2 (192.168.42.2)
> Transmission Control Protocol, Src Port: 41722 (41722), Dst Port: 49154 (49154), Seq: 3397, Ack: 397,
v Distributed Computing Environment / Remote Procedure Call (DCE/RPC) Request, Fragment: Single, FragLe
    Version: 5
    Version (minor): 0
    Packet type: Request (0)
  > Packet Flags: 0x03
  > Data Representation: 10000000 (Order: Little-endian, Char: ASCII, Float: IEEE)
    Frag Length: 2900
    Auth Length: 16
    Call ID: 3
    Alloc hint: 2852
    Context ID: 0
    Opnum: 1
    [Response in frame: 33]
    Encrypted stub data: 79f59f919a13498ae337d2705b7e4fd17b8043d1268cd3d484714a0c764e33d4be166f4c…
  v Auth Info: NTLMSSP, Packet privacy, AuthContextId(79231)
      Auth type: NTLMSSP (10)
      Auth level: Packet privacy (6)
      Auth pad len: 0
      Auth Rsrvd: 0
      Auth Context ID: 79231
    v NTLMSSP Verifier
```

```
0000  23 a6 60 fe 80 c5 b3 8f  0a c7 4d e1 d5 6d 06 5b   #·`····· ··M··m·[
0010  98 b7 91 a1 69 f2 9c 05  a3 f1 7a dd 35 c2 83 00   ····i··· ··z·5···
0020  55 67 4b 40 31 21 a6 f2  0d 1d 63 b0 87 76 f6 b8   UgK@1!·· ··c··v··
0030  30 b3 3b d0 1f de e5 10  d7 18 94 6b ae 73 11 87   0·;····· ···k·s··
0040  d7 ee 71 eb b5 df e2 a9  48 c9 f6 f9 47 63 62 d8   ··q····· H···Gcb·
0050  d8 a7 23 26 20 f5 9a e9  4d a8 e3 8f a4 e3 4d 30   ··#& ··· M·····M0
0060  63 2e f2 95 94 26 ea 29  87 29 0c d1 07 c1 02 54   c.···&·) ·)·····T
0070  ce e3 cc 73 8f 04 58 29  ef 52 13 af bc e4 54 67   ···s··X) ·R····Tg
0080  47 dd a7 96 c9 46 53 cc  51 72 4e 2c 41 13 8d 81   G····FS· QrN,A···
0090  8c f5 36 59 cd 28 04 90  62 5b b5 4a f9 3e 1a 27   ··6Y·(·· b[·J·>·'
00a0  a6 db 6b 29 22 31 e2 23  09 05 1a 18 3e 02 b9 99   ··k)"1·# ····>···
00b0  52 ac 3d b5 ae 11 83 6a  be f6 f2 b7 25 f9 ee 64   R·=····j ····%··d
00c0  8b cf 2c af 57 c8 a5 2f  87 33 5f a0 23 1c dd 05   ··,·W··/ ·3_·#···
00d0  95 d9 8d c2 ce 39 6c ec  02 43 ce d0 ef 62 f8 f6   ·····9l· ·C···b··
00e0  eb 33 74 09 e4 62 d3 72  bc 42 ed 6b f6 76 fb 09   ·3t··b·r ·B·k·v··
00f0  15 30 56 4e e7 f3 ce 12  7b c4 59 67 09 97 59 2e   ·0VN···· {·Yg··Y.
0100  bc 2f f8 b9 f6 82 eb cd  01 fb 32 60 eb d9 b0 c1   ·/····· ··2`····
0110  a9 2f 51 ce 31 65 b0 88  0b ec 43 08 ad 25 fa 7b   ·/Q·1e·· ··C··%·{
0120  a8 92 fd 4c b6 7e ce 6f  7c c5 c8 ff 57 e1 3a d1   ···L·~·o |···W·:·
0130  93 25 d2 9f 36 97 b5 53  22 f7 6b 8d 18 94 fa 4f   ·%··6··S "·k····O
0140  dc 54 97 81 56 00 14 85  e6 7f ce f4 23 88 64 14   ·T··V··· ····#·d·
0150  be f2 aa bf e7 81 33 f0  8f 04 d2 ff e7 44 25 f7   ····· 3· ·····D%·
0160  af 3c 4f e9 51 34 7b 74  b9 45 d0 a2 ab 5b 49 7b   ·<O·Q4{t ·E···[I{
0170  0d 5e 18 98 e7 00 c0 46  81 b3 a2 22 ce 10 fb 10   ·^·····F ···"····
0180  35 86 3c 43 7e fe 14 83  9b 52 8a 6f 04 e0 8f 0a   5·<C~··· ·R·o····
0190  1a 9c 9d 12 79 fd 47 0d  76 52 52 3f 68 94 1f 6e   ····y·G· vRR?h··n
01a0  99 59 7e ef 60 36 86 3b  8b 2a 62 09 db f8 8b d5   ·Y~·`6·; ·*b·····
01b0  35 87 09 80 3a 37 fd 34  9c 5e ec ef 5f 12 05 b6   5···:7·4 ·^··_···
```

Decrypted data is garbage

Frame (2954 bytes) | Decrypted NTLMSSP Verifier (12 bytes) | Decrypted data (2852 bytes) | Decrypted stub data (2852 bytes)

create two tasks then enum RPC_C_AUTHN_LEVEL_PKT_PRIVACY.pcapng | Packets: 41 · Displayed: 26 (63.4%) | Profile: Default

tenable

In Wireshark 😭

Wireshark Foundation › wireshark › Issues › #19000

Closed | Issue created 2 months ago by Clément Notin Contributor | Edit | Reopen issue

NTLMSSP decryption works only for the first RPC call then fails for the others following

Summary

I observed that during a DCE/RPC exchange, the NTLMSSP decryption feature works well for the first RPC call (request & response are properly decrypted), but then it fails for all the following calls (decrypted data is garbage).

This is a bug that I initially discovered when working on my SharkFest EU 22 talk, you can see it live: https://youtu.be/5O1tKxEa1iY?t=2100

I'm currently investigating the issue, but I'm reporting it already 😊

Steps to reproduce

It can be reproduced using the following Python3 script, which uses the impacket library. It runs fine on a Linux system, against a Windows target. Admin privileges on the remote Windows target are required.

```
from impacket.dcerpc.v5 import transport, rpcrt, epm, tsch
from impacket.dcerpc.v5.dtypes import NULL
```

Source: https://gitlab.com/wireshark/wireshark/-/issues/19000

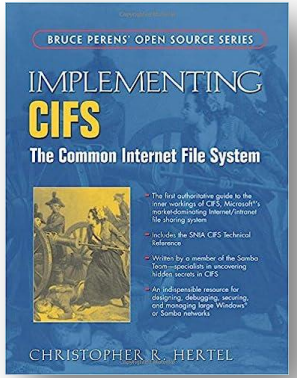
tenable

NTLM🔍

Brain Overflow Alert:

The next section describes the NTLMv2 algorithm. It's not really that difficult, but it can get tedious--especially if your head is still swimming from the LM and NTLM algorithms. Jerry Carter of the Samba Team warns that your brain may explode if you try to understand it all the first time through. (Most veteran CIFS engineers have had this happen at least twice.)

You may want to skim through section 2.8.5 and possibly section 2.8.9, which describes **M**essage **A**uthentication **C**odes (MACs). You can always come back and read them again after you've iced your cranium.



Source: https://book.huihoo.com/implementing-cifs/SMB.html

NTLM

davenport.sourceforge.net/ntlm.html

The NTLM Authentication Protocol and Security Support Provider

Abstract

This article seeks to describe the NTLM authentication protocol and related security support provider functionality at an intermediate to advanced level of detail, suitable as a reference for implementors. It is hoped that this document will evolve into a comprehensive description of NTLM; at this time there are omissions, both in the author's knowledge and in his documentation, and almost certainly inaccuracies. However, this document should at least be able to provide a solid foundation for further research. The information presented herein was used as the basis for the implementation of NTLM authentication in the open-source jCIFS library, available at http://jcifs.samba.org. This documentation is based on independent research by the author and analysis of functionality implemented in the Samba software suite.

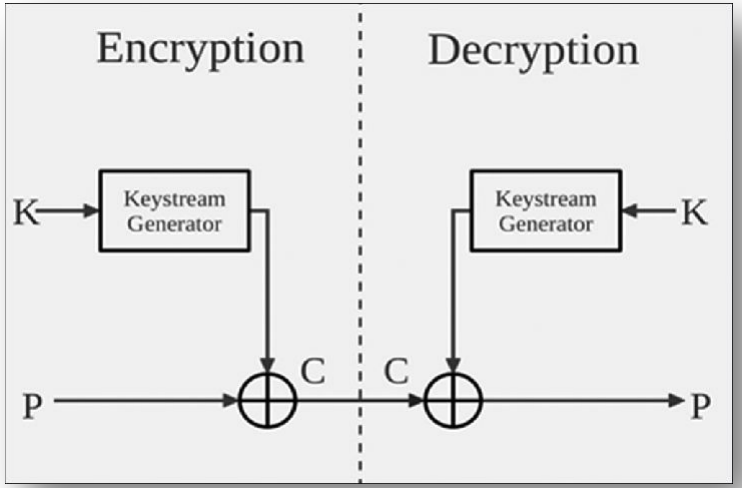
Contents

Source: https://davenport.sourceforge.net/ntlm.html

tenable

NTLM🔍

*"the **same RC4 keystream is used for both signing and sealing** [...] An RC4 cipher is initialized using the previously negotiated key. This is done once (before the first signing operation), and the **keystream is never reset**."*



Source: https://davenport.sourceforge.net/ntlm.html

tenable

NTLM🔍

Encryption by client/server

```
EncryptMessage(req1, keystream)
MakeSignature(req1, keystream)

EncryptMessage(req2, keystream)
MakeSignature(req2, keystream)
```

Decryption by Wireshark

```
DecryptMessage(req1, keystream)
MakeSignature(req1, keystream)

DecryptMessage(req2, keystream)
MakeSignature(req2, keystream)
```

Wireshark bug

Encryption by client/server

```
EncryptMessage(req1, keystream)
MakeSignature(req1, keystream)

EncryptMessage(req2, keystream)
MakeSignature(req2, keystream)
```

Decryption by Wireshark

```
DecryptMessage(req1, keystream)
```
~~MakeSignature(req1, keystream)~~

```
DecryptMessage(req2, keystream)
```
~~MakeSignature(req2, keystream)~~

tenable

Fixed 💪

fixed in v4.0.6
backported to v3.6.14

Wireshark Foundation › wireshark › Merge requests › !10474

ntlmssp: fix decrypt error by decrypting verifier after payload

Merged **Clément Notin** requested to merge `cnotin/wireshark:pr-fix-…` into `master` 2 months ago

Overview 6 Commits 1 Pipelines 11 Changes 1

Fix #19000 (closed)

As described in #19000 (comment 1357871061), the issue was that the DCERPC calls the underlying dissector (NTLMSSP in our case) first to deal with the verifier `dissect_auth_verf`, then to deal with the payload `decode_encrypted_data`. The issue is that previously, the NTLMSSP code to deal with the verifier also decrypted it. Actually it failed because it couldn't find the packet info which wasn't created yet. Then the payload was decrypted, before going to the next request. The issue is that the verifier was never decrypted, and since the same RC4 cipher is used without reset for all subsequent requests, the cipher is shifted and it doesn't decrypt properly anymore.

My suggestion here is to skip decrypting the verifier, just storing its encrypted value. Then when the payload decryption is called, decrypt it, then call the function to finally decrypt the verifier. So the verifier is always decrypted after decrypting the payload, and before decrypting the next payload from the next request.

I'm confident about that logic, but I'm not very confident with my code unfortunately 🤔, also because I don't know all the Wireshark concepts (the `tree`, the `key`, etc.) so please review it carefully 🔍 I managed to not touch the dcerpc code itself, nor any other dissector, just NTLMSSP so the impact is minimal...

Of course I confirm that the decryption of the entire RPC exchange from the PCAP shared in #19000 (closed) now works great! 🎉

👍 0 👎 0

Merge request pipeline #855328290 passed for 6d466cab 2 months ago

Approved by

Merged by John Thacker 2 months ago

Source: https://gitlab.com/wireshark/wireshark/-/merge_requests/10474

create_two_tasks_then_enum_RPC_C_AUTHN_LEVEL_PKT_PRIVACY_NTLMv2_password_clem.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

dcerpc

No.	Time	Source	Destination	Protocol	Length	Source Port	Destination Port	Info
4	23:17:12,519480964	192.168.80.128	192.168.42.2	DCERPC	166	41722	49154	Bind: call_id: 1, Fragment: Single, 1 context items: TaskScheduler
6	23:17:12,520545557	192.168.42.2	192.168.80.128	DCERPC	338	49154	41722	Bind_ack: call_id: 1, Fragment: Single, max_xmit: 4280 max_recv: 4
8	23:17:12,526323317	192.168.80.128	192.168.42.2	DCERPC	438	41722	49154	AUTH3: call_id: 1, Fragment: Single, NTLMSSP_AUTH, User: cnn-lab.l
10	23:17:12,527899906	192.168.80.128	192.168.42.2	TaskSchedulerService	2954	41722	49154	SchRpcRegisterTask request
13	23:17:12,553455927	192.168.42.2	192.168.80.128	TaskSchedulerService	166	49154	41722	SchRpcRegisterTask response
15	23:17:12,556213808	192.168.80.128	192.168.42.2	TaskSchedulerService	2954	41722	49154	SchRpcRegisterTask request
18	23:17:12,643620795	192.168.42.2	192.168.80.128	TaskSchedulerService	166	49154	41722	SchRpcRegisterTask response
20	23:17:12,646182977	192.168.80.128	192.168.42.2	TaskSchedulerService	130	41722	49154	SchRpcEnumTasks request
22	23:17:12,648513562	192.168.42.2	192.168.80.128	TaskSchedulerService	1302	49154	41722	SchRpcEnumTasks response

1

```
> Frame 10: 2954 bytes on wire (23632 bits), 2954 bytes captured (2
> Ethernet II, Src: VMware_08:66:01 (00:0c:29:08:66:01), Dst: VMwar
> Internet Protocol Version 4, Src: 192.168.80.128 (192.168.80.128)
> Transmission Control Protocol, Src Port: 41722, Dst Port: 49154,
> Distributed Computing Environment / Remote Procedure Call (DCE/RP
> Microsoft Task Scheduler Service, SchRpcRegisterTask
```

```
0000  67 15 00 00 13 00 00 00  00 00 00 00 13 00 00 00  g....... ........
0010  5c 00 66 00 69 00 72 00  73 00 74 00 74 00 61 00  \.f.i.r. s.t.t.a.
0020  73 00 6b 00 6a 00 77 00  6c 00 70 00 6d 00 64 00  s.k.j.w. l.p.m.d.
0030  49 00 66 00 00 00 ab ab  65 05 00 00 00 00 00 00  I.f..... e.......
0040  65 05 00 00 3c 00 3f 00  78 00 6d 00 6c 00 20 00  e...<.?. x.m.l. .
0050  76 00 65 00 72 00 73 00  69 00 6f 00 6e 00 3d 00  v.e.r.s. i.o.n.=.
0060  22 00 31 00 2e 00 30 00  22 00 20 00 65 00 6e 00  ".1..0.. ". .e.n.
0070  63 00 6f 00 64 00 69 00  6e 00 67 00 3d 00 22 00  c.o.d.i. n.g.=.".
0080  55 00 54 00 46 00 2d 00  31 00 36 00 22 00 3f 00  U.T.F.-. 1.6.".?.
0090  3e 00 0a 00 3c 00 54 00  61 00 73 00 6b 00 20 00  >...<.T. a.s.k. .
00a0  76 00 65 00 72 00 73 00  69 00 6f 00 6e 00 3d 00  v.e.r.s. i.o.n.=.
00b0  22 00 31 00 2e 00 32 00  22 00 20 00 78 00 6d 00  ".1..2.. ". .x.m.
00c0  6c 00 6e 00 73 00 3d 00  22 00 68 00 74 00 74 00  l.n.s.=. ".h.t.t.
00d0  70 00 3a 00 2f 00 2f 00  73 00 63 00 68 00 65 00  p.:././. s.c.h.e.
00e0  6d 00 61 00 73 00 2e 00  6d 00 69 00 63 00 72 00  m.a.s... m.i.c.r.
00f0  6f 00 73 00 6f 00 66 00  74 00 2e 00 63 00 6f 00  o.s.o.f. t...c.o.
0100  6d 00 2f 00 77 00 69 00  6e 00 64 00 6f 00 77 00  m./.w.i. n.d.o.w.
0110  73 00 2f 00 32 00 30 00  30 00 34 00 2f 00 30 00  s./.2.0. 0.4./.0.
0120  32 00 2f 00 6d 00 69 00  74 00 2f 00 74 00 61 00  2./.m.i. t./.t.a.
0130  73 00 6b 00 22 00 3e 00  0a 00 20 00 20 00 3c 00  s.k.".>. .. . .<.
0140  54 00 72 00 69 00 67 00  67 00 65 00 72 00 73 00  T.r.i.g. g.e.r.s.
0150  3e 00 0a 00 20 00 20 00  20 00 20 00 3c 00 43 00  >... . . . .<.C.
0160  61 00 6c 00 65 00 6e 00  64 00 61 00 72 00 54 00  a.l.e.n. d.a.r.T.
0170  72 00 69 00 67 00 67 00  65 00 72 00 3e 00 0a 00  r.i.g.g. e.r.>...
0180  20 00 20 00 20 00 20 00  20 00 20 00 3c 00 53 00  . . . . . .<.S.
0190  74 00 61 00 72 00 74 00  42 00 6f 00 75 00 6e 00  t.a.r.t. B.o.u.n.
01a0  64 00 61 00 72 00 79 00  3e 00 32 00 30 00 31 00  d.a.r.y. >.2.0.1.
01b0  35 00 2d 00 30 00 37 00  2d 00 31 00 35 00 54 00  5.-.0.7. -.1.5.T.
01c0  32 00 30 00 3a 00 33 00  35 00 3a 00 31 00 33 00  2.0.:.3. 5.:.1.3.
01d0  2e 00 32 00 37 00 35 00  37 00 32 00 39 00 34 00  ..2.7.5. 7.2.9.4.
01e0  3c 00 2f 00 53 00 74 00  61 00 72 00 74 00 42 00  <./.S.t. a.r.t.B.
```

Frame (2954 bytes) | Decrypted data (2852 bytes) | Decrypted NTLMSSP Verifier (12 bytes) | Decrypted stub data (2852 bytes)

Distributed Computing Environment / Remote Procedure Call (DCE/RPC): Protocol | Selected Packet: 10 · Packets: 26 · Displayed: 9 (34.6%) | Profile: Default

create_two_tasks_then_enum_RPC_C_AUTHN_LEVEL_PKT_PRIVACY_NTLMv2__password_clem.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

dcerpc

No.	Time	Source	Destination	Protocol	Length	Source Port	Destination Port	Info
4	23:17:12,519480964	192.168.80.128	192.168.42.2	DCERPC	166	41722	49154	Bind: call_id: 1, Fragment: Single, 1 context items: TaskScheduler
6	23:17:12,520545557	192.168.42.2	192.168.80.128	DCERPC	338	49154	41722	Bind_ack: call_id: 1, Fragment: Single, max_xmit: 4280 max_recv: 4
8	23:17:12,526323317	192.168.80.128	192.168.42.2	DCERPC	438	41722	49154	AUTH3: call_id: 1, Fragment: Single, NTLMSSP_AUTH, User: cnn-lab.l
10	23:17:12,527899906	192.168.80.128	192.168.42.2	TaskSchedulerService	2954	41722	49154	SchRpcRegisterTask request
13	23:17:12,553455927	192.168.42.2	192.168.80.128	TaskSchedulerService	166	49154	41722	SchRpcRegisterTask response
15	23:17:12,556213808	192.168.80.128	192.168.42.2	TaskSchedulerService	2954	41722	49154	SchRpcRegisterTask request
18	23:17:12,643620795	192.168.42.2	192.168.80.128	TaskSchedulerService	166	49154	41722	SchRpcRegisterTask response
20	23:17:12,646182977	192.168.80.128	192.168.42.2	TaskSchedulerService	130	41722	49154	SchRpcEnumTasks request
22	23:17:12,648513562	192.168.42.2	192.168.80.128	TaskSchedulerService	1302	49154	41722	SchRpcEnumTasks response

2

```
> Frame 15: 2954 bytes on wire (23632 bits), 2954 bytes captured (2
> Ethernet II, Src: VMware_08:66:01 (00:0c:29:08:66:01), Dst: VMwar
> Internet Protocol Version 4, Src: 192.168.80.128 (192.168.80.128)
> Transmission Control Protocol, Src Port: 41722, Dst Port: 49154,
> Distributed Computing Environment / Remote Procedure Call (DCE/RP
> Microsoft Task Scheduler Service, SchRpcRegisterTask
```

```
0000  90 05 00 00 14 00 00 00  00 00 00 00 14 00 00 00   ........ ........
0010  5c 00 73 00 65 00 63 00  6f 00 6e 00 64 00 74 00   \·s·e·c· o·n·d·t·
0020  61 00 73 00 6b 00 6a 00  77 00 6c 00 70 00 6d 00   a·s·k·j· w·l·p·m·
0030  64 00 49 00 66 00 00 00  65 05 00 00 00 00 00 00   d·I·f··· e·····
0040  65 05 00 00 3c 00 3f 00  78 00 6d 00 6c 00 20 00   e···<·?· x·m·l· ·
0050  76 00 65 00 72 00 73 00  69 00 6f 00 6e 00 3d 00   v·e·r·s· i·o·n·=·
0060  22 00 31 00 2e 00 30 00  22 00 20 00 65 00 6e 00   "·1·.·0· "· ·e·n·
0070  63 00 6f 00 64 00 69 00  6e 00 67 00 3d 00 22 00   c·o·d·i· n·g·=·"·
0080  55 00 54 00 46 00 2d 00  31 00 36 00 22 00 3f 00   U·T·F·-· 1·6·"·?·
0090  3e 00 0a 00 3c 00 54 00  61 00 73 00 6b 00 20 00   >···<·T· a·s·k· ·
00a0  76 00 65 00 72 00 73 00  69 00 6f 00 6e 00 3d 00   v·e·r·s· i·o·n·=·
00b0  22 00 31 00 2e 00 32 00  22 00 20 00 78 00 6d 00   "·1·.·2· "· ·x·m·
00c0  6c 00 6e 00 73 00 3d 00  22 00 68 00 74 00 74 00   l·n·s·=· "·h·t·t·
00d0  70 00 3a 00 2f 00 2f 00  73 00 63 00 68 00 65 00   p·:·/·/· s·c·h·e·
00e0  6d 00 61 00 73 00 2e 00  6d 00 69 00 63 00 72 00   m·a·s·.· m·i·c·r·
00f0  6f 00 73 00 6f 00 66 00  74 00 2e 00 63 00 6f 00   o·s·o·f· t·.·c·o·
0100  6d 00 2f 00 77 00 69 00  6e 00 64 00 6f 00 77 00   m·/·w·i· n·d·o·w·
0110  73 00 2f 00 32 00 30 00  30 00 34 00 2f 00 30 00   s·/·2·0· 0·4·/·0·
0120  32 00 2f 00 6d 00 69 00  74 00 2f 00 74 00 61 00   2·/·m·i· t·/·t·a·
0130  73 00 6b 00 22 00 3e 00  0a 00 20 00 20 00 3c 00   s·k·"·>· ··· ·<·
0140  54 00 72 00 69 00 67 00  67 00 65 00 72 00 73 00   T·r·i·g· g·e·r·s·
0150  3e 00 0a 00 20 00 20 00  20 00 20 00 3c 00 43 00   >··· ·· ·· ··<·C·
0160  61 00 6c 00 65 00 6e 00  64 00 61 00 72 00 54 00   a·l·e·n· d·a·r·T·
0170  72 00 69 00 67 00 67 00  65 00 72 00 3e 00 0a 00   r·i·g·g· e·r·>···
0180  20 00 20 00 20 00 20 00  20 00 20 00 3c 00 53 00    ·· ·· ·· ··<·S·
0190  74 00 61 00 72 00 74 00  42 00 6f 00 75 00 6e 00   t·a·r·t· B·o·u·n·
01a0  64 00 61 00 72 00 79 00  3e 00 32 00 30 00 31 00   d·a·r·y· >·2·0·1·
01b0  35 00 2d 00 30 00 37 00  2d 00 31 00 35 00 54 00   5·-·0·7· -·1·5·T·
01c0  32 00 30 00 3a 00 33 00  35 00 3a 00 31 00 33 00   2·0·:·3· 5·:·1·3·
01d0  2e 00 32 00 37 00 35 00  37 00 32 00 39 00 34 00   ·2·7·5· 7·2·9·4·
01e0  3c 00 2f 00 53 00 74 00  61 00 72 00 74 00 42 00   <·/·S·t· a·r·t·B·
```

Frame (2954 bytes) | Decrypted data (2852 bytes) | Decrypted NTLMSSP Verifier (12 bytes) | Decrypted stub data (2852 bytes)

Distributed Computing Environment / Remote Procedure Call (DCE/RPC): Protocol | Selected Packet: 15 · Packets: 26 · Displayed: 9 (34.6%) | Profile: Default

create_two_tasks_then_enum_RPC_C_AUTHN_LEVEL_PKT_PRIVACY_NTLMv2__password_clem.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

dcerpc

No.	Time	Source	Destination	Protocol	Length	Source Port	Destination Port	Info
4	23:17:12,519480964	192.168.80.128	192.168.42.2	DCERPC	166	41722	49154	Bind: call_id: 1, Fragment: Single, 1 context items: TaskScheduler
6	23:17:12,520545557	192.168.42.2	192.168.80.128	DCERPC	338	49154	41722	Bind_ack: call_id: 1, Fragment: Single, max_xmit: 4280 max_recv: 4
8	23:17:12,526323317	192.168.80.128	192.168.42.2	DCERPC	438	41722	49154	AUTH3: call_id: 1, Fragment: Single, NTLMSSP_AUTH, User: cnn-lab.l
10	23:17:12,527899906	192.168.80.128	192.168.42.2	TaskSchedulerService	2954	41722	49154	SchRpcRegisterTask request
13	23:17:12,553455927	192.168.42.2	192.168.80.128	TaskSchedulerService	166	49154	41722	SchRpcRegisterTask response
15	23:17:12,556213808	192.168.80.128	192.168.42.2	TaskSchedulerService	2954	41722	49154	SchRpcRegisterTask request
18	23:17:12,643620795	192.168.42.2	192.168.80.128	TaskSchedulerService	166	49154	41722	SchRpcRegisterTask response
20	23:17:12,646182977	192.168.80.128	192.168.42.2	TaskSchedulerService	130	41722	49154	SchRpcEnumTasks request
22	23:17:12,648513562	192.168.42.2	192.168.80.128	TaskSchedulerService	1302	49154	41722	SchRpcEnumTasks response

3

```
> Frame 22: 1302 bytes on wire (10416 bits), 1302 bytes captured (1
> Ethernet II, Src: VMware_f3:9b:97 (00:50:56:f3:9b:97), Dst: VMwar
> Internet Protocol Version 4, Src: 192.168.42.2 (192.168.42.2), Ds
> Transmission Control Protocol, Src Port: 49154, Dst Port: 41722,
> Distributed Computing Environment / Remote Procedure Call (DCE/RP
> Microsoft Task Scheduler Service, SchRpcEnumTasks
```

```
0000  17 00 00 00 17 00 00 00  00 00 02 00 17 00 00 00   ........ ........
0010  04 00 02 00 08 00 02 00  0c 00 02 00 10 00 02 00   ........ ........
0020  14 00 02 00 18 00 02 00  1c 00 02 00 20 00 02 00   ........ ....  ...
0030  24 00 02 00 28 00 02 00  2c 00 02 00 30 00 02 00   $···(··· ,···0···
0040  34 00 02 00 38 00 02 00  3c 00 02 00 40 00 02 00   4···8··· <···@···
0050  44 00 02 00 48 00 02 00  4c 00 02 00 50 00 02 00   D···H··· L···P···
0060  54 00 02 00 58 00 02 00  5c 00 02 00 09 00 00 00   T···X··· \·......
0070  00 00 00 00 09 00 00 00  45 00 50 00 46 00 53 00   ........ E·P·F·S·
0080  76 00 62 00 61 00 4c 00  00 00 00 00 12 00 00 00   v·b·a·L· ........
0090  00 00 00 00 12 00 00 00  66 00 69 00 72 00 73 00   ........ f·i·r·s·
00a0  74 00 74 00 61 00 73 00  6b 00 61 00 66 00 67 00   t·t·a·s· k·a·f·g·
00b0  62 00 6b 00 4f 00 65 00  76 00 00 00 12 00 00 00   b·k·O·e· v·......
00c0  00 00 00 00 12 00 00 00  66 00 69 00 72 00 73 00   ........ f·i·r·s·
00d0  74 00 74 00 61 00 73 00  6b 00 42 00 76 00 71 00   t·t·a·s· k·B·v·q·
00e0  5a 00 78 00 4e 00 54 00  71 00 00 00 12 00 00 00   Z·x·N·T· q·......
00f0  00 00 00 00 12 00 00 00  66 00 69 00 72 00 73 00   ........ f·i·r·s·
0100  74 00 74 00 61 00 73 00  6b 00 43 00 63 00 44 00   t·t·a·s· k·C·c·D·
0110  62 00 66 00 67 00 73 00  67 00 00 00 12 00 00 00   b·f·g·s· g·......
0120  00 00 00 00 12 00 00 00  66 00 69 00 72 00 73 00   ........ f·i·r·s·
0130  74 00 74 00 61 00 73 00  6b 00 63 00 7a 00 4c 00   t·t·a·s· k·c·z·L·
0140  68 00 41 00 61 00 61 00  44 00 00 00 12 00 00 00   h·A·a·a· D·......
0150  00 00 00 00 12 00 00 00  66 00 69 00 72 00 73 00   ........ f·i·r·s·
0160  74 00 74 00 61 00 73 00  6b 00 6a 00 77 00 6c 00   t·t·a·s· k·j·w·l·
0170  70 00 6d 00 64 00 49 00  66 00 00 00 12 00 00 00   p·m·d·I· f·......
0180  00 00 00 00 12 00 00 00  66 00 69 00 72 00 73 00   ........ f·i·r·s·
0190  74 00 74 00 61 00 73 00  6b 00 58 00 48 00 54 00   t·t·a·s· k·X·H·T·
01a0  65 00 41 00 41 00 4c 00  62 00 00 00 09 00 00 00   e·A·A·L· b·......
01b0  00 00 00 00 09 00 00 00  47 00 49 00 70 00 65 00   ........ G·I·p·e·
01c0  77 00 7a 00 51 00 62 00  00 00 00 00 09 00 00 00   w·z·Q·b· ........
01d0  00 00 00 00 09 00 00 00  4b 00 4e 00 70 00 72 00   ........ K·N·p·r·
01e0  4d 00 51 00 4f 00 4f 00  00 00 00 00 4e 00 00 00   M·Q·O·O· ····N···
```

Frame (1302 bytes) | Decrypted data (1200 bytes) | Decrypted NTLMSSP Verifier (12 bytes) | Decrypted stub data (1200 bytes)

Distributed Computing Environment / Remote Procedure Call (DCE/RPC): Protocol | Selected Packet: 22 · Packets: 26 · Displayed: 9 (34.6%) | Profile: Default