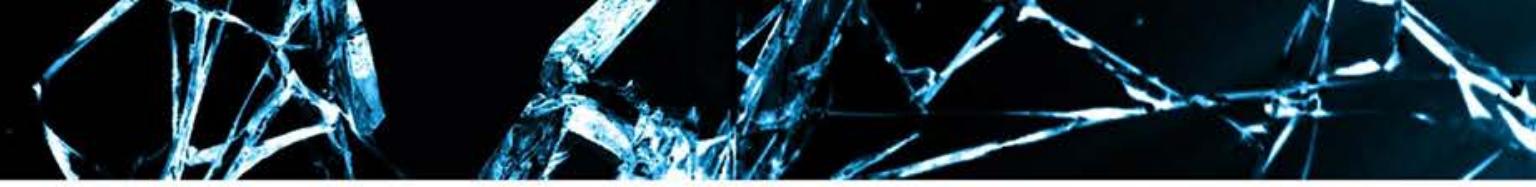




USA 2016

JULY 30 - AUGUST 4, 2016 / MANDALAY BAY / LAS VEGAS



Cunning with CNG: Soliciting Secrets from Schannel

“Black Hat Sound Bytes”

What you get out of this talk

- Ability to decrypt Schannel TLS connections that use ephemeral key exchanges
- Ability to decrypt and extract private certificate and session ticket key directly from memory
- Public Cert/SNI to PID/Logon Session Mapping





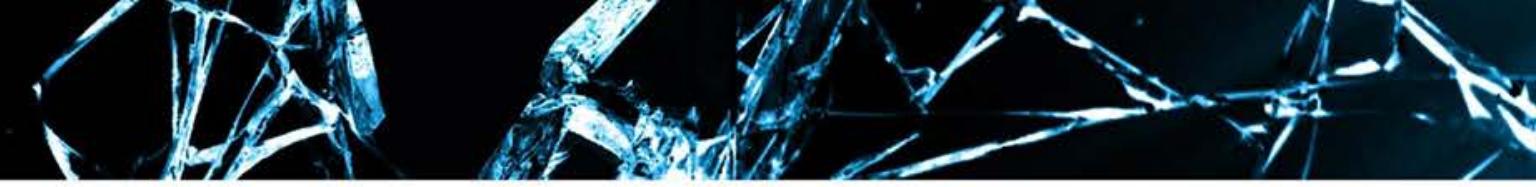
Agenda

- ❑ A very short SSL/TLS Review
- ❑ A background on Schannel & CNG
- ❑ The Secret Data
- ❑ The Forensic Context
- ❑ Demo >.>



Disclaimer

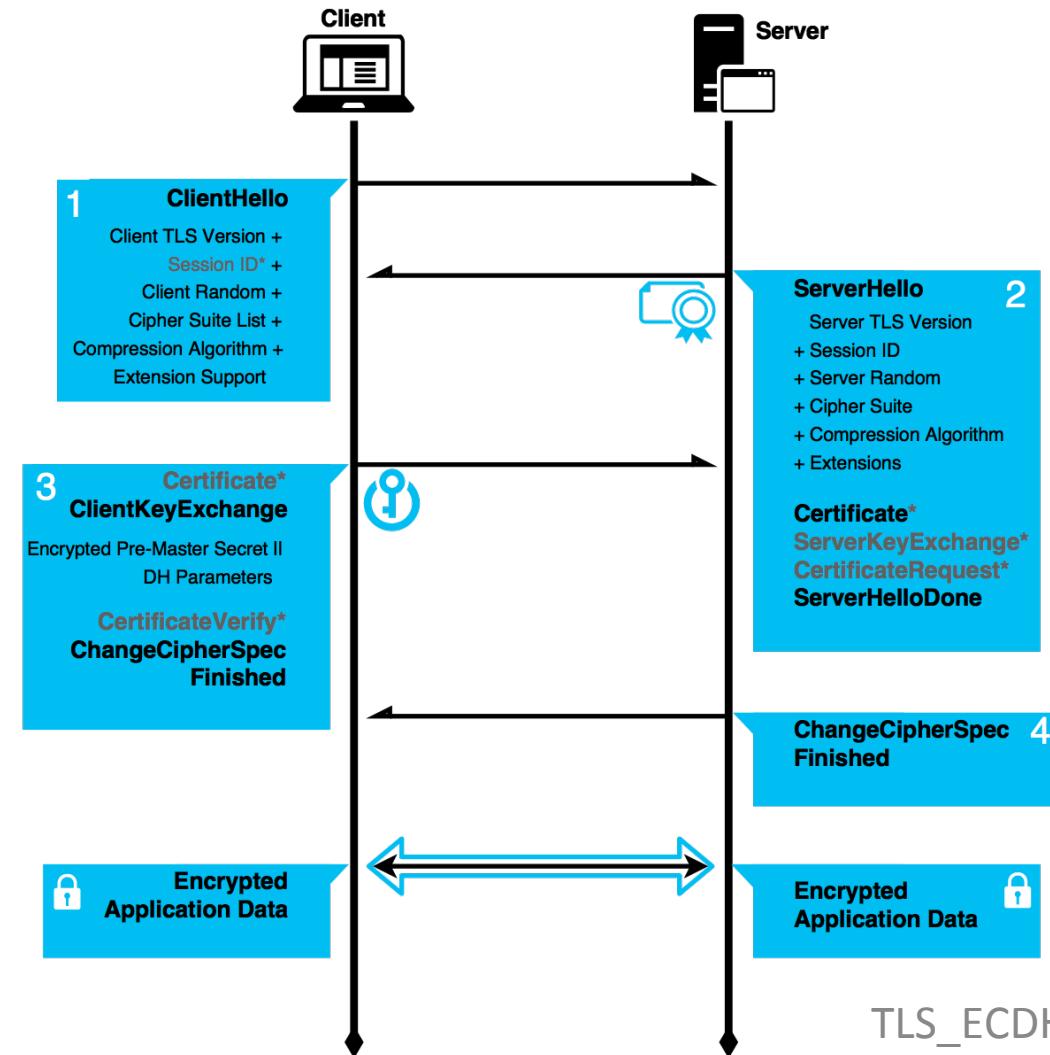
- ❑ This is NOT an exploit
 - ❑ It's just the spec :D
 - ❑ ...and some implementation specific oddities
- ❑ Microsoft has done nothing [especially] wrong
 - ❑ To the contrary, their documentation was actually pretty great
- ❑ Windows doesn't track sessions for processes that load their own TLS libs
 - ❑ I'm looking at you Firefox and Chrome
- ❑ Windows doesn't track sessions for process that don't use TLS...
 - ❑ That'd be you TeamViewer...



Background

TLS, Schannel, and CNG

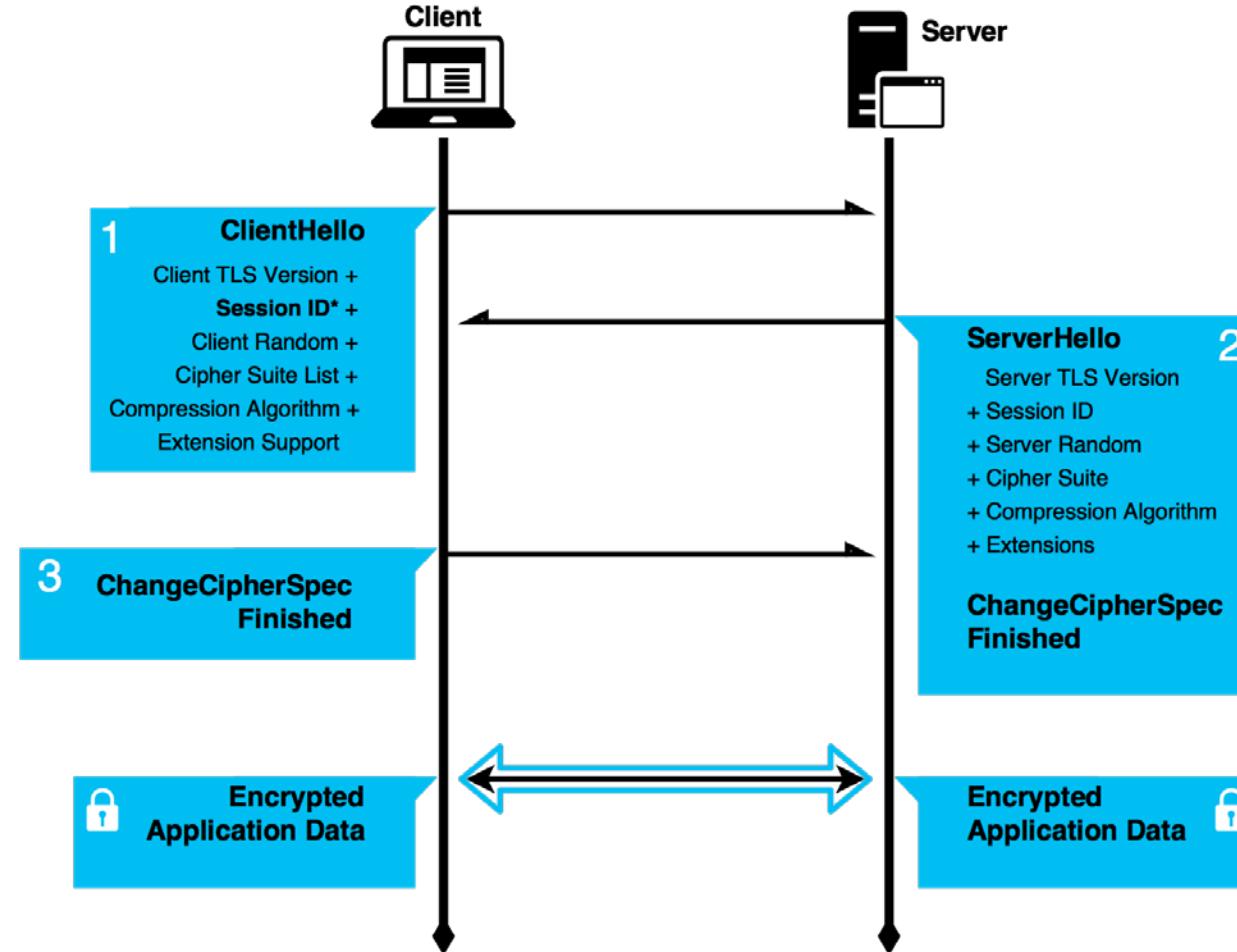
The infamous TLS Handshake



Initial Connection

TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

The ~~infamous~~ TLS Handshake



Session Resumption

Perfect Forward Secrecy

What we *want* to do

- One time use keys, no sending secrets!

What TLS *actually* does

- Caches values to enable resumption
 - recommends 'An upper limit of 24 hours is suggested for session ID lifetimes'
- When using session ticket extension, sends the encrypted state over the network
 - basically returning to the issue with RSA, but using a more ephemeral key...

What implementations *also* do

- Store symmetric key schedules (so you can find the otherwise random keys...)
- Cache ephemeral keys and reuse for a while...



Schannel & CNG

Secure Channel

- ❑ It's TLS -> the Secure Channel for Windows!
- ❑ A library that gets loaded into the “key isolation process” **and** the “client” process
 - ❑ Technically a Security Support Provider (SSP)
- ❑ Spoiler: the Key Isolation process is LSASS

The CryptoAPI-Next Generation (CNG)

- ❑ Introduced in Vista (yes you read correctly)
- ❑ Provides Common Criteria compliance
- ❑ Used to store secrets and ‘crypt them
 - ❑ Storage via the Key Storage Providers (KSPs)
 - ❑ Generic data encryption via DPAPI
 - ❑ Also brings modern ciphers to Windows (AES for example) and ECC
- ❑ Importantly, **ncrypt** gets called out as the “key storage router” and gateway to the CNG Key Isolation service

Schannel Preferred Cipher Suites

```
Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 6.0.6002]
Copyright (c) 2006 Microsoft Corporation. All rights reserved.

C:\Windows\system32>Temp\ListCipherSuites.exe

Sample Code from: https://msdn.microsoft.com/en-us/library/windows
930.aspx

  Listing Cipher Suites
TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_RSA_WITH_RC4_128_SHA
TLS_RSA_WITH_3DES_EDE_CBC_SHA
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA_P256
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA_P384
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA_P521
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA_P256
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA_P384
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA_P521
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA_P256
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA_P384
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA_P521
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA_P256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA_P384
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA_P521
TLS_DHE_DSS_WITH_AES_128_CBC_SHA
TLS_DHE_DSS_WITH_AES_256_CBC_SHA
TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA
TLS_RSA_WITH_RC4_128_MD5
SSL_CK_RC4_128_WITH_MD5
SSL_CK_DES_192_EDE3_CBC_WITH_MD5
TLS_RSA_WITH_NULL_MD5
TLS_RSA_WITH_NULL_SHA
C:\Windows\system32>
```

```
Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>\Temp\ListCipherSuites.exe

Sample Code from: https://msdn.microsoft.com/en-us/library/windo
930.aspx

Listing Cipher Suites
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384_P256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384_P384
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384_P521
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256_P256
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256_P384
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256_P521
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA_P256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA_P384
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA_P521
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA_P256
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA_P384
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA_P521
TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
TLS_RSA_WITH_AES_256_GCM_SHA384
TLS_RSA_WITH_AES_128_GCM_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA256
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384_P384
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384_P521
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256_P256
```

```
C:\Administrator: Command Prompt - powershell
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Windows\system32>powershell
Windows PowerShell
Copyright (C) 2015 Microsoft Corporation. All rights reserved.

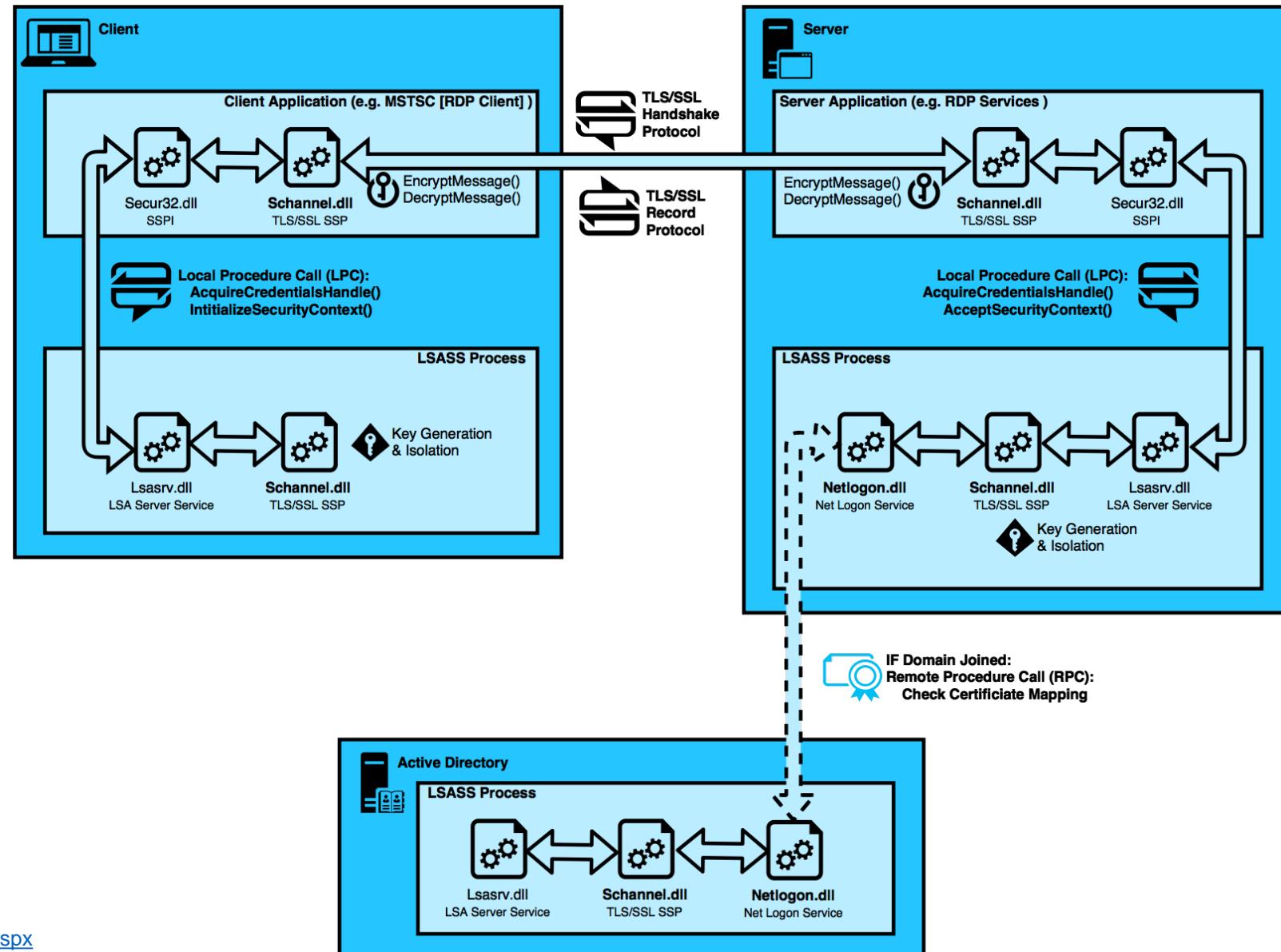
PS C:\Windows\system32> $(Get-TlsCipherSuite).Name
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
TLS_DHE_RSA_WITH_AES_256_CBC_SHA
TLS_DHE_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_GCM_SHA384
TLS_RSA_WITH_AES_128_GCM_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA256
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_3DES_EDE_CBC_SHA
TLS_DHE_DSS_WITH_AES_256_CBC_SHA256
```

*ListCipherSuites sample code found here: <https://technet.microsoft.com/en-us/library/bb870930.aspx>

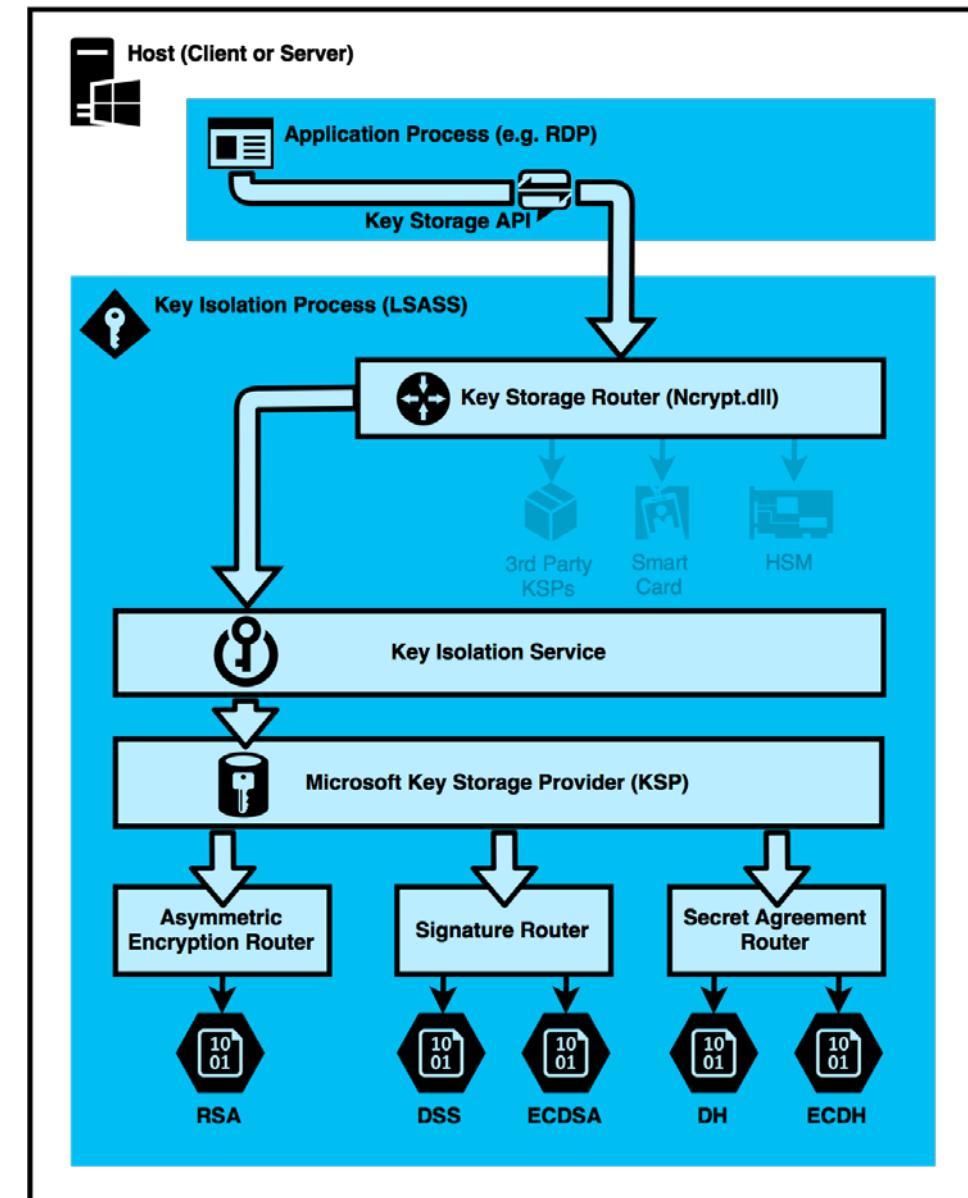
Microsoft's TLS/SSL Docs

- **ClientCacheTime:** “The first time a client connects to a server through the Schannel SSP, a full TLS/SSL handshake is performed.”
- “When this is complete, **the master secret, cipher suite, and certificates are stored** in the session cache on the respective client and server.”*
- **ServerCacheTime:** “...Increasing ServerCacheTime above the default values **causes Lsass.exe to consume additional memory**. Each session cache element typically requires 2 to 4 KB of memory”*
- **MaximumCacheSize:** “This entry controls the maximum number of cache elements. [...] **The default value is 20,000 elements.**” *

Schannel Ops

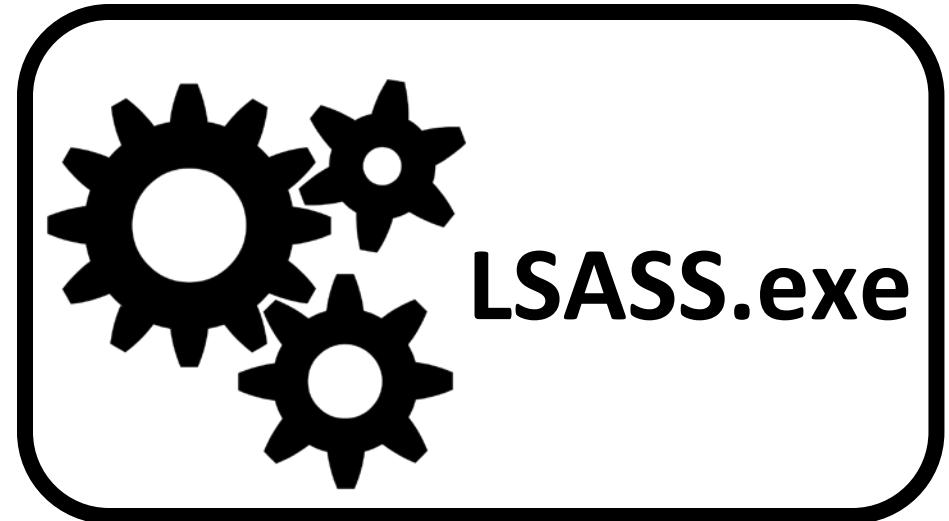


CNG Key Isolation



Background Summary

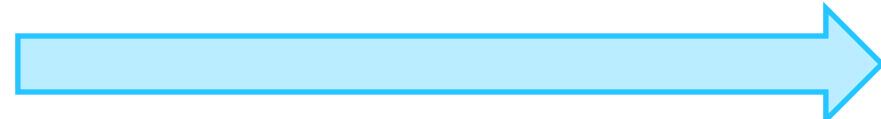
We're Looking Here



For These



Because of That



What are we trying to accomplish?



We want to be able to see data that has been protected with TLS/SSL and subvert efforts at implementing Perfect Forward Secrecy



We want to gather any contextual information that we can use for forensic purposes, regardless of whether or not we can accomplish the above



We (as an adversary) want to be able to get access to a single process address space and be able to dump out things that would enable us to monitor/modify future traffic, or possibly impersonate the target

- We want to do this without touching disk



Secrets

The Keys



Session Keys



Master Secret



Pre-Master Secret



Ephemeral Private Key*



Persistent Private Key
(Signing)

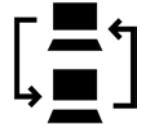


Session Ticket Key*

The Keys? What do they get us?



=



a single **connection**



=



a single **session**



=



multiple **sessions**

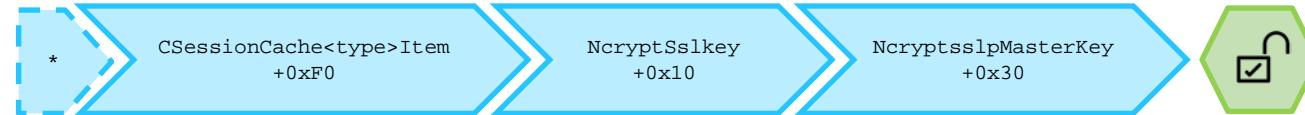


=



multiple sessions + **identity**

The Keys? We got 'em...all.

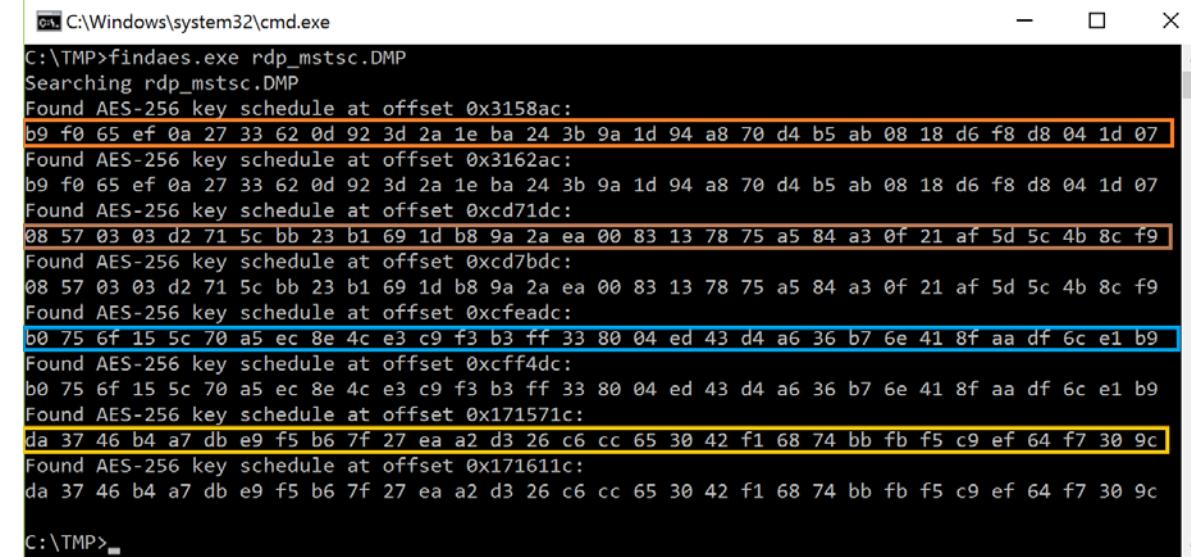


Session Keys

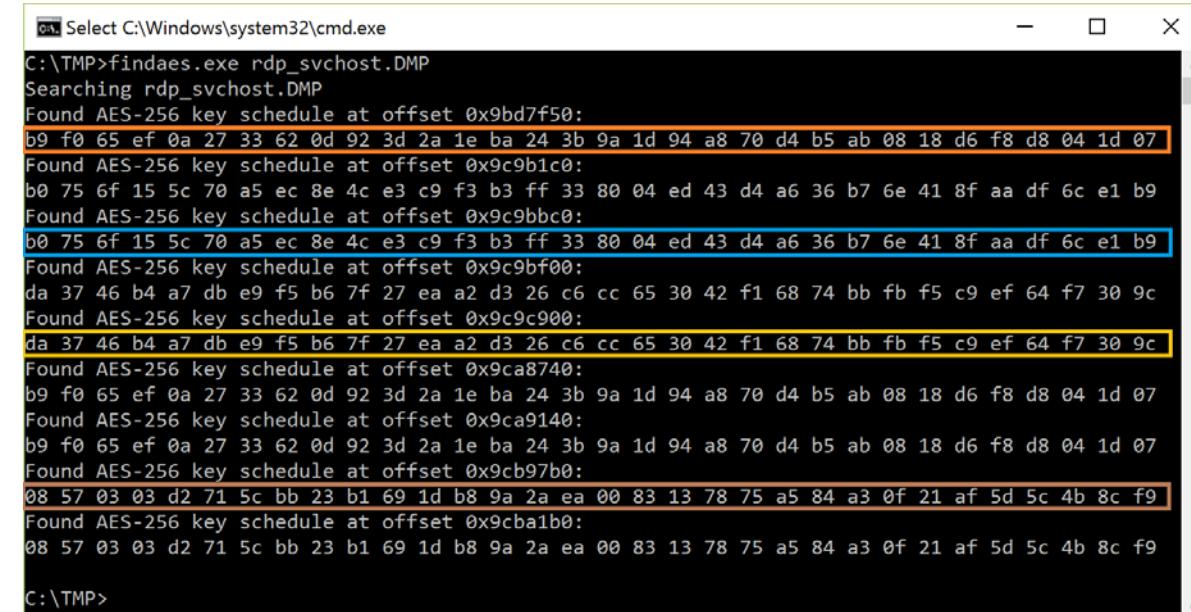
- Smallest scope / most ephemeral
- Required for symmetric encrypted comms
- Not going to be encrypted

Approach Premise:

- Start with AES
- AES keys are relatively small and pseudo-random
- AES key schedules are larger and deterministic
 - ... they are a “schedule” after all.
- Key schedules usually calculated once and stored*
- Let's scan for matching key schedules on both hosts

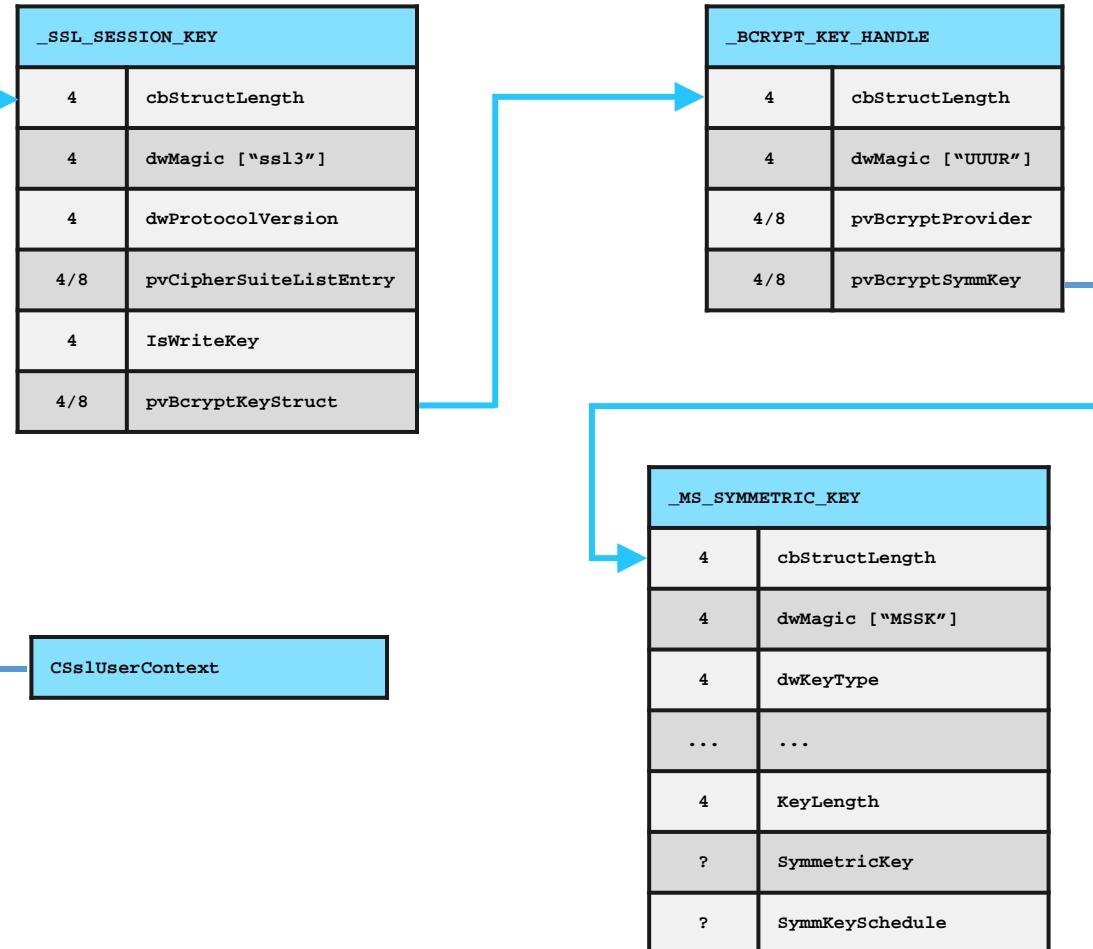


```
C:\Windows\system32\cmd.exe
C:\TMP>findaes.exe rdp_mstsc.DMP
Searching rdp_mstsc.DMP
Found AES-256 key schedule at offset 0x3158ac:
b9 f0 65 ef 0a 27 33 62 0d 92 3d 2a 1e ba 24 3b 9a 1d 94 a8 70 d4 b5 ab 08 18 d6 f8 d8 04 1d 07
Found AES-256 key schedule at offset 0x3162ac:
b9 f0 65 ef 0a 27 33 62 0d 92 3d 2a 1e ba 24 3b 9a 1d 94 a8 70 d4 b5 ab 08 18 d6 f8 d8 04 1d 07
Found AES-256 key schedule at offset 0xcd71dc:
08 57 03 03 d2 71 5c bb 23 b1 69 1d b8 9a 2a ea 00 83 13 78 75 a5 84 a3 0f 21 af 5d 5c 4b 8c f9
Found AES-256 key schedule at offset 0xcd7bdc:
08 57 03 03 d2 71 5c bb 23 b1 69 1d b8 9a 2a ea 00 83 13 78 75 a5 84 a3 0f 21 af 5d 5c 4b 8c f9
Found AES-256 key schedule at offset 0xcfeadc:
b0 75 6f 15 5c 70 a5 ec 8e 4c e3 c9 f3 b3 ff 33 80 04 ed 43 d4 a6 36 b7 6e 41 8f aa df 6c e1 b9
Found AES-256 key schedule at offset 0xcf4fdc:
b0 75 6f 15 5c 70 a5 ec 8e 4c e3 c9 f3 b3 ff 33 80 04 ed 43 d4 a6 36 b7 6e 41 8f aa df 6c e1 b9
Found AES-256 key schedule at offset 0x171571c:
da 37 46 b4 a7 db e9 f5 b6 7f 27 ea a2 d3 26 c6 cc 65 30 42 f1 68 74 bb fb f5 c9 ef 64 f7 30 9c
Found AES-256 key schedule at offset 0x171611c:
da 37 46 b4 a7 db e9 f5 b6 7f 27 ea a2 d3 26 c6 cc 65 30 42 f1 68 74 bb fb f5 c9 ef 64 f7 30 9c
C:\TMP>
```



```
Select C:\Windows\system32\cmd.exe
C:\TMP>findaes.exe rdp_svchost.DMP
Searching rdp_svchost.DMP
Found AES-256 key schedule at offset 0x9bd7f50:
b9 f0 65 ef 0a 27 33 62 0d 92 3d 2a 1e ba 24 3b 9a 1d 94 a8 70 d4 b5 ab 08 18 d6 f8 d8 04 1d 07
Found AES-256 key schedule at offset 0x9c9b1c0:
b0 75 6f 15 5c 70 a5 ec 8e 4c e3 c9 f3 b3 ff 33 80 04 ed 43 d4 a6 36 b7 6e 41 8f aa df 6c e1 b9
Found AES-256 key schedule at offset 0x9c9bbc0:
b0 75 6f 15 5c 70 a5 ec 8e 4c e3 c9 f3 b3 ff 33 80 04 ed 43 d4 a6 36 b7 6e 41 8f aa df 6c e1 b9
Found AES-256 key schedule at offset 0x9c9bf00:
da 37 46 b4 a7 db e9 f5 b6 7f 27 ea a2 d3 26 c6 cc 65 30 42 f1 68 74 bb fb f5 c9 ef 64 f7 30 9c
Found AES-256 key schedule at offset 0x9c9c900:
da 37 46 b4 a7 db e9 f5 b6 7f 27 ea a2 d3 26 c6 cc 65 30 42 f1 68 74 bb fb f5 c9 ef 64 f7 30 9c
Found AES-256 key schedule at offset 0x9ca8740:
b9 f0 65 ef 0a 27 33 62 0d 92 3d 2a 1e ba 24 3b 9a 1d 94 a8 70 d4 b5 ab 08 18 d6 f8 d8 04 1d 07
Found AES-256 key schedule at offset 0x9ca9140:
b9 f0 65 ef 0a 27 33 62 0d 92 3d 2a 1e ba 24 3b 9a 1d 94 a8 70 d4 b5 ab 08 18 d6 f8 d8 04 1d 07
Found AES-256 key schedule at offset 0x9cb97b0:
08 57 03 03 d2 71 5c bb 23 b1 69 1d b8 9a 2a ea 00 83 13 78 75 a5 84 a3 0f 21 af 5d 5c 4b 8c f9
Found AES-256 key schedule at offset 0x9cba1b0:
08 57 03 03 d2 71 5c bb 23 b1 69 1d b8 9a 2a ea 00 83 13 78 75 a5 84 a3 0f 21 af 5d 5c 4b 8c f9
C:\TMP>
```

Session Keys



Look familiar? Bcrypt keys are used a lot: think Mimikatz

Command - Dump

```

0:000> .foreach(key {s -[lw]a 0 L?800000000000 3lss}) {.echo *** Session Key ***;dd $(^
*** Session Key ***
000000e1`7b047050 00000d2e
000000e1`7b047054 73736c33 3lss
000000e1`7b047058 00000000`00000303
000000e1`7b047060 00007ffe`6fc11910 ncryptsslp!CipherSuiteList+0x1400
000000e1`7b047068 00000000`00000000
000000e1`7b047070 000000e1`7b0470c0 55555552`00000cbe
*
000000e1`7b0470c0 00000cbe
000000e1`7b0470c4 55555552 RUUU
000000e1`7b0470c8 000000e1`784e3af0 55555551`00000130
000000e1`7b0470d0 000000e1`7b0470e0 4d53534b`00000c80
000000e1`7b0470d8 00000000`00000000
*
000000e1`7b0470e0 00000c80
000000e1`7b0470e4 4d53534b KSSM
000000e1`7b0470e8 00010002 00000005 00000010 00000001
000000e1`7b0470f8 00000100 00000001
000000e1`7b047100 000000e1`784e83c0 4d535341`00000028
000000e1`7b047118 00000020
* AES Key:
000000e1`7b04711c b0 75 6f 15 5c 70 a5 ec-8e 4c e3 c9 f3 b3 ff 33 .uo.\p...L....3
000000e1`7b04712c 80 04 ed 43 d4 a6 36 b7-6e 41 8f aa df 6c el b9 ...C..6.nA....1.

*** Session Key ***
000000e1`7b047d90 00000d2e
000000e1`7b047d94 73736c33 3lss
000000e1`7b047d98 00000000`00000303
000000e1`7b047da0 00007ffe`6fc11910 ncryptsslp!CipherSuiteList+0x1400
000000e1`7b047da8 00000000`00000001
000000e1`7b047db0 000000e1`7b047e00 55555552`00000cbe
*
000000e1`7b047e00 00000cbe
000000e1`7b047e04 55555552 RUUU
000000e1`7b047e08 000000e1`784e3af0 55555551`00000130
000000e1`7b047e10 000000e1`7b047e20 4d53534b`00000c80
000000e1`7b047e18 00000000`00000000
*
000000e1`7b047e20 00000c80
000000e1`7b047e24 4d53534b KSSM
000000e1`7b047e28 00010002 00000005 00000010 00000001
000000e1`7b047e38 00000100 00000001
000000e1`7b047e40 000000e1`784e83c0 4d535341`00000028
000000e1`7b047e58 00000020
* AES Key:
000000e1`7b047e5c da 37 46 b4 a7 db e9 f5-b6 7f 27 ea a2 d3 26 c6 .7F.....`...&
000000e1`7b047e6c cc 65 30 42 f1 68 74 bb-fb f5 c9 ef 64 f7 30 9c .e0B.ht....d.0. v
< | >
0:000> ${key}+1C L1;r @$t0 = $p;.echo *;dd @$t0 L1;dc @$t0+4 L1;dpp @$t0+8 L1;dpp @
$St0+10 L2;.echo *;r $t0 = $p;dd @$t0 L1;dc @$t0+4 L1;dd @$t0+8 L6;dpp @$t0+20
L1;dd @$t0+30+$ptrsize L1;.echo * AES Key:;db @$t0+34+$ptrsize Ldwo(@$t0+30+
$ptrsize);.echo)
  
```

The Ncrypt SSL Provider (ncryptsslp.dll)

Ncryptsslp Validation function Symbols

Command - Dump \\vmware-host\Share...

```
0:000> x /1 ncryptsslp!*Validate*
ncryptsslp!SslpValidateEphemeralHandle
ncryptsslp!SslpValidateMasterKeyHandle
ncryptsslp!SslpValidateProvHandle
ncryptsslp!SslpValidateHashHandle
ncryptsslp!SslpValidateKeyPairHandle
```

0:000>

These functions do three things:

- Check the first dword for a size value
- Check the second dword for a magic ID
- Return the passed handle* if all is good

Ncryptsslp Validation function Symbols

Command - Dump \\vmware-host\Shared Folders\Documents\Challenges\zzTm...

```
0:000> uf ncryptsslp!SslpValidateMasterKeyHandle
ncryptsslp!SslpValidateMasterKeyHandle:
00007fff`df75b5b8 4885c9    test    rcx,rcx
00007fff`df75b5bb 7412    je     ncryptsslp!SslpValidateMasterKeyHandle+0x17

ncryptsslp!SslpValidateMasterKeyHandle+0x5:
00007fff`df75b5bd 833950    cmp     dword ptr [rcx],50h
00007fff`df75b5c0 720d    jb     ncryptsslp!SslpValidateMasterKeyHandle+0x17

ncryptsslp!SslpValidateMasterKeyHandle+0xa:
00007fff`df75b5c2 817904356c7373  cmp     dword ptr [rcx+4],73736C35h
00007fff`df75b5c9 7504    jne    ncryptsslp!SslpValidateMasterKeyHandle+0x17
```

```
ncryptsslp!SslpValidateMasterKeyHandle+0x13:
00007fff`df75b5cb 488bc1    mov     rax,rcx
00007fff`df75b5ce c3    ret
```

```
ncryptsslp!SslpValidateMasterKeyHandle+0x17:
00007fff`df75b5cf 33c0    xor     eax,eax
00007fff`df75b5d1 c3    ret
```

0:000>

The Ncrypt SSL Provider (ncryptsslp.dll)

SSL Magic	Size (x86)	Size (x64)	Validation Functions
ssl1	0xE4	0x130	SslpValidateProvHandle
ssl2	0x24	0x30	SslpValidateHashHandle
ssl3	?	?	<none>
ssl4	0x18	0x20	SslpValidate KeyPair Handle
ssl5	0x48	0x50	SslpValidate MasterKey Handle
ssl6	0x18	0x20	SslpValidate Ephemeral Handle
ssl7	?	?	<none>



ssl3 was already discussed,
appears in the following functions:

TlsGenerateSessionKeys+0x251
SPSslDecryptPacket+0x43
SPSslEncryptPacket+0x43
SPSslImportKey+0x19a
SPSslExportKey+0x76
Ssl2GenerateSessionKeys+0x22c





Pre-Master Secret (PMS)

- ❑ The 'ssl7' struct appears to be used specifically for the RSA PMS
- ❑ As advised by the RFC, it gets destroyed quickly, once the Master Secret (MS) has been derived
- ❑ Client generates random data, populates the ssl7 structure, and encrypts
- ❑ In ECC the PMS is x-coordinate of the shared secret derived (which is a point on the curve), so this doesn't /seem/ to get used in that case

Functions where ssl7 appears:

```
ncryptsslp!SPSslGenerateMasterKey+0x75  
ncryptsslp!SPSslGenerateMasterKey+0x5595  
ncryptsslp!SPSslGeneratePreMasterKey+0x15e  
ncryptsslp!TlsDecryptMasterKey+0x6b
```

Bottom line:

It's vestigial for our purposes - it doesn't do anything another secret can't



Master Secret

- ❑ Basically the Holy Grail for a given connection
 - ❑ It always exists
 - ❑ It's what gets cached and used to derive the session keys
- ❑ Structure for storage is simple - secret is unencrypted (as you'd expect)
- ❑ This + **Unique ID** = decryption, natively in tools like wireshark

So...how do we get there?

_SSL_MASTER_SECRET	
4	cbStructLength
4	dwMagic ["ssl5"]
4	dwProtocolVersion
0 / 4	dwUnknown1* [alignment?]
4 / 8	pCipherSuiteListEntry
4	bIsClientCache
48	rgbMasterSecret
4	dwUnknown2 [reserved?]



Master Secret

<u>_SSL_MASTER_SECRET</u>	
4	cbStructLength
4	dwMagic ["ssl5"]
4	dwProtocolVersion
0/4	dwUnknown1* [alignment?]
4/8	pCipherSuiteListEntry
4	bIsClientCache
48	rgbMasterSecret
4	dwUnknown2 [reserved?]

```
Command - Dump \\vmware-host\Shared Folders\Documents\Challenges\zzTmp...
0:000> .foreach(ms {s -[1]d 0 L?8000000000000000} ){.echo ***Raw Master Secret***^
***Raw Master Secret***
000000c9`86d9e980 50 00 00 00 35 6c 73 73-03 03 00 00 00 00 00 00 P...5lss.....
000000c9`86d9e990 10 1a 76 df ff 7f 00 00-00 00 00 01 7b 37 90 ...v.....{7.
000000c9`86d9e9a0 22 83 5c 78 ca 16 f7 12-53 00 39 fb 5d ea dd 03 ".\x....S.9.]...
000000c9`86d9e9b0 a7 73 8a ba c7 a5 92 67-b7 45 97 2c 01 a3 25 15 s.....g.E.,.%.
000000c9`86d9e9c0 44 0d fa d7 4c 45 c1 9a-25 a6 51 f1 00 00 00 00 D...LE..%Q....
0:000> .foreach(ms {s -[1]d 0 L?8000000000000000} ){.echo ***Parsed Master Secret***^
***Parsed Master Secret***
000000c9`86d9e980 00000050
000000c9`86d9e984 73736c35
000000c9`86d9e988 00000000`00000303
000000c9`86d9e990 00007fff`df761a10 ncryptsslp!CipherSuiteList+0x1500
000000c9`86d9e998 00000000
* Secret:
000000c9`86d9e99c 01 7b 37 90 22 83 5c 78-ca 16 f7 12 53 00 39 fb .{7.".\\x....S.9.
000000c9`86d9e9ac 5d ea dd 03 a7 73 8a ba-c7 a5 92 67 b7 45 97 2c ]....s.....g.E.,
000000c9`86d9e9bc 01 a3 25 15 44 0d fa d7-4c 45 c1 9a 25 a6 51 f1 ..%.D...LE..%Q.
*
000000c9`86d9e9cc 00000000
0:000> .foreach(ms {s -[1]d 0 L?8000000000000000} ){.echo ***Raw Master Secret***^
***Raw Master Secret***
000000c9`86d9f080 50 00 00 00 35 6c 73 73-03 03 00 00 00 00 00 00 P...5lss.....
000000c9`86d9f090 10 1a 76 df ff 7f 00 00-00 00 00 00 c3 05 82 ff ...v.....{7.
000000c9`86d9f0a0 a5 6d ea 9c 0a ed 59 42-33 69 d8 ef b1 8a 79 6c .m....YB3i....yl
000000c9`86d9f0b0 60 5e 46 1e 7b 45 d0 12-88 71 14 c0 0b 32 86 ab `^F.{E...q....2...
000000c9`86d9f0c0 4a b4 84 1a fa 12 7a f0-3f 24 6b cb 00 00 00 00 J.....z.?sk.....
<                                >
0:000> .foreach(ms {s -[1]d 0 L?8000000000000000} ){.echo ***Raw Master Secret***;db ${ms}-4 Ldwo(${ms}-4);.echo;.echo ***Parsed Master Secret***;dd ${ms}-4 L1;dc ms L1;dp ${ms}+4 L1;dps ${ms}+4+$ptrsize L1;dd ${ms}+4+2*$ptrsize L1;.echo * Secret:;db ${ms}+3*$ptrsize L30;.echo *;dd ${ms}-4+dwo(${ms}-4)-4 L1;.echo}
0:000> r @$t0 = 000000c9`86d9e980;dc @$t0 L2;dp @$t0+10 L1;.echo *;dpu poi(@$t0+10) L2
000000c9`86d9e980 00000050 73736c35
000000c9`86d9e990 00007fff`df761a10
*
00007fff`df761a10 0000c030`00000c00
00007fff`df761a18 00007fff`df762010 "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384"
0:000>
```



Master Secret Mapped to Unique Identifier

- The Master Key is linked back to a unique ID through an “**NcryptSslKey**”
- The NcryptSslKey is referenced by an “**SessionCacheItem**”
- The SessionCacheItem contains either the SessionID, or a pointer and length value for a SessionTicket
- Instantiated as either client or server item

At this point, we can find cache items, and extract the Master Secret + Unique ID

_SSL_MASTER_SECRET	
4	cbStructLength
4	dwMagic ["ssl5"]
4	dwProtocolVersion
0/4	dwUnknown1* [alignment?]
4/8	pCipherSuiteListEntry
4	bIsClientCache
48	rgbMasterSecret
4	dwUnknown2 [reserved?]

_SESSION_CACHE_CLIENT_ITEM	
4/8	pVftable
...	...
@0x10	pMasterKey
...	...
@0x88	rgbSessionID[0x20]
...	...
@0x128	pSessionTicket
@0x130	cbSessionTicketLength

_NCRYPT_SSL_KEY_HANDLE	
4	cbStructLength
4	dwMagic ["BDDD"]
4/8	pNcryptSslProvider
4/8	pNcryptSslKey

... Houston, we has plaintext.



Master Secret Mapped to Unique Identifier

```
Command - Dump \\vmware-host\Shared Folders\Documents\Challenges\_TSYS\t...
**** CLIENT CACHE ITEM ****
* SessID *
000000d4`12bfdf28 97 42 00 00 58 16 79 ae-7a 06 4f 3e 4a 35 06 82 .B..X.y.z.O>J5..
000000d4`12bfdf38 dc a9 e8 39 eb ca 07 07-5b 1a 94 4d 8b 1b 71 f7 ...9....[..M..q.
* Master Key *
000000d4`12b810fc 89 7a df 53 3d 0e 87 ea-db c4 1b cl al 3a db 24 .z.S=.....$.
000000d4`12b8110c 12 51 a5 6f 05 04 35 fa-d0 d5 4b 10 64 f8 3c 50 .Q.o..5...K.d.<P
000000d4`12b8111c ce db 9d 98 de 04 60 08-cd e0 4a 40 97 79 5d f2 .....`...J@.y.

**** CLIENT CACHE ITEM ****
* SessID *
000000d4`12bfe098 f5 35 00 00 be 2c eb cb-15 a3 8f 38 b9 9a 20 75 .5....,...8.. u
000000d4`12bfe0a8 1e d0 d5 39 57 89 01 dd-de 69 27 8d bb f9 73 8e ...9W....i'...s.
* Master Key *
000000d4`12b811fc 71 6a 1d 49 36 56 bf 53-4e 43 6f fb 58 ff 2e 40 qj.I6V.SNCo.X..@
000000d4`12b8120c 00 05 16 b7 35 db d5 df-af f9 3f 37 b5 ac 90 ba ....5....?7....
000000d4`12b8121c 1c 3a 25 ba 3e 15 05 b8-f3 aa 16 8a 65 7e 00 7b .:%,>.....e~.{

**** CLIENT CACHE ITEM ****
* Sess Ticket *
000000d4`12bda9f0 f8 0a ee e7 9a 1a 2f 64-1f e7 9d b6 da af 8a el ...../d.....
* Master Key *
000000d4`12b808fc 4c e4 8c 51 7e 22 7c 5e-ac 47 ad 6d 78 13 2b 17 L..Q~|^^.G.mx.+
000000d4`12b8090c db 7d 4c 12 cf 98 30 0f-ad 21 ab b0 17 d7 85 bb .)L..0.!.....
000000d4`12b8091c 39 ae e0 09 88 72 cb 3c-9f 0d 4d d6 20 f2 36 e4 9....r.<.M. .6.

**** CLIENT CACHE ITEM ****
* SessID *
000000d4`12bfe378 bc b3 af f3 58 1f cc b9-fe 26 8d 46 f9 9f 5e 2c ....X....&F..^,
000000d4`12bfe388 6c c9 e5 9e 51 c6 71 4d-70 99 7e 63 b9 c6 fe 73 1...Q.qMp.^c...s
* Master Key *
000000d4`12b5607c e4 5e 18 94 51 97 c2 f0-a2 ad db 90 1a 95 58 f1 .^..Q.....X.
000000d4`12b5608c 94 24 1d 2b 48 8c dc 3d-1f 81 el 27 1a cb 4d c7 $.+H..=...'.M.
000000d4`12b5609c 76 e3 c7 72 17 7c 7d 04-62 af ec a5 7a 3d 9c b2 v...r.|).b...z=..

**** CLIENT CACHE ITEM ****
* SessID *
000000d4`12bfe4e8 c7 d0 f9 52 fb 3f c4 99-9a 69 2c e3 67 4a cb 1a ...R.?...i.,gJ..
000000d4`12bfe4f8 4b 2c 79 1e ce 2c 6d 16-21 af 95 e6 41 4e c3 b0 K,y...,m.!...AN..
* Master Key *
000000d4`12b3c27c db 93 02 6b 71 e0 32 3b-60 e2 53 7f 0e ee bf 4f ...kq.2;`S....0
000000d4`12b3c28c c3 21 09 4b 8a 9a 6c cd-8c f0 f5 0c 7f a6 8c 29 !.K..l....)
000000d4`12b3c29c 4f 6c 49 0d 5a f3 df 88-1d b5 85 e2 a1 0a 0a ea OIIZ.....
<   ,>
0:000> .foreach(sessItem {s -[1w]q 0 L2800000000000000 schannel!
CSessionCacheClientItem:: vftable')(.echo **** CLIENT CACHE ITEM ****;.if
(dwo($sessItem)+88) > 0){.echo * SessID *;db $sessItem+88 L20} .else{.echo *
Sess Ticket *; db poi($sessItem+128) L10};.echo * Master Key *;db
poi(poi($sessItem+10)+10)+lc L30;.echo
```



Wireshark SSL Log Format

RSA Session-

ID: 97420000581679ae7a064f3e4a350682dca9e839ebca0
7075b1a944d8b1b71f7 **Master-**

Key: 897adf533d0e87eadbc41bclal3adb241251a56f0504
35fad0d54b1064f83c50cedb9d98de046008cde04a409779
5df2

RSA Session-

ID: f5350000be2cebc15a38f38b99a20751ed0d53957890
1ddde69278dbbf9738e **Master-**

Key: 716a1d493656bf534e436ffb58ff2e40000516b735db
d5dfaff93f37b5ac90ba1c3a25ba3e1505b8f3aa168a657e
007b

RSA Session-

ID: bcb3aff3581fccb9fe268d46f99f5e2c6cc9e59e51c67
14d70997e63b9c6fe73 **Master-**

Key: e45e18945197c2f0a2addb901a9558f194241d2b488c
dc3d1f81e1271acb4dc776e3c772177c7d0462afeca57a3d
9cb2

RSA Session-

ID: c7d0f952fb3fc4999a692ce3674acb1a4b2c791ece2c6
d1621af95e6414ec3b0 **Master-**

Key: db93026b71e0323b60e2537f0eeebf4fc321094b8a9a
6cccd8cf0f50c7fa68c294f6c490d5af3df881db585e2a10a
0aea

Ephemeral & Persistent Private Keys

- Both share the same structure
- Both store secrets in a Key Storage Provider Key struct (KPSK)
- The “Key Type” is compared with different values
 - ssl6 gets compared with a list stored in bcryptprimitives
 - ssl4 gets compared with a list stored in NCRYPTPROV
- The Key Storage Provider Key (KPSK) is referenced indirectly through an “Ncrypt Key” struct*

_SSL_KEY_PAIR	
4	cbStructLength
4	dwMagic ["ssl4" "ssl6"]
4	dwKeyType
4	dwUnknown1 [alignment?]
4/8	pKspProvider
4/8	pKspKey

_KSP_KEY	
4	cbStructLength
4	dwMagic ["KSPK"]
4	dwKeyType
...	...
@0x60	pMSKY
@0xD0	pDpapiBlob
@0xD8	dwDpapiBlobLength

_NCRYPT_KEY_HANDLE	
4	cbStructLength
4	dwMagic [0x44440002]
4	dwKeyType
4	dwUnknown1 [alignment?]
4/8	pKspProvider
4/8	pKspKey

🔑 Ephemeral Private Key

- ❑ For performance, reused across connections
 - ❑ Given the public connection params, we can derive the PMS and subsequently MS
- ❑ Stored unencrypted in a LE byte array
 - ❑ Inside of MSKY struct
- ❑ The curve parameters are stored in the KPSK
 - ❑ Other parameters (A&B, etc) are stored in MSKY w/ the key
- ❑ Verified by generating the Public & comparing
 - ❑ The Public Key is also stored in the first pointer of the **CEphemData** struct that points to “**ss16**”

```

Command - Dump \\vmware-host\Shared Folders\Documents\Challenges\zzTm...
0:000> .foreach(ek {s -[lw]q 0 L?8000000000000 schannel!CEphemKeyData::`vftable'}){.echo *****
000000c9`86d2a000 00007fff`ed23b140 schannel!CEphemKeyData::`vftable'
* Public *
000000c9`86d9e688 f2 55 0c 8b e6 02 51 3b-98 e5 8a 7c 4a 8f 4b 1a .U....Q;...|J.K.
000000c9`86d9e698 f8 b8 fd bd 6d d1 8c 85-78 f6 51 2a 8e 7a 5a 62 ...m....x.Q*.zzB
000000c9`86d9e6a8 fd 34 cc 5b 75 c4 bf ca-3c 6b 0e b8 84 b7 32 35 .4.[u....<k....25
000000c9`86d9e6b8 da 94 bb 1d 0c cc d0 00-8d 8b b7 2e 7a a4 79 27 .....z.y'
* Curve *
000000c9`86d22fb0 000000c9`86d5fa00 "nistP256"
** G(x)
000000c9`8698e71c 6b 17 d1 f2 e1 2c 42 47-f8 bc e6 e5 63 a4 40 f2 k....,BG....c.@.
000000c9`8698e72c 77 03 7d 81 2d eb 33 a0-f4 a1 39 45 d8 98 c2 96 w.).-3...9E....
** G(y)
000000c9`8698e71c 6b 17 d1 f2 e1 2c 42 47-f8 bc e6 e5 63 a4 40 f2 k....,BG....c.@.
000000c9`8698e72c 77 03 7d 81 2d eb 33 a0-f4 a1 39 45 d8 98 c2 96 w.).-3...9E....
* Private [LE!] *
000000c9`86d5fc80 d0 27 df dd c2 c6 78 5e-be 78 7e 70 94 c2 b7 1a .'....x~p....
000000c9`86d5fc90 b9 90 50 67 72 0d ce f7-63 45 e4 da 90 0c 59 78 ..Pgr....cE....Yx

*****
000000c9`86d2cee0 00007fff`ed23b140 schannel!CEphemKeyData::`vftable'
* Public *
000000c9`86d9ff88 d6 35 6b cc 81 66 c0 6e-74 86 f8 36 8c be c7 42 .5..f.nt..6...B
000000c9`86d9ff98 00 3e 6b 34 49 7f 95 1a-d5 e3 1f 6e 72 75 aa 77 .>k4I.....nru.w
000000c9`86d9ffa8 96 6d 93 0d 13 40 d3 f0-cd f6 ac 30 16 01 7a 44 .m....@....0.zD
000000c9`86d9ffb8 2b ca 16 ea 9d fa 00 5d-63 f2 08 77 3d 23 1a 5d +.....]c..w=#[]
* Curve *
000000c9`86d231f0 000000c9`86d60540 "nistP256"
** G(x)
000000c9`8698e93c 6b 17 d1 f2 e1 2c 42 47-f8 bc e6 e5 63 a4 40 f2 k....,BG....c.@.
000000c9`8698e94c 77 03 7d 81 2d eb 33 a0-f4 a1 39 45 d8 98 c2 96 w.).-3...9E....
** G(y)
000000c9`8698e93c 6b 17 d1 f2 e1 2c 42 47-f8 bc e6 e5 63 a4 40 f2 k....,BG....c.@.
000000c9`8698e94c 77 03 7d 81 2d eb 33 a0-f4 a1 39 45 d8 98 c2 96 w.).-3...9E....
* Private [LE!] *
000000c9`86d602c0 cd 94 db f9 f0 68 b9 6e-05 8f b6 7c fa d0 8c 9e .....h.n....|.....
000000c9`86d602d0 da 78 97 2c 01 24 42 32-0b 8f 35 b5 4a 4f 7d 9e .x.,.$B2..5.JO}.

<                                >
0:000> .foreach(ek {s -[lw]q 0 L?8000000000000 schannel!CEphemKeyData::`vftable'})
{.echo *****
;dps ek L1;.echo * Public *;db poi(${ek}+10)+8 L40;r @$t0 =
poi(poi(poi(${ek}+8)+10)+10);.echo * Curve *; dpu @$t0+1f0 L1;.echo **
G(x);db poi(@$t0+1f8)+7c L20;.echo ** G(y);db poi(@$t0+1f8)+7c L20;.echo *
Private [LE!] *;db poi(poi(poi(@$t0+60)+10)+18) L20;.echo}

```

🔑 “Persistent” Private Key

- The RSA Key that is stored on disk
 - Unique instance for each private RSA Key – by default, the system has several
 - E.g. one for Terminal Services
- RSA Keys are DPAPI protected
 - Lots of research about protection / exporting
 - Note the MK GUID highlighted from the Blob
- The Key is linked to a given Server Cache Item
- Verified by comparing the DPAPI blob in memory to protected certificate on disk
 - Also verified through decryption

```
Command - Dump \\vmware-host\Shared Folders\Documents\Challenges\zzTmp\...
0:000> .foreach(key {s -[wl]q 0x0 L?800000000000 schannel!CSessionCacheServerItem::`vftable'}){r @$t0 = poi(poi(poi(poi(poi($key)+F0)+48))+8)+10)+18+10};.echo *** Private Key ***;db poi(@$t0+D0) Ldwo(@$t0+D8);.echo]
*** Private Key ***
000000c9`85d06630 01 00 00 00 d0 8c 9d df-01 15 d1 11 8c 7a 00 c0 .....z..
000000c9`85d06640 4f c2 97 eb 01 00 00 00 66 68 6a f9 d8 1b d1 4a 0.....fhj...J
000000c9`85d06650 85 fc 1a 77 28 7d 5c d1-04 00 00 00 2c 00 00 00 ...w()\...
000000c9`85d06660 43 00 72 00 79 00 70 00-74 00 6f 00 41 00 50 00 C.r.y.p.t.o.A.P.
000000c9`85d06670 49 00 20 00 50 00 72 00-69 00 76 00 61 00 74 00 I..P.r.i.v.a.t.
000000c9`85d06680 65 00 20 00 4b 00 65 00-79 00 00 00 10 66 00 00 e..K.e.y.f..
000000c9`85d06690 00 01 00 00 20 00 00 00-8a 60 40 b8 f7 4f ec f9 .....`0..0.
000000c9`85d066a0 37 6f cc 0b 14 82 e6 3f-40 79 65 5f 94 51 a3 75 7o..?ye_Q.u
000000c9`85d066b0 5a da e5 6f 81 89 ff d4-00 00 00 0e 80 00 00 Z.o.....
000000c9`85d066c0 00 02 00 00 20 00 00 00-d2 41 1d a7 8b f7 ce b4 ....A....
000000c9`85d066d0 51 a6 85 13 39 0d da f1-00 54 ce e7 04 a8 e0 17 Q...9....T....
000000c9`85d066e0 a7 9d c6 98 df 6f ef a3-50 05 00 00 2b 9f 70 ce .....o..P...+p.
000000c9`85d066f0 0c 3f fb f1 3f a6 78 87-0c 47 d9 b0 60 33 5d 27 ?.?..x..G..`3]'.
000000c9`85d06700 82 af 5d eb b7 21 b2 36-2a 58 a2 88 56 61 69 8c ..]..!..6*X..Vai.
000000c9`85d06710 3e 11 20 ff 27 24 b5 dc-e9 b2 fd 3d b0 c9 5e 31 >..'$....=..^1
000000c9`85d06720 e6 56 5e de 81 a9 78 ea-ea 16 c7 52 a4 70 9b 34 .V^..x...R.p.4
000000c9`85d06730 7c 6c b8 9a 86 fb 02 d7-e5 a5 c2 e3 be 2e c7 65 |l.....e
000000c9`85d06740 21 f1 99 0a 5b 0d 34 98-ad 10 af 45 b7 79 f5 3e !...[.4...E.y.>
000000c9`85d06750 8a 95 be 29 83 be 68 74-78 64 d1 b3 db 13 2d 10 ...)htxd.....
000000c9`85d06760 42 d0 95 f5 02 2d d4 9a-97 87 00 b1 6e 76 d0 7c B....-....nv.|_
000000c9`85d06770 e1 67 d1 90 94 ea b0 9e-a7 bd 37 12 2f 48 76 56 .g.....7/HvV
000000c9`85d06780 25 94 e9 cf 28 f6 ae 6e-dc ba f3 77 0b b2 ce 26 %...(.n..w...&
000000c9`85d06790 fa 33 32 0b b9 13 48 9a-77 0f b7 47 29 92 da c7 .32..H.w..G)...
000000c9`85d067a0 7a 21 aa 12 04 8c 0b 27-6e fd 24 48 ab 91 8c 98 z!....'n.$H...
000000c9`85d067b0 3d 68 7c 0b 48 91 58 f7-6e a2 85 d8 a9 ec 2a ac =h|.H.X.n....*.
000000c9`85d067c0 9d b3 39 e5 51 24 e1 d9-41 eb 51 64 12 8b 2a 65 ..9.Q$.A.Qd..*e
000000c9`85d067d0 62 4e ce e4 83 b1 e9 a7-0a a1 46 d5 46 fe 4b c3 bN.....F.F.K.
000000c9`85d067e0 f2 8e fa d9 28 b9 38 86-1a 84 95 58 93 db 2d 40 ....(.8....X..@_
000000c9`85d067f0 5f 4b 47 bc 95 51 ce bc-b3 a2 bd 12 47 37 18 68 _KG..Q....G7.h
000000c9`85d06800 fe c6 f9 55 9a 28 61 c5-c8 8a 55 07 04 ef 3a 2a ....U.(....U..*:
000000c9`85d06810 3b d2 b8 e8 26 09 6f c1-a4 ab 75 fd 82 93 35 a6 ;...&.o..u...5.
000000c9`85d06820 00 aa 92 14 9c 77 10 af-b9 05 93 af 3a 47 6d d2 .....w.....:Gm.
000000c9`85d06830 a3 b3 d8 cf 98 f2 72 e4-95 9e 07 ed 4d 7d 28 2e .....r.....M)(.
000000c9`85d06840 fe c8 d0 bd 42 75 26 fb-e9 94 0c ea af 03 4d cd ....Bu&....M.
000000c9`85d06850 e1 3e 98 07 4e 3c 87 53-80 76 93 c6 bd 15 c3 47 .>..N<.S.v....G
000000c9`85d06860 0c aa af 20 88 11 84 15-0b 71 64 32 35 fd a7 2d .....qd25..-
000000c9`85d06870 5d 00 3a 5a dd 16 39 8f-be b0 e2 64 88 2c 12 84 ].:Z..9...d.,..
000000c9`85d06880 3d c7 b1 cd 67 08 bf d7-6e a1 f8 a0 58 f3 21 fc =...g..n....X.!.
000000c9`85d06890 ef 40 4e a0 e3 92 b9 77-fe 7f fc a8 b8 ba 47 b2 .@N..w.....G.
000000c9`85d068a0 be 0f 48 65 d1 22 a2 e2-c8 e6 92 11 ba d3 0a a8 ..He.".....
0:000> .foreach(key {s -[wl]q 0x0 L?800000000000 schannel!CSessionCacheServerItem::`vftable'} ){r @$t0 = poi(poi(poi(poi(poi($key)+F0)+48))+8)+10)+18+10};.echo *** Private Key ***;db poi(@$t0+D0) Ldwo(@$t0+D8);.echo]
```

🔑 Decrypting Persistent Key - DPAPI

- ❑ Can extract the blob from memory and decrypt w/ keys from disk
 - ❑ DPAPIck / Mimikatz

OR

- ❑ Can decrypt directly from memory :D
 - ❑ MasterKeys get cached in Memory
 - ❑ On Win10 in: dpapisrv!g_MasterKeyCacheList
 - ❑ See Mimilib for further details
 - ❑ Even though symbols are sort of required, we could likely do without them
 - ❑ There are only two Bcrypt key pointers in lsasrv's .rdata section (plus one lock)
 - ❑ Identifying the IV is more challenging

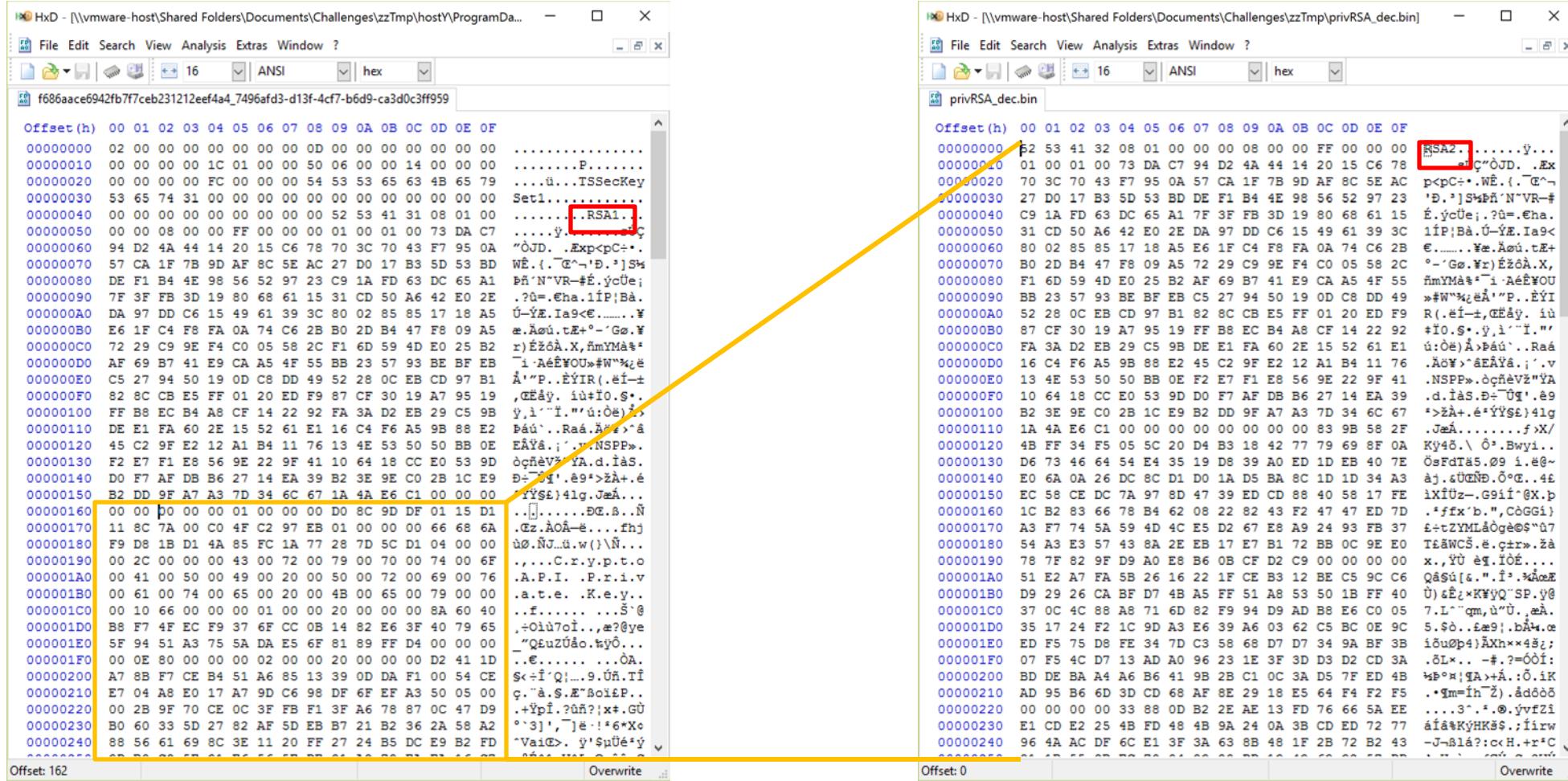
```
Command - Dump \\vmware-host\Shared Folders\Documents\Challenges\zzT...
0:000> db lsasrv!InitializationVector L8;.echo
00007fff`edaec528 60 4a dd 90 31 45 80 97 `J..1E...
0:000> r @$t0 = poi(poi(lsasrv!h3DesKey)+10); db @$t0+3c Ldwo(@$t0+38);.echo
000000c9`85fb005c 3a 6a dd 76 0f 30 0b 7c-3d 43 f4 db e4 12 66 3a :j.v.0.|=C....f:
000000c9`85fb006c 8c f2 b1 f0 a0 a5 5c e4 .....`.
0:000> !list -x ".if(dwo(@$extret+18) == f96a6866){dq @$extret L2;dq @$extret+10 L1;
000000c9`85d6a3b0 000000c9`85d5cd10 000000c9`85dbbd0
000000c9`85d6a3c0 00000000`000003e7
000000c9`85d6a3c8 f96a6866 4ad11bd8 771afc85 d15c7d28
000000c9`85d6a3d8 01dcce0`db1e2d0c
000000c9`85d6a3e0 00000040
000000c9`85d6a3e4 e5 ce 98 a2 ff 50 0d 78-4a 8e 17 54 8f f1 fe b7 .....P.xJ..T....
000000c9`85d6a3f4 21 7e 17 70 2f 7d 52 aa-db cd 75 93 25 48 7a c2 !~.p/}R...u.%Hz.
000000c9`85d6a404 70 62 1a d9 5a e8 97 19-64 c6 62 a0 2a 75 66 9e pb...Z..d.b.*uf.
000000c9`85d6a414 a5 c8 33 0c d7 af b6 b1-fb 38 ab 89 b7 d5 cf 14 ..3.....8.....
000000c9`85d6a424 00000000
000000c9`85d6a430 "C:\Windows\system32\Microsoft\Protect\S-1-5-18\"
```



```
0:000> !list -x ".if(dwo(@$extret+18) == f96a6866){dq @$extret L2;dq @$extret+10
L1;dd @$extret+18 L10/4;dq @$extret+28 L1;dd @$extret+30 L1;db @$extret+34
Ldwo(@$extret+30);dd @$extret+74 L1;du /c ff poi(@$extret+78)}" dpapisrv!
g_MasterKeyCacheList|
```

Cached DPAPI MK + Params to Decrypt

🔑 Decrypting Persistent Key - DPAPI



Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
 00000000 02 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 00000010 00 00 00 00 1C 01 00 00 50 06 00 00 14 00 00 00P.....
 00000020 00 00 00 00 FC 00 00 00 54 53 53 65 63 4B 65 79ü...TSSecKey
 00000030 53 65 74 31 00 00 00 00 00 00 00 00 00 00 00 00 Set1.....
 00000040 00 00 00 00 00 00 00 00 52 53 41 31 08 01 00RSA1..
 00000050 00 00 08 00 00 FF 00 00 00 01 00 01 00 73 DA C7ý.....
 00000060 94 D2 4A 44 14 20 15 C6 78 70 3C 70 43 F7 95 0A "ÖJD..Exp<PC+*
 00000070 57 CA 1F 7B 9D AF 8C 5E AC 27 D0 17 B3 5D 53 BD WÉ.("G~'D."]S4
 00000080 DE F1 B4 4E 98 56 52 97 23 C9 1A FD 63 DC 65 A1 Bñ'VR#É.yÜe;
 00000090 7F 3F FB 3D 19 80 68 61 15 31 CD 50 A6 42 E0 2E .?Ü=.Eha.1ÍP;Bá.
 000000A0 DA 97 DD C6 15 49 61 39 3C 80 02 85 85 17 18 A5 Ú-Ý.E.Ia9€.....Y
 000000B0 E6 1F C4 F8 FA 0A 74 C6 2B B0 2D B4 47 F8 09 A5 æ.Äöú.tZ+°-'Gö.Y
 000000C0 72 29 C9 FA F4 C0 05 58 2C F1 6D 59 4D E0 25 B2 r)ÉzöÁ.X,ñmYMå%
 000000D0 AF 69 B7 41 E9 CA A5 4F 55 BB 23 57 93 BE BF EB "i.AéÉWOU»#W"çé
 000000E0 C5 27 94 50 19 OD 08 C8 49 52 28 OC EB CD 97 B1 Á"\"P..ÉYIR.(éí-
 000000F0 82 8C CB E5 FF 01 20 ED F9 87 CF 30 19 A7 95 19 ,GÉáy. iù+íO.S*.
 00000100 FF B8 EC B4 A8 CF 14 22 92 FA 3A D2 EB 29 C5 9B y,i"í."ú:ðe);
 00000110 DE E1 FA 60 2E 15 52 61 E1 16 C4 F6 A5 9B 88 E2 Béú..Raá.äí>"á
 00000120 45 C2 9F E2 12 A1 B4 11 76 13 4E 53 50 50 BB 0E EÄYÁ.;..NSPP».
 00000130 F2 E7 F1 E8 56 9E 22 9F 41 10 64 18 CC E0 53 9D öçñév"YA.d.iás.
 00000140 D0 F7 AF DB 27 14 EA 39 B2 3E 9C 00 2C 1E E9 D+öí..äg>žá+.é
 00000150 B2 DD F9 A7 A3 7D 34 6C 67 1A 4E E6 C1 00 00 00 YÝSE)4lg.Jem...
 00000160 00 00 00 00 01 00 00 00 D0 8C 9D DF 01 15 D1DE.B...N
 00000170 11 8C 7A 00 C0 4F C2 97 EB 01 00 00 00 66 68 6A .Gz.AOA-é...fhj
 00000180 F9 68 1B D1 4A 85 FC 1A 77 28 7D 5C D1 04 00 00 ü.ÑJ.ü.w()\\Ñ...
 00000190 00 2C 00 00 00 43 00 72 00 79 00 70 00 74 00 6FC.r.y.p.t.o
 000001A0 00 41 00 50 00 49 00 20 00 50 00 72 00 69 00 76 .A.P.I. .P.r.i.v
 000001B0 00 61 00 74 00 65 00 20 00 4B 00 65 00 79 00 00 .a.t.e. .K.e.y..
 000001C0 00 10 66 00 00 01 00 20 00 00 00 8A 60 40 ..f..... .S`@
 000001D0 B8 F7 4F EC F9 37 6F CC OB 14 82 E6 3F 40 79 65 .+ö1i7o!...æ@ye
 000001E0 5F 94 51 A3 75 5A DA E5 6F 81 89 FF D4 00 00 _"QéuZÚáo.wý...
 000001F0 00 0E 80 00 00 02 00 00 00 00 D2 41 1D ..€..... ...ðA.
 00000200 A7 8B F7 CE B4 51 A6 85 13 39 0D DA F1 00 54 CE \$<+í Q!...9.ÜH.Tí
 00000210 E7 04 A8 E0 17 A7 9D C6 98 DF 6F EF A3 50 05 00 ç."à.S.E"BoiFp..
 00000220 00 2B 9F 70 CE 0C 3F FB F1 3F A6 78 87 0C 47 D9 .+ÝpI.?ñ?;|x#.GU
 00000230 B0 60 33 5D 27 82 AF 5D EB B7 21 B2 36 2A 58 A2 ^`3',_jé!^*Xc
 00000240 88 56 61 69 8C 3E 11 20 FF 27 24 B5 DC E9 B2 FD ^ValD>. y'Süéü'y
 00000250 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 Offset: 162 Overwrite ..

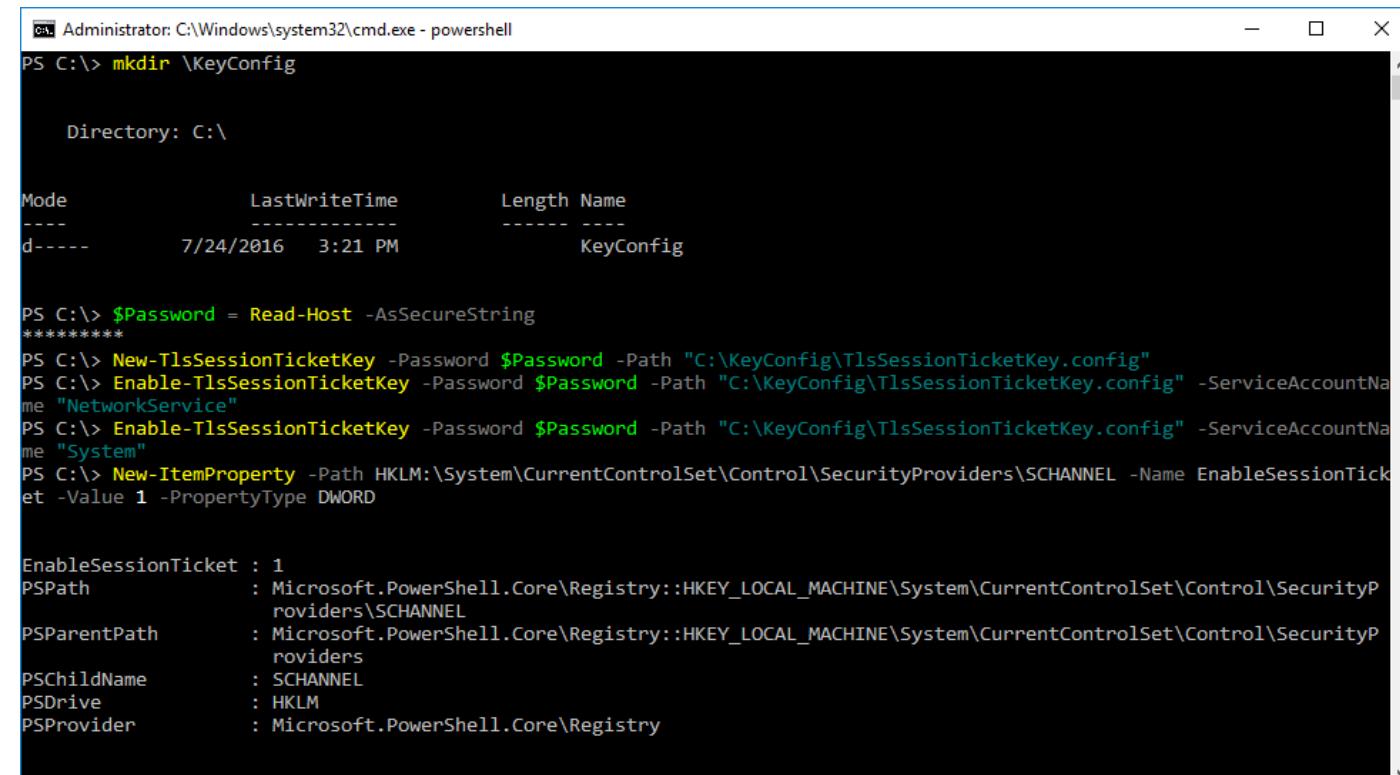
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
 00000000 F2 53 41 32 08 01 00 00 00 08 00 00 FF 00 00 00 RSA2.....ý...
 00000010 01 00 01 00 73 DA C7 94 D2 4A 44 14 20 15 C6 78 .æ"ÖJD. .Ex
 00000020 70 3C 70 43 F7 95 0A 57 CA 1F 7B 9D AF 8C 5E AC p<pC+..WÉ.("G~
 00000030 27 D0 17 B3 5D 53 BD DE F1 B4 4E 98 56 52 97 23 D."]SþBñ'VR#
 00000040 C9 1A FD 63 DC 65 A1 7F 3F FB 3D 19 80 68 61 15 É.yÜe.;?Ü=.Eha.
 00000050 31 CD 50 A6 42 E0 2E 97 DD C6 15 49 61 39 3C 1í;Bá.Ú-Ý.E.Ia9<
 00000060 80 02 85 85 17 18 A5 E6 1F C4 F8 FA 0A 74 C6 2B €.....Wá.Äöú.tZ+
 00000070 B0 2D B4 47 F8 09 A5 72 29 C9 9E F4 C0 05 58 2C "Gö.Wr)Ézó.A.X,
 00000080 F1 6D 59 4D E0 25 B2 AF 69 B7 41 E9 CA A5 4F 55 fmYMå%"-i.AéÉWOU
 00000090 BB 23 57 93 BE BF EB C5 27 94 50 19 0D C8 DD 49 »#W"çé"\"P..ÉYI
 000000A0 52 28 0C EB CD 97 B1 82 8C CB E5 FF 01 20 ED F9 R(.éí-,GÉáy. iù
 000000B0 87 CF 30 19 A7 95 19 FF B8 EC B4 A8 CF 14 22 92 +íO.S..ý,i"í."
 000000C0 FA 3A D2 EB 29 C5 9B DE E1 FA 60 2E 15 52 61 E1 ú:ðe)Á.äéú..Raá
 000000D0 16 C4 F6 A5 9B 68 E2 45 C2 9F E2 12 A1 B4 11 76 .ÄöY"äEÄYá.;.v
 000000E0 13 4E 53 50 50 BB 0E F2 E7 F1 E8 56 9E 22 9F 41 .NSPP».öçñéVZ"Y
 000000F0 10 64 18 CC E0 53 9D D0 F7 AF DB B6 27 14 EA 39 .d.iás.D+Üq'.ë9
 00000100 B2 3E 9E CO 2B 1C E9 B2 DD 9F A7 3D 74 6C 67 "žá+.é.äYSE)4lg
 00000110 1A 4E E6 C1 00 00 00 00 00 00 00 83 9B 58 2F .Jéá.....fX/
 00000120 4B FF 34 F5 05 5C 20 D4 B3 18 42 77 79 69 8F OA Kýéð.\\ ð.Bwy!.
 00000130 D6 73 46 64 54 E4 35 19 D8 39 A0 ED 1D EB 40 7E ðsfðtä5.ð9 i.ðð~
 00000140 E0 6A 0A 26 DC 6C D1 D0 1A D5 B8 8C 1D 1D 34 A3 áj.äÜñð.ð.ç.4ð
 00000150 EC 58 CE DC TA 97 8D 47 39 ED CD 88 40 58 17 FE ixñúz-.Ggíñ.ð.X.b
 00000160 1C B2 83 66 78 B4 62 08 22 82 43 F2 47 47 ED 7D .ffx'b.",CòGGi
 00000170 A3 F7 74 5A 59 4D 4C E5 D2 67 E8 A9 24 93 FB 37 f+ZYMå.ögë@S"ñ7
 00000180 54 A3 E3 57 43 8A 2E EB 17 E1 72 BB 0C 9E E0 TéäWCŠ.é.çír>,žá
 00000190 78 7F 82 9F D9 A0 E8 B6 OB CF D2 C9 00 00 00 x.,ý.éq.íðé...
 000001A0 51 E2 A7 FA 5B 26 16 22 1F CE B3 12 BE C5 9C C6 QäSá.í.í.äæE
 000001B0 D9 29 26 CA BF D7 4B A5 F1 8A 53 50 1B FF 40 Ü).éž.äKýQ'SP.yð
 000001C0 37 0C 4C 88 A8 71 6D 82 F9 94 D9 AD B8 E6 C0 05 7.L."qm,ú.ü..äá.
 000001D0 35 17 24 F2 1C 9D A3 E6 39 A6 03 62 C5 BC 0E 9C 5.ð.äé9;.bÅ.øe
 000001E0 ED F5 75 D8 FE 34 7D C3 58 68 D7 34 9A BF 3B iðuðp4)ÄXhxx4ð;:
 000001F0 07 F5 4C D7 13 AD A0 96 23 1E 3F 3D 32 CD 3A .ðL..-#.?ðði:
 00000200 BD DE BA A4 A6 B6 41 9B 2B C1 0C 3A D5 7F ED 4B 4þþr;|QA+ð.ð.ik
 00000210 AD 95 B6 6D 3D CD 68 AF 8E 29 18 E5 64 F4 F2 F5 .*qm=íñ"ž).äðððð
 00000220 00 00 00 33 88 D2 B2 AE 13 FD 76 66 5A EE ...3.º.ø.yvZí
 00000230 E1 CD E2 25 4B FD 48 4B 9A 24 0A 3B CD ED 72 77 áíáKýHkš.;íirw
 00000240 96 4A AC DF 6C E1 3F 3A 63 8B 48 1F 2B 72 B2 43 -J.áslá?;cxH.+rçC
 00000250 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 Offset: 0 Overwrite ..





Session Tickets

- ❑ Not seemingly in widespread use with IIS?
 - ❑ Comes around w/ Server 2012 R2
 - ❑ Documentation is lacking.
- ❑ Enabled via reg key + powershell cmdlets?
 - ❑ Creates an “Administrator managed” session ticket key
- ❑ Schannel functions related to Session Tickets load the keyfile from disk
- ❑ Export-TlsSessionTicketKey :D



```
Administrator: C:\Windows\system32\cmd.exe - powershell
PS C:\> mkdir \KeyConfig
Directory: C:\

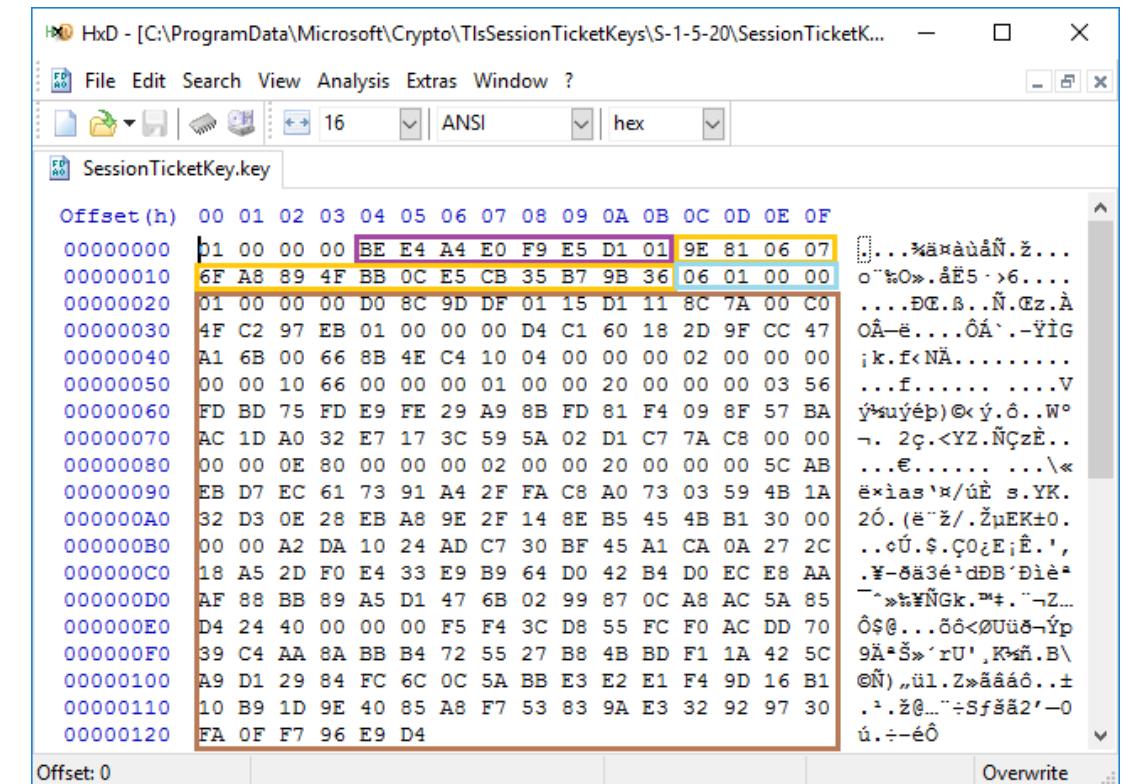
Mode          LastWriteTime      Length Name
----          -----          ----  --
d---  7/24/2016   3:21 PM           KeyConfig

PS C:\> $Password = Read-Host -AsSecureString
*****
PS C:\> New-TlsSessionTicketKey -Password $Password -Path "C:\KeyConfig\TlsSessionTicketKey.config"
PS C:\> Enable-TlsSessionTicketKey -Password $Password -Path "C:\KeyConfig\TlsSessionTicketKey.config" -ServiceAccountName "NetworkService"
PS C:\> Enable-TlsSessionTicketKey -Password $Password -Path "C:\KeyConfig\TlsSessionTicketKey.config" -ServiceAccountName "System"
PS C:\> New-ItemProperty -Path HKLM:\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL -Name EnableSessionTicket -Value 1 -PropertyType DWORD

EnableSessionTicket : 1
PSPath              : Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\SecurityProviders\SCHANNEL
PSParentPath        : Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\SecurityProviders
PSChildName         : SCHANNEL
PSDrive             : HKLM
PSProvider          : Microsoft.PowerShell.Core\Registry
```

Session Ticket Key

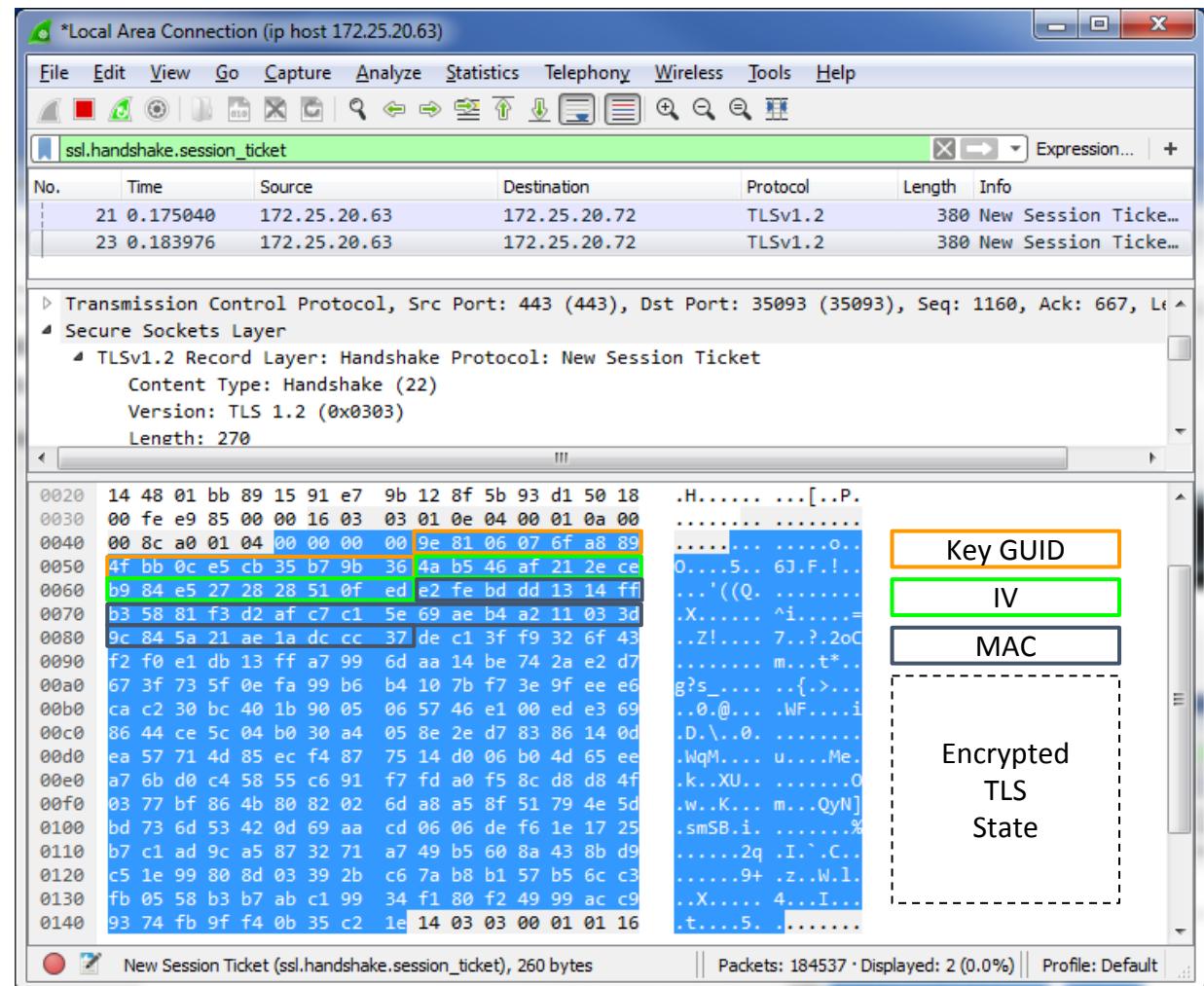
- Keyfile contains a DPAPI blob, preceded by a SessionTicketKey GUID + 8 byte value
- Key gets loaded via schannel
 - The heavy lifting (at least in Win10) is done via mskeyprotect
- AES key derived from decrypted blob via BCryptKeyDerivation()
- Key gets cached inside mskeyprotect!
 - No symbols for cache : /
 - No bother, we can just find the Key GUID that's cached with it :D



- Possibly Salt or MAC?
- Session Ticket Key GUID
- Size of ensuing DPAPI Blob
- DPAPI Blob (contains it's own fields)

Decrypting Session Tickets

- ❑ Session Ticket structure pretty much follows the RFC (5077), except:
 - ❑ MAC & Encrypted State are flipped (makes a lot of sense)
- ❑ After extracting/deriving the Symm key, it's just straight AES 256
- ❑ Contents of the State are what you'd expect:
 - ❑ Timestamp
 - ❑ Protocol/Ciphersuite info
 - ❑ MS struct



Decrypting Session Tickets

The screenshot shows a Wireshark capture window titled "ssl.handshake.session_ticket". The packet list pane displays two TLSv1.2 frames, both labeled "New Session Ticket, Change Cipher Spec, Hello Request". The details pane shows the TLSv1.2 Record Layer: Handshake Protocol: New Session Ticket content, including Content Type (Handshake), Version (TLS 1.2), and Length (270). The bytes pane shows the raw hex and ASCII data of the captured session ticket. The bottom status bar indicates "New Session Ticket (ssl.handshake.session_ticket), 260 bytes".

Wireshark interface details:

- File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
- ssl.handshake.session_ticket
- No. Time Source Destination Protocol Length Info
- 21 0.175040 172.25.20.63 172.25.20.72 TLSv1.2 380 New Session Ticket, Change Cipher Spec, Hello Req...
- 23 0.183976 172.25.20.63 172.25.20.72 TLSv1.2 380 New Session Ticket, Change Cipher Spec, Hello Req...

Selected packet details:

- Frame 23: 380 bytes on wire (3040 bits), 380 bytes captured (3040 bits) on interface 0
- Ethernet II, Src: VMware_df:02:a2 (00:0c:29:df:02:a2), Dst: Dell_10:44:a0 (f0:1f:af:10:44:a0)
- Internet Protocol Version 4, Src: 172.25.20.63, Dst: 172.25.20.72
- Transmission Control Protocol, Src Port: 443 (443), Dst Port: 35093 (35093), Seq: 1160, Ack: 667, Len: 326
- Secure Sockets Layer
 - TLSv1.2 Record Layer: Handshake Protocol: New Session Ticket
 - Content Type: Handshake (22)
 - Version: TLS 1.2 (0x0303)
 - Length: 270

Decrypted text pane shows the session ticket content as a dump file:

```
Command - Dump C:\Users\jkambic\Downloads\_DEFCON-BH\sessionTicket_for_decrypt.lsass.DMP
```

Hex dump of the session ticket:

00000000	00 00 00 00
00000010	a0 00 00 00
00000020	00 00 00 00
00000030	80 00 00 00
00000040	00 04 00 00
00000050	22 b2 b5 00
00000060	00 00 00 00
00000070	30 c0 00 00
00000080	66 e7 47 00
00000090	e9 d0 a2 00
00000091	08 3d 9e 00

Initialization vector: 4ab546af212cecb984e52

Decrypted text:

[Download as a binary file]

Master Secret





Secrets are cool and all...

But Jake, what if I don't have a packet capture?

(And I don't care about future connections?)



The Context

Inherent Metadata TLS Provides

Core SSL/TLS functionality

- ❑ Timestamps
 - ❑ The random values *typically* start with a 4-byte timestamp (if you play by the RFCs)
- ❑ Identity / fingerprinting
 - ❑ Public Key
 - ❑ Session ID*
 - ❑ Offered Cipher Suites / Extensions
- ❑ Session ID's are arbitrary, but are not always random -> Schannel is a perfect example
 - ❑ uses **MaximumCacheEntries** parameter when creating the first dword of the random, leading to a(n imperfect) fingerprint of two zero bytes in 3/4th byte*

TLS Extensions

- ❑ Server Name Indication (SNI)
 - ❑ Virtual hosts
- ❑ Application-Layer Protocol Negotiation (ALPN)
 - ❑ Limited, but what protocol comes next
 - ❑ fingerprinting?
- ❑ Session Tickets
 - ❑ Key GUID



Schannel Caching Parameters

Parameters:

- ❑ The following control upper-limit of cache time:
 - `m_dwClientLifespan`
 - `m_dwServerLifespan`
 - `m_dwSessionTicketLifespan`
- ❑ All of which:
are set to `0x02255100` (**10hrs** in ms)
- ❑ Also of Interest:
 - `m_dwMaximumEntries` (set to `0x4e20` or **20,000 entries** by default)
 - `m_dwEnableSessionTicket` controls use of session tickets (e.g. 0, 1, 2)
 - `m_dwSessionCleanupIntervalInSeconds` (set to `0x012c` or **300 seconds** by default)

HOWEVER:

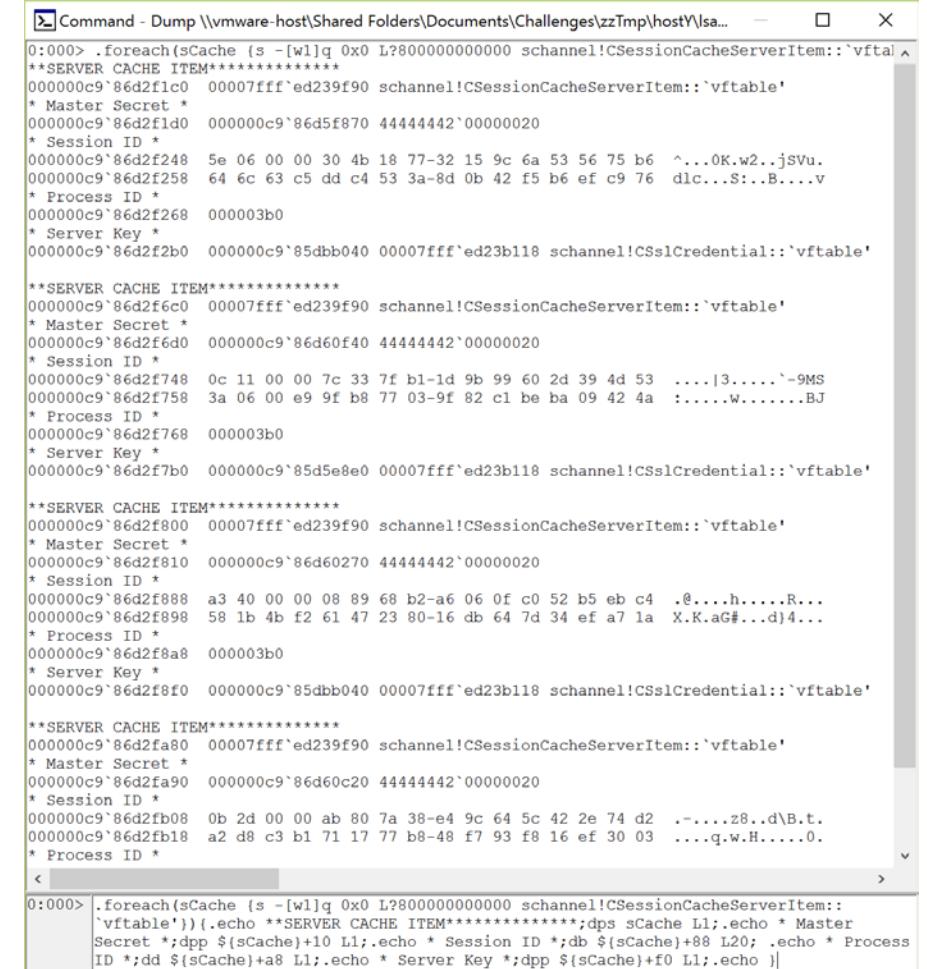
- ❑ Schannel is the library, the `process` has control
- ❑ Proc can purge its own cache at will
 - ❑ For example, IIS reportedly* purges after around two hours
- ❑ Schannel maintains track of process, frees cache items after client proc terminates :<
 - ❑ Haven't looked at the exact mechanism
 - ❑ As you'll see, the upside is that the Process ID is stored in the Cache

This is your Schannel Cache (x64)

```
'_SSL_SESSION_CACHE_CLIENT_ITEM': [ 0x148, {
    'Vftable': [0x0, ['pointer64', ['void']]],
    'MasterKey': [0x10, ['pointer64', ['void']]],
    'PublicCertificate': [0x18, ['pointer64', ['void']]],
    'PublicKey': [0x28, ['pointer64', ['void']]],
    'NcryptSslProv': [0x60, ['pointer64', ['void']]],
    'SessionIdLen': [0x86, ['short short']],
    'SessionId': [0x88, ['array', 0x20, ['unsigned char']]],
    'ProcessId': [0xa8, ['unsigned long']],
    'MaxLifeTime': [0xB0, ['unsigned long']],
    'CertSerializedCertificateChain': [0xB0, ['pointer64', ['void']]],
    'UnkList1Flink': [0xB8, ['pointer64', ['void']]],
    'UnkList1Blink': [0xC0, ['pointer64', ['void']]],
    'UnkCacheList2Flink': [0xC8, ['pointer64', ['void']]],
    'UnkCacheList2Blink': [0xD0, ['pointer64', ['void']]],
    'ServerName': [0x108, ['pointer64', ['void']]],
    'LogonSessionUID': [0x110, ['pointer64', ['void']]],
    'CSessCacheManager': [0x120, ['pointer64', ['void']]],
    'SessionTicket': [0x138, ['pointer64', ['void']]],
    'SessionTicketLen': [0x140, ['int']],
}],
```

This is your Schannel Cache (x64)

```
'_SSL_SESSION_CACHE_SERVER_ITEM': [ 0x110, {
    'vftable': [0x0, ['pointer64', ['void']]],
    'NcryptKey': [0x10, ['pointer64', ['void']]],
    'NcryptSslProv': [0x60, ['pointer64', ['void']]],
    'SessionId': [0x88, ['array', 0x20, ['unsigned char']]],
    'ProcessId': [0xa8, ['unsigned long']],
    'MaxLifeTime': [0xB0, ['unsigned long']],
    'LastError?': [0xE8, ['unsigned long']],
    'CSslCredential': [0xF0, ['pointer64', ['void']]],
}, ],
```



```
0:000> .foreach(sCache { s -[wl]q 0x0 L?8000000000000000 schannel!CSessionCacheServerItem::`vftable`}){.echo **SERVER CACHE ITEM*****;dps sCache L1;.echo * Master Secret *;dpp ${sCache}+10 L1;.echo * Session ID *;db ${sCache}+88 L20;.echo * Process ID *;dd ${sCache}+a8 L1;.echo * Server Key *;dpp ${sCache}+f0 L1;.echo }
```

This is your Schannel Cache on ~~Vista~~

```
'_SSL_SESSION_CACHE_CLIENT_ITEM': [ 0xf0, {
    'Flink': [0x0, ['pointer', ['void']]],
    'Blink': [0x4, ['pointer', ['void']]],
    'ProcessId': [0x8, [['unsigned long']]],
    'MasterKey': [0x14, ['pointer', ['NcryptSslKey']]],
    'CipherSuiteId': [0x1C, ['pointer', ['void']]],
    'ECCurveParam': [0x20, ['pointer', ['void']]],
    'NcryptSslProv': [0x28, ['pointer', ['void']]],
    'PublicCertificate': [0x2C, ['pointer', ['void']]],
    'PublicCert2': [0x34, ['pointer', ['void']]],
    'PublicKeyStruct': [0x3C, ['pointer', ['void']]],
    'PublicCertStruct3': [0x44, ['pointer', ['void']]],
    'ServerName': [0x80, ['pointer', ['void']]],
    'SessionIdSize': [0x94, ['short short']],
    'SessionId': [0x98, ['array', 0x20, ['unsigned char']]],
    'ErrorCode': [0xEC, ['pointer64', ['void']]]
} ],
```



```
Command - Dump \\vmware-host\Shared Folders\Documents\Challenge...  
*** Cache Item ***  
* ProcId: 001d76f0 00000cf8  
* NcrysSslKey: 001d76fc 001dab40 00000018  
* SNI: 001d7768 01f9e480 "live.sysinternals.com"  
* SessionID: 001d7780 59 19 00 00 07 4a 6c cc-d6 b0 e2 b2 5f cd d1 30 Y....Jl.....0  
001d7790 bf ee 06 b1 ec 20 e3 57-e3 79 52 72 d7 f5 a5 41 .....W.yRr..A  
  
*** Cache Item ***  
* ProcId: 001d7828 00000cf8  
* NcrysSslKey: 001d7834 001dabe0 00000018  
* SNI: 001d78a0 01fa3cb8 "www.torproject.org"  
* SessionID: 001d78b8 ba ce 7b 7e ca 6d e8 ae fb 08 bb 71 83 ..{~.m.....q.  
001d78c8 e7 87 ed 78 e5 12 f3 c0-24 a3 b6 0b e8 a2 43 b9 ...x....$.C.  
  
*** Cache Item ***  
* ProcId: 001d7960 00000cf8  
* NcrysSslKey: 001d796c 01fa3f98 00000018  
* SNI: 001d79d8 01fa3d18 "urs.microsoft.com"  
* SessionID: 001d79f0 99 0e 00 00 d8 3f de 02-53 c3 68 49 59 89 c2 c0 .....?..S.hIY...  
001d7a00 71 ca bd 8f 5f 7b bd 59-08 6c df 44 8c a7 b7 7b q..._.Y.l.D...  
  
*** Cache Item ***  
* ProcId: 001d7e40 00000cf8  
* NcrysSslKey: 001d7e4c 01fa3ed8 00000018  
* SNI: 001d7eb8 01f75d88 "login.live.com"  
* SessionID: 001d7ed0 f6 07 00 00 5d 3d bc aa-f7 91 9a 5e f5 3e b7 10 ....]=.....^.>..  
001d7ee0 ab dc 7c d1 1f 3a 0a 95-08 02 80 cc ee 92 4c d1 ..|:.....L.  
  
*** Cache Item ***  
* ProcId: 001d7bd0 00000cf8  
* NcrysSslKey: 001d7bd4 01f5-0570 00000018  
<  
0:000> !list -x ".echo *** Cache Item ***;.echo * ProcId::dd @$extret+8 L1;.echo *  
* NcrysSslKey::dpp @$extret+14 L1;.echo * SNI::dpu @$extret+80 L1;.echo *  
SessionID::db @$extret+98 L20" 001d8ba0
```



Automating it

Volatility / Rekall

- ❑ Plugins for both – by default (no args) they:
 - ❑ Find LSASS
 - ❑ Scan Writeable VADs / Heap for Master Key signature (Volatility) or directly for SessionCacheItems (Rekall)
 - ❑ Dump out the wireshark format shown earlier
- ❑ Hoping to have functional powershell module or maybe incorporation into mimikatz? (Benjamin Delphy is kinda the man for LSASS)

```
% vol.py --plugins=./plugins --profile=Win10x64 -f ./Win10-Test-c2a4a77d.vmdk  
em lsasslkey  
Volatility Foundation Volatility Framework 2.5  
RSA Session-ID:b93c0000a110690b4ae9111bce5725c6c47a037b3c39c49c75ce51e1c2eb79ee M  
aster-Key:bc28467999b99fd3fdf3a24642c5d93b9ab43e51627f6e0145ef120ba98a1c3223f3dbe  
0154e30d7869bdb7ab66f5318  
RSA Session-ID:173300000f84a86aebb2c5de0af20e6d5c2cab95ab65043e14c6e19cee54ee17 M  
aster-Key:9dd750e12e6e4439b08326d4a1f9eba2d2fe65c2a26c2088e7cec22ce1d91e9f219b704  
547a2b2eccb9a81d557d5ae1a  
RSA Session-ID:3c2c000024b8f70dd2613d8b13d0c4ac4daaefbe53ab4b7cb9763e80feccb4f1 M  
aster-Key:2d119c64695ffc9c143c136471f5625d8cde92d35721f5f2849b92639603799a45e1e60  
1786cbf89b00c186969d44983  
RSA Session-ID:d4170000da09f8596739215e216c496568fa66e42ac32b974d440949dff33d2b M  
aster-Key:44b503bef7842ea9a416fbf8b63b932b23b7b687fbf5297b253eac427877c8e11595e14  
c3f00c40bf2a0f4688de0b7aa  
RSA Session-ID:432a0000bf4f622f0fc119974a0ef30cd838c3a025b83abbdcdcbce7b2325d2d9 M  
aster-Key:552699d61e21d1b871af4b05a54003bf03eade60666dd1e54b94c3b5ec98f296db4ae99  
baed4e23882175e5ffd88be31  
RSA Session-ID:6f230000a021aac48d15544524c1454e4ec01d5adb305d8d9d57ab2b991dd597 M  
aster-Key:8bc9e9df653e3cbf533be84c6897787bd453b8cee9d5389e9c3659ebf997d9c8d0666aa  
dca5be2258f30b9251215a717
```

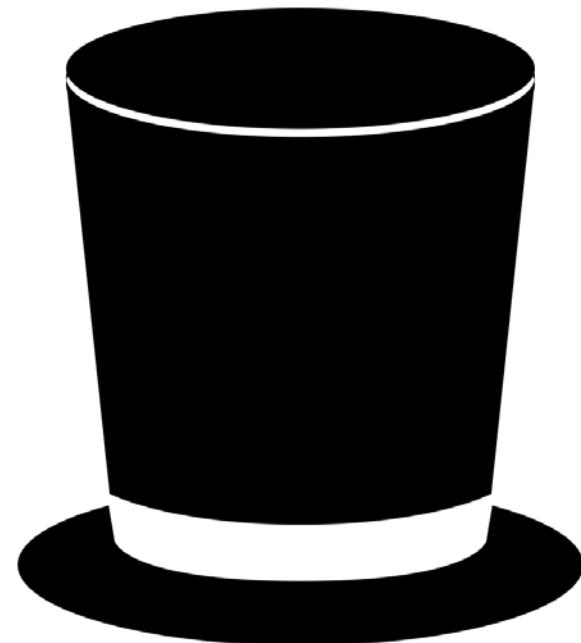


Limitations

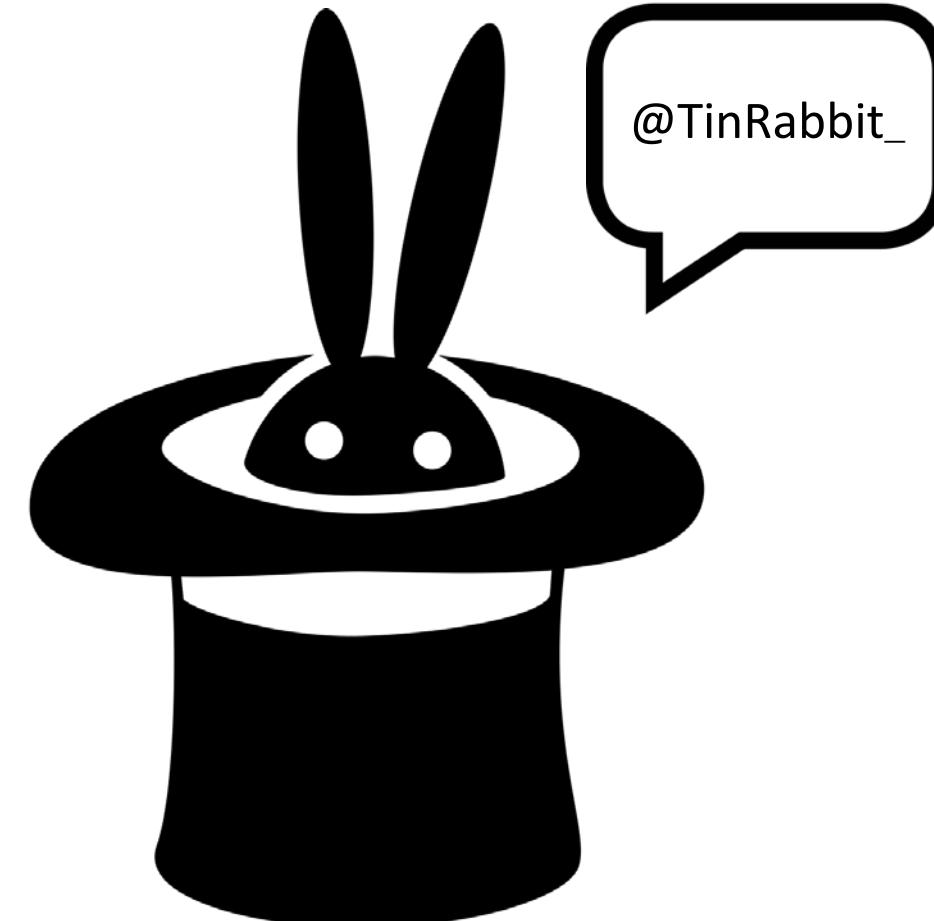
- **We're working with internal, undocumented structures**
 - They change over time -- sometime around April 2016, an element appears to have been inserted in cache after the SessionID and before the SNI
 - Not a huge deal, except when differences amongst instances of same OS (e.g. ones that have and have not been updated)
- **Relying on symbols for some of this**
 - MS giveth and can taketh away.
 - Still, can be done without them, just slightly less efficiently.
- **You need to be able to read LSASS memory**
 - Not a huge deal in 2016, but still merits mention -- you need to own the system
 - If you own the system, you can already do bad stuff (keylog / tap net interface)
 - This is why it's probably most useful in a forensic context



Demo



Fin.



Questions?



Special Thanks

For general support, helpful comments, their time, and encouragement.

Áine Doyle - Badass Extraordinaire (OCSC)

Dr. John-Ross Wallrabenstein - Sypris Electronics

Dr. Marcus Rogers - Purdue Cyber Forensics Laboratory

Michael Hale Ligh (MHL) - Volexity

Tatiana Ringenberg - Sypris Electronics