



Mitigating the unkn0wn

When your SMB exploit fails

Nicolas Joly



Aug. 25-26 2017 in Taipei



Microsoft



Security Engineer at the MSRC

- Exploiting stuff, breaking things
- Have played pwn2own before, now judging entries...



Aug. 25-26 2017 in Taipei



Protecting customers and evaluating risk

Rate this article ★★★★★



MSRC Team April 14, 2017

Share 152

2412

677

0

Today, Microsoft triaged a large release of exploits made publicly available by Shadow Brokers. Understandingly, customers have expressed concerns around the risk this disclosure potentially creates. Our engineers have investigated the disclosed exploits, and most of the exploits are already patched. Below is our update on the investigation.

When a potential vulnerability is reported to Microsoft, either from an internal or external source, the Microsoft Security Response Center (MSRC) kicks off an immediate and thorough investigation. We work to swiftly validate the claim and make sure legitimate, unresolved vulnerabilities that put customers at risk are fixed. Once validated, engineering teams prioritize fixing the reported issue as soon as possible, taking into consideration the time to fix it across any impacted product or service, as well as versions, the potential threat to customers, and the likelihood of exploitation.

Most of the exploits that were disclosed fall into vulnerabilities that are already patched in our supported products. Below is a list of exploits that are confirmed as already addressed by an update. We encourage customers to ensure their computers are up-to-date.

Code Name	Solution
"EternalBlue"	Addressed by MS17-010
"EmeraldThread"	Addressed by MS10-061
"EternalChampion"	Addressed by MS17-010
"ErraticGopher"	Addressed prior to the release of Windows Vista. CVE-2017-8461
"EsikmoRoll"	Addressed by MS14-068
"EternalRomance"	Addressed by MS17-010
"EducatedScholar"	Addressed by MS09-050
"EternalSynergy"	Addressed by MS17-010
"EclipsedWing"	Addressed by MS08-067

<https://blogs.technet.microsoft.com/msrc/2017/04/14/protecting-customers-and-evaluating-risk/>

Microsoft Security Bulletin MS17-010 - Critical

Security Update for Microsoft Windows SMB Server (4013389)

Published: March 14, 2017

Operating System	Windows SMB Remote Code Execution Vulnerability – CVE-2017-0143	Windows SMB Remote Code Execution Vulnerability – CVE-2017-0144	Windows SMB Remote Code Execution Vulnerability – CVE-2017-0145	Windows SMB Remote Code Execution Vulnerability – CVE-2017-0146	Windows SMB Information Disclosure Vulnerability – CVE-2017-0147	Windows SMB Remote Code Execution Vulnerability – CVE-2017-0148
Windows Vista						
Windows Vista Service Pack 2 (4012598)	Critical Remote Code Execution	Critical Remote Code Execution	Critical Remote Code Execution	Critical Remote Code Execution	Important Information Disclosure	Critical Remote Code Execution

6 CVEs, 5 Critical

<https://technet.microsoft.com/en-us/library/security/ms17-010.aspx>

EternalBlue



Aug. 25-26 2017 in Taipei



EternalBlue

CVE-2017-0144 – Integer overflow due to storing a Ulong as a Ushort in SrvOs2FeaListSizeToNt

```
if (variableBuffer >= lastValidLocation ||  
    (variableBuffer + fea->cbName + 1 + SmbGetUshort(&fea->cbValue)) > lastValidLocation) {  
    SmbPutUshort( &FeaList->cbList, PTR_DIFF_SHORT(fea, FeaList) );  
    break;  
}
```

```
typedef struct _FEA {
```

```
    BYTE fEA;  
    BYTE cbName;  
    USHORT cbValue;
```

```
} FEA;
```

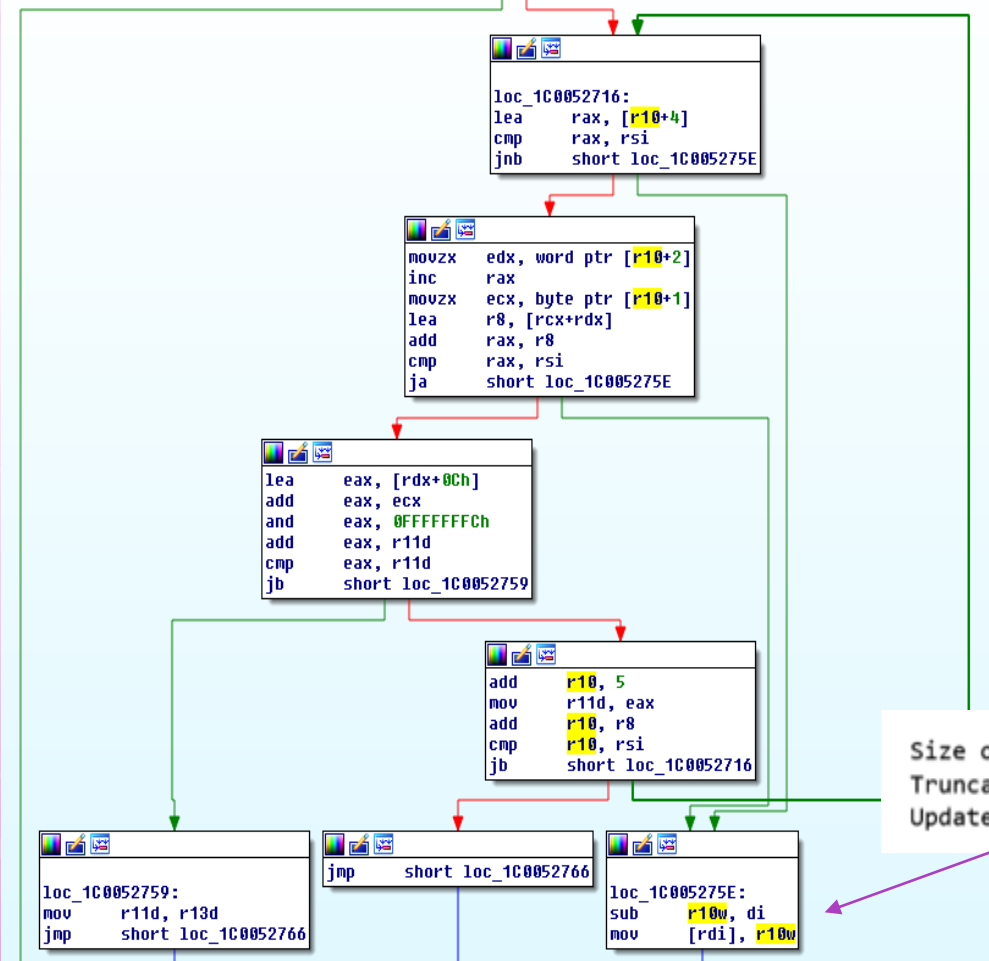
```
typedef struct _FEALIST {
```

```
    ULONG cbList;  
    FEA list[1];  
} FEALIST, *PFEALIST;
```

<http://blog.trendmicro.com/trendlabs-security-intelligence/ms17-010-eternalblue/>

EternalBlue

CVE-2017-0144 – Integer overflow due to storing a Ulong as a Ushort in SrvOs2FeaListSizeToNt



SrvOs2FeaListSizeToNt inlined in
SrvOs2FeaListToNt

Integer truncation here

Size of list: 0x00010000
Truncated size: 0x0000FF5D
Updated size of list: 0x0001FF5D [actually larger than the original]

EternalBlue

CVE-2017-0144 – Integer overflow due to storing a Ulong as a Ushort in SrvOs2FeaListSizeToNt

Frame 10: 1138 bytes on wire (9104 bits), 1138 bytes captured on interface (9104 bits) on vmnic0

Ethernet II, Src: Vmware_c0:00:01 (00:50:56:c0:00:01), Dst: 192.168.193.1

Internet Protocol Version 4, Src: 192.168.193.1, Dst: 192.168.193.1

Transmission Control Protocol, Src Port: 6022, Dst Port: 445

NetBIOS Session Service

SMB (Server Message Block Protocol)

- SMB Header
- NT Trans Request (0xa0)
 - Word Count (WCT): 20
 - Max Setup Count: 1
 - Reserved: 0000
 - Total Parameter Count: 30
 - Total Data Count: 66512**
 - Max Parameter Count: 30
 - Max Data Count: 0
 - Parameter Count: 30
 - Parameter Offset: 75
 - Data Count: 976
 - Data Offset: 104
 - Setup Count: 1
 - Function: Unknown (0)
 - Unknown NT transaction (0) Setup
 - Byte Count (BCC): 1004
 - Unknown NT transaction (0) Parameters

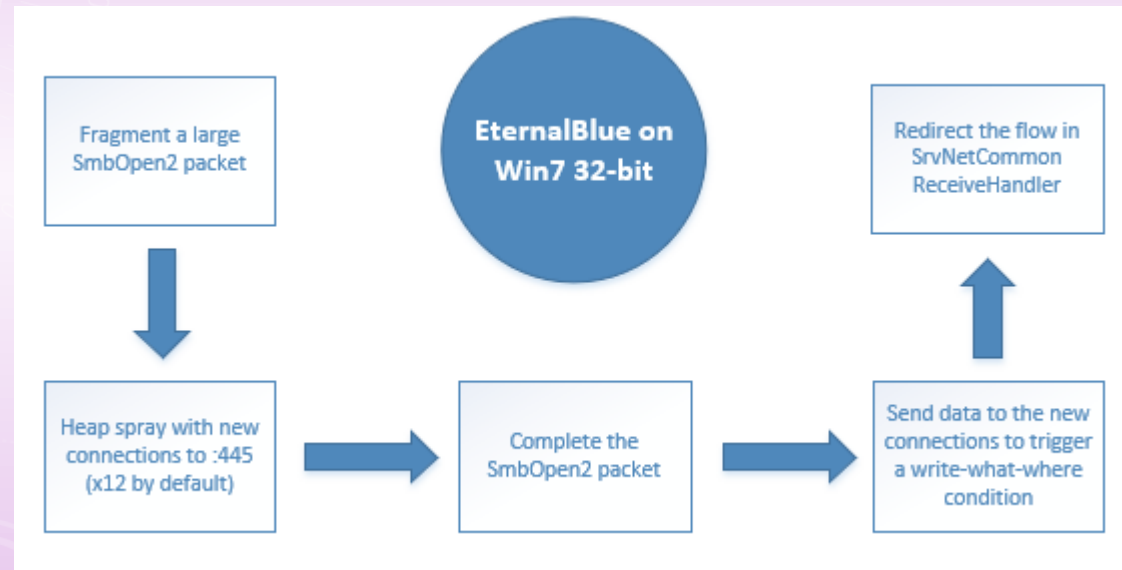
0000 00 0c 29 95 98 26 00 50 56 c0 00 01 08 00 45 00 ..).&.P V....E.
0010 04 64 4a 3e 40 00 80 06 a8 6e c0 a8 c1 01 c0 a8 .dJ>@... .n.....
0020 c1 94 17 86 01 bd e4 43 f0 c1 2b 60 ec c3 50 18C ..+...P.
0030 08 03 98 8d 00 00 00 00 04 38 ff 53 4d 42 a0 008.SMB..
0040 00 00 00 18 07 c0 00 00 00 00 00 00 00 00 00 00
0050 00 00 00 08 ff fe 00 08 40 00 14 01 00 00 1e 00@..
0060 00 00 d0 03 01 00 1e 00 00 00 00 00 00 00 1e 00
0070 00 00 4b 00 00 00 d0 03 00 00 68 00 00 00 01 00K.....h..
0080 00 00 00 ec 03 00 00 00 00 00 00 00 00 00 00 00
0090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00a0 00 00 00 00 01 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 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0160 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0170 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

FEA List size

<http://blog.trendmicro.com/trendlabs-security-intelligence/ms17-010-eternalblue/>

EternalBlue

CVE-2017-0144 – Integer overflow due to storing a Ulong as a Ushort in SrvOs2FeaListSizeToNt



EternalBlue in action

EternalBlue

CVE-2017-0144 – Integer overflow due to storing a Ulong as a Ushort in SrvOs2FeaListSizeToNt

114	0.128374	33.0.0.15	33.0.0.2	TCP	66 49827 → 445 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
115	0.128744	33.0.0.2	33.0.0.15	TCP	66 445 → 49827 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
116	0.128798	33.0.0.15	33.0.0.2	TCP	54 49827 → 445 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
117	0.128845	33.0.0.15	33.0.0.2	TCP	186 [TCP segment of a reassembled PDU]
118	0.129051	33.0.0.15	33.0.0.2	TCP	66 49828 → 445 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
119	0.129179	33.0.0.2	33.0.0.15	TCP	66 445 → 49828 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
120	0.129227	33.0.0.15	33.0.0.2	TCP	54 49828 → 445 [ACK] Seq=1 Ack=1 Win=262656 Len=0
121	0.129279	33.0.0.15	33.0.0.2	TCP	186 [TCP segment of a reassembled PDU]
122	0.129430	33.0.0.15	33.0.0.2	TCP	66 49829 → 445 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
123	0.129530	33.0.0.2	33.0.0.15	TCP	66 445 → 49829 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
124	0.129577	33.0.0.15	33.0.0.2	TCP	54 49829 → 445 [ACK] Seq=1 Ack=1 Win=262656 Len=0
125	0.129617	33.0.0.15	33.0.0.2	TCP	186 [TCP segment of a reassembled PDU]

158	1.142216	33.0.0.15	33.0.0.2	TCP	4126 [TCP segment of a reassembled PDU]
159	1.142276	33.0.0.15	33.0.0.2	TCP	4126 [TCP segment of a reassembled PDU]
160	1.142333	33.0.0.15	33.0.0.2	TCP	4126 [TCP segment of a reassembled PDU]
161	1.142385	33.0.0.15	33.0.0.2	TCP	4126 [TCP segment of a reassembled PDU]
162	1.142443	33.0.0.15	33.0.0.2	TCP	4126 [TCP segment of a reassembled PDU]

Spraying with new connections to :445

EternalBlue

CVE-2017-0144 – Integer overflow due to storing a Ulong as a Ushort in SrvOs2FeaListSizeToNt

Sending new connections leads
to spray the pool with MDLs:

```
854aeff0 da 79 3c 1f 00 00 00 00 00 00 06 2a 00 00 00 00
854af000 00 10 01 00 00 00 00 00 42 00 00 00 00 00 00 00
854af010 08 00 12 c0 00 00 00 00 00 00 00 00 60 f1 4a 85
854af020 a0 0e 01 00 00 00 00 00 3c f0 4a 85 00 00 00 00
854af030 f7 ff 00 00 78 6b 45 87 a4 f0 4a 85 00 00 00 00
854af040 60 00 04 10 00 00 00 00 60 f1 4a 85 00 f0 4a 85
854af050 a0 0e 01 00 60 01 00 00 af fc 07 00 b0 fc 07 00
```

```
854aeff0 b4 00 00 00 80 00 a8 00 00 00 00 00 00 00 00 00
854af000 00 00 00 00 00 00 00 00 00 ff ff 00 00 00 00 00
854af010 ff ff 00 00 00 00 00 00 00 00 00 00 00 00 00 00
854af020 00 00 00 00 00 00 00 00 00 00 ff df ff 00 00 00 00
854af030 00 00 00 00 20 f0 df ff 00 11 df ff ff ff ff ff
854af040 60 00 04 10 00 00 00 00 80 ef df ff 00 00 00 00
854af050 10 00 d0 ff ff ff ff ff 18 01 d0 ff ff ff ff ff
```

This points to HAL

Pool before and after the overflow

EternalBlue

CVE-2017-0144 – Integer overflow due to storing a Ulong as a Ushort in SrvOs2FeaListSizeToNt

```
95d1b26a 8b866c010000 mov     eax,dword ptr [esi+16Ch]
95d1b270 85c0          test    eax,eax
95d1b272 0f8493000000 je      srvnet!SrvNetCommonReceiveHandler+0xff (95d1b30b)
95d1b278 837e0803     cmp     dword ptr [esi+8],3
95d1b27c 7557         jne     srvnet!SrvNetCommonReceiveHandler+0xc9 (95d1b2d5)
95d1b27e ff7528       push    dword ptr [ebp+28h]
95d1b281 ff7524       push    dword ptr [ebp+24h]
95d1b284 ff751c       push    dword ptr [ebp+1Ch]
95d1b287 ff7514       push    dword ptr [ebp+14h]
95d1b28a 51          push    ecx
95d1b28b ff750c       push    dword ptr [ebp+0Ch]
95d1b28e ff7518       push    dword ptr [ebp+18h]
95d1b291 ffb6ac000000 push    dword ptr [esi+0ACh]
95d1b297 ffb6a8000000 push    dword ptr [esi+0A8h]
95d1b29d ff5004       call    dword ptr [eax+4]
```

Redirecting the flow in srvnet!SrvNetCommonReceiveHandler

EternalBlue

CVE-2017-0144 – Integer overflow due to storing a Ulong as a Ushort in SrvOs2FeaListSizeToNt

Fix? Use PTR_DIFF instead of PTR_DIFF_SHORT

Here is the correct vulnerable code path for Windows 10 version 1511:

```
srv!SrvOs2FeaListToNt+0x8a:  
fffff801`8de5247e 66442bd7      sub     r10w,di  
fffff801`8de52482 66448917      mov     word ptr [rdi],r10w
```

How the vulnerability was patched with MS17-010:

```
srv!SrvOs2FeaListSizeToNt+0x6e:  
fffff800`a2521d72 452bc2      sub     r8d,r10d  
fffff800`a2521d75 458902      mov     dword ptr [r10],r8d
```

The 16-bit registers were replaced with 32-bit versions, to prevent the mathematical miscalculation leading to buffer overflow.

<https://zerosum0x0.blogspot.co.uk/2017/06/>

EternalChampion



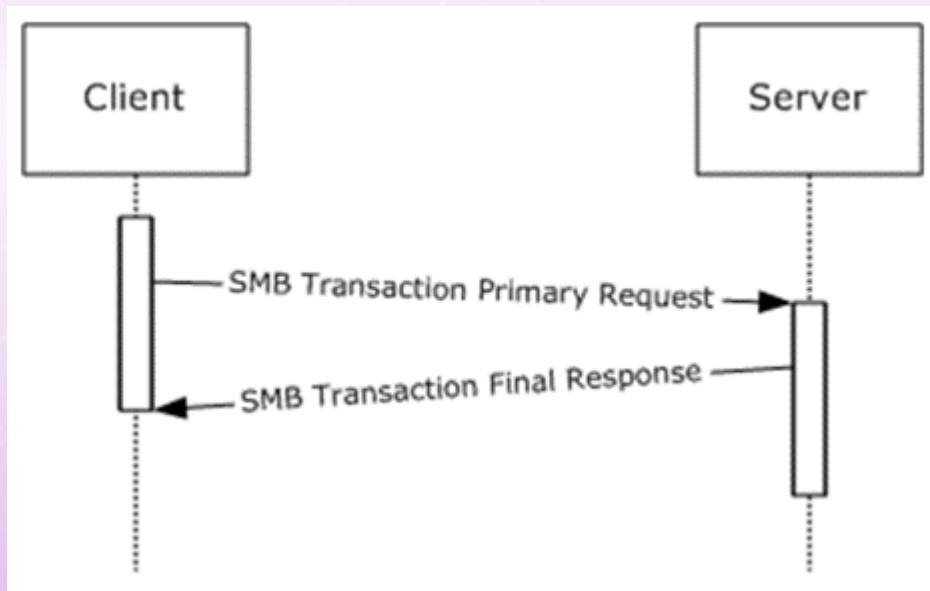
Aug. 25-26 2017 in Taipei



EternalChampion

CVE-2017-0146 – Race condition with Transaction requests

Sending simple transactions



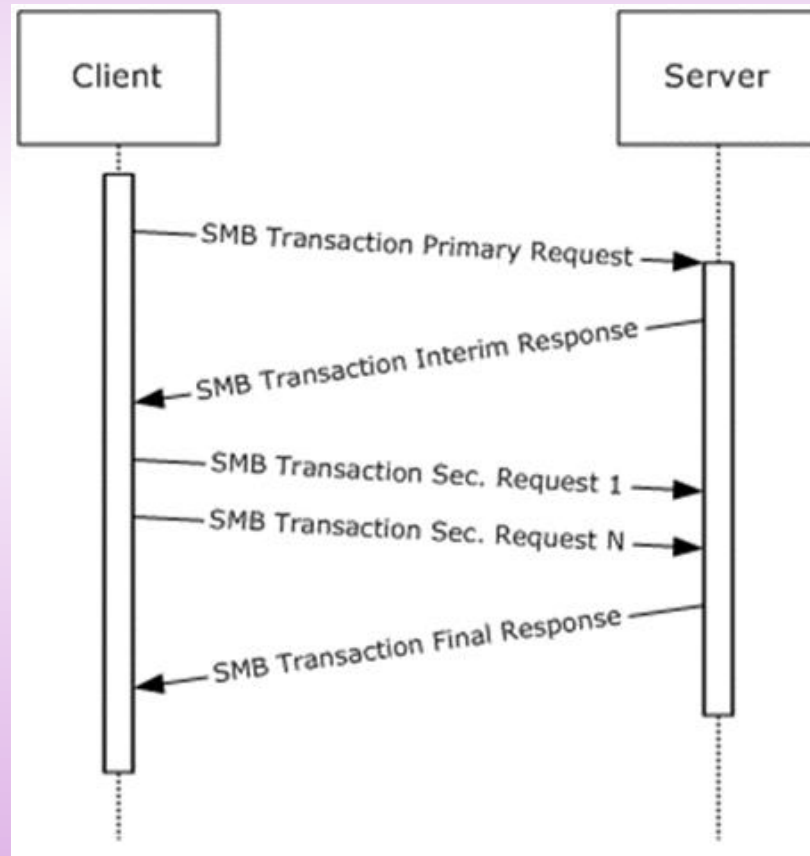
```
srv!TRANSACTION
...
+0x010 Connection      : Ptr64 _CONNECTION
...
+0x058 cMaxBufferSize  : Uint4B
+0x060 InSetup         : Ptr64 Uint2B
+0x068 OutSetup        : Ptr64 Uint2B
+0x070 InParameters    : Ptr64 Char
+0x078 OutParameters   : Ptr64 Char
+0x080 InData          : Ptr64 Char // data received
+0x088 OutData         : Ptr64 Char // data to send (same buffer as InData)
+0x090 SetupCount      : Uint4B
+0x098 ParameterCount  : Uint4B
+0x09c TotalParameterCount : Uint4B
+0x0a4 DataCount       : Uint4B // data received so far
+0x0a8 TotalDataCount  : Uint4B // total data being expected
...
+0x0e3 Executing       : UChar
```

<https://msdn.microsoft.com/en-us/library/ee441928.aspx>

EternalChampion

CVE-2017-0146 – Race condition with Transaction requests

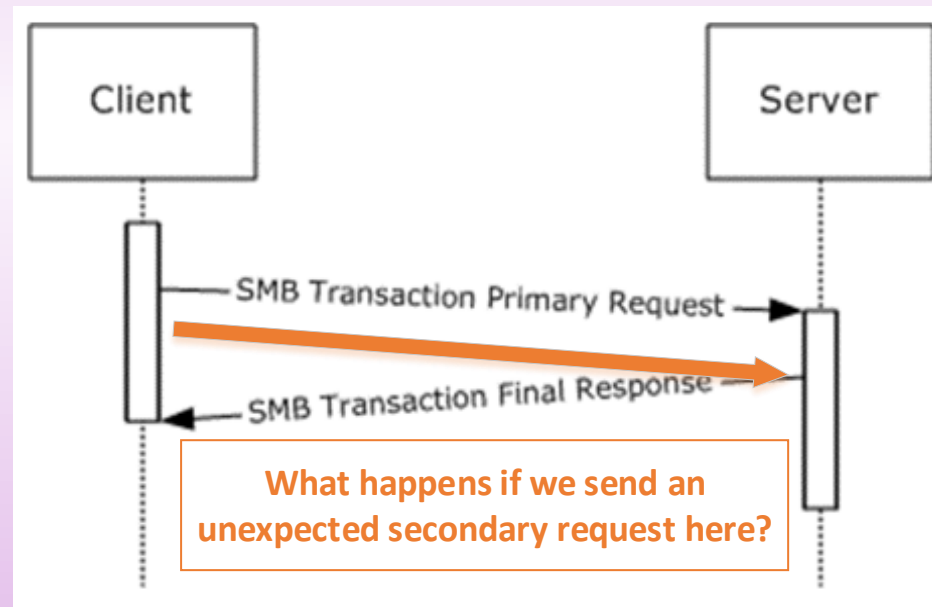
But, if parameters or data don't fit in the first Transaction, we can use secondary transactions (while $xCount < TotalxCount$, send...)



EternalChampion

CVE-2017-0146 – Race condition with Transaction requests

How does the server know when to process the request? Can we race it?

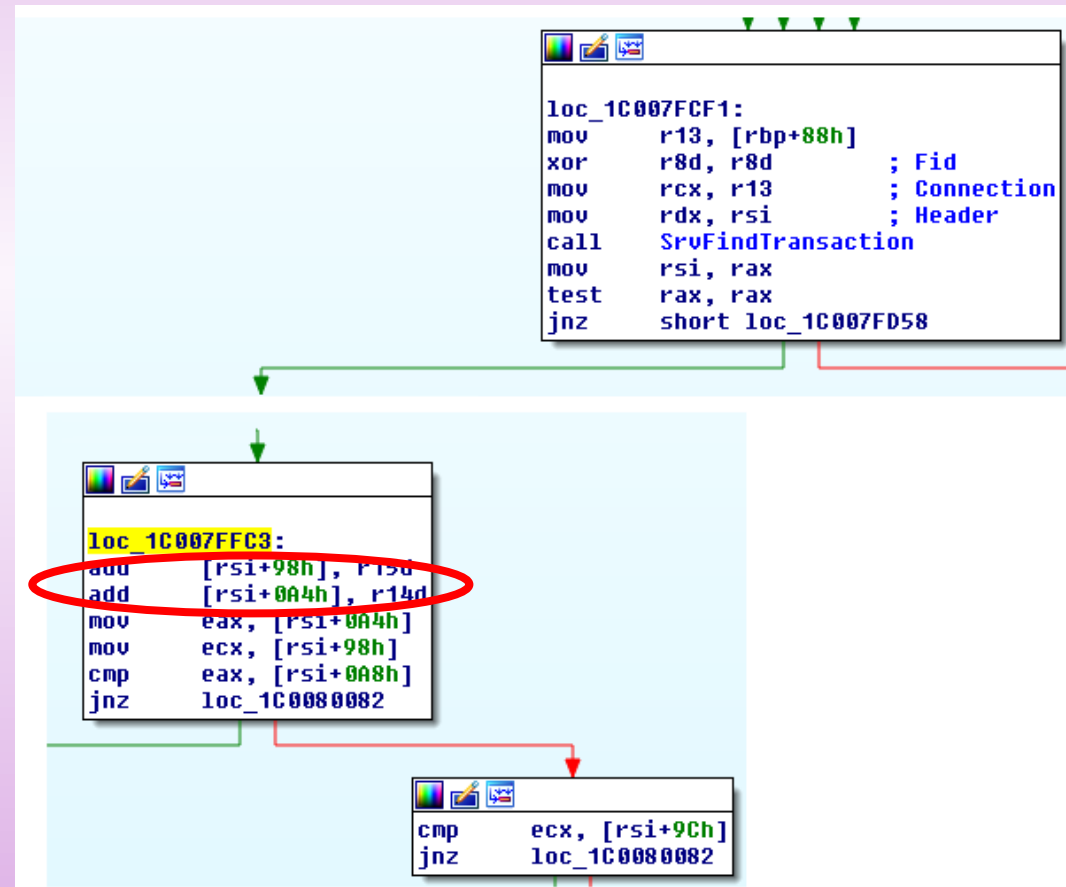


<https://msdn.microsoft.com/en-us/library/ee441928.aspx>

EternalChampion

CVE-2017-0146 – Race condition with Transaction requests

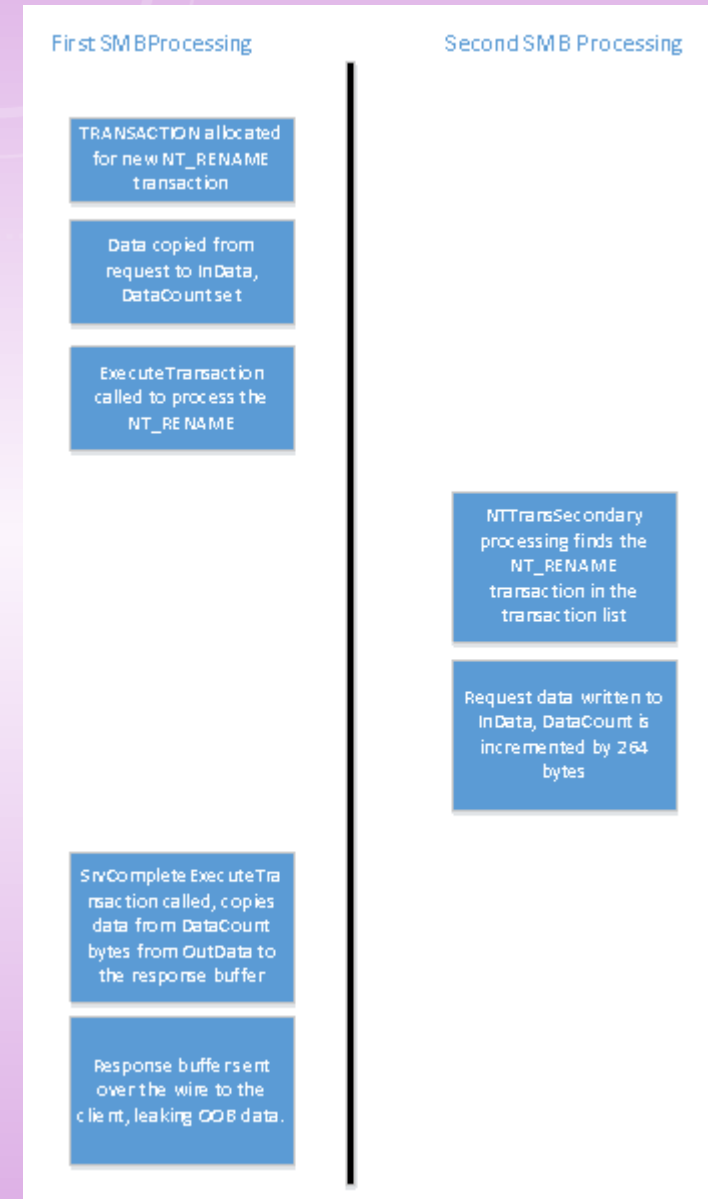
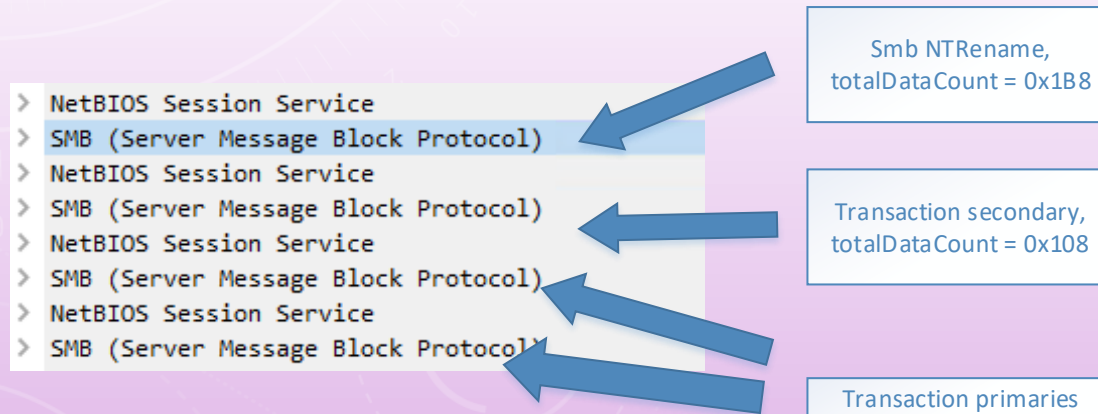
Even if the primary transaction is being processed,
the secondary transaction will still increment
some fields, like Transaction -> DataCount and
ParameterCount



EternalChampion

CVE-2017-0146 – Race condition with Transaction requests

Exploit is in two ways. First make an info leak with SrvSmbNtRename and an additional secondary transaction, followed by several transaction requests to spray the heap:



EternalChampion

CVE-2017-0146 – Race condition with Transaction requests

68	22.781089	33.0.0.2	33.0.0.15	SMB	570 NT Trans Response, NT RENAME
69	22.781099	33.0.0.2	33.0.0.15	SMB	93 Trans Response
70	22.781108	33.0.0.15	33.0.0.2	TCP	54 49997 → 445 [ACK] Seq=3610 Ack=1763 Win=2102272 Len=0
71	22.781126	33.0.0.2	33.0.0.15	SMB	93 Trans Response
72	22.781126	33.0.0.2	33.0.0.15	SMB	93 Trans Response
73	22.781137	33.0.0.15	33.0.0.2	TCP	54 49997 → 445 [ACK] Seq=3610 Ack=1841 Win=2102272 Len=0
74	22.781148	33.0.0.2	33.0.0.15	SMB	93 Trans Response
75	22.781156	33.0.0.2	33.0.0.15	SMB	93 Trans Response
76	22.781164	33.0.0.15	33.0.0.2	TCP	54 49997 → 445 [ACK] Seq=3610 Ack=1919 Win=2102016 Len=0
77	22.781172	33.0.0.2	33.0.0.15	SMB	93 Trans Response
78	22.781367	33.0.0.2	33.0.0.15	SMB	93 Trans Response
79	22.781368	33.0.0.2	33.0.0.15	SMB	93 Trans Response
80	22.781368	33.0.0.2	33.0.0.15	SMB	93 Trans Response
81	22.781368	33.0.0.2	33.0.0.15	SMB	93 Trans Response
82	22.781369	33.0.0.2	33.0.0.15	SMB	93 Trans Response
83	22.781369	33.0.0.2	33.0.0.15	SMB	93 Trans Response
84	22.781406	33.0.0.15	33.0.0.2	TCP	54 49997 → 445 [ACK] Seq=3610 Ack=2192 Win=2101760 Len=0
85	22.781426	33.0.0.2	33.0.0.15	SMB	93 Trans Response
86	22.781426	33.0.0.2	33.0.0.15	SMB	398 NT Trans Response, <unknown>[Unresembled Packet]
87	22.781427	33.0.0.2	33.0.0.15	SMB	93 Trans Response

First response to the Smb NtRename

Second response to the secondary transaction, this time leaking memory

EternalChampion

CVE-2017-0146 – Race condition with Transaction requests

By spraying Transaction objects it becomes possible to have one located right after the SmbNtRename transaction buffer and leak some pointers from the Transaction object:

```
VOID SRVFASTCALL
RestartTransactionResponse (
    IN OUT PWORK_CONTEXT WorkContext
)
//
//
//
```

```
if ( paramLength != 0 ) {
    RtlMoveMemory(
        paramPtr,
        transaction->OutParameters + paramDisp,
        paramLength
    );
}

if ( dataLength != 0 ) {
    RtlMoveMemory(
        dataPtr,
        transaction->OutData + dataDisp,
        dataLength
    );
}
```

9a448188	68	25	4d	13	36	08	0e	52	7b	54	07	20	40	74	45	41	h%M.6..R{T. @tEA
9a448198	78	4b	7f	73	60	3e	30	00	26	22	1b	1d	29	1b	67	0a	xK.s`>0.&"...).g.
9a4481a8	7b	04	01	00	77	2c	01	5d	00	00	00	00	b8	51	00	00	{...w..].....Q..
9a4481b8	00	02	01	06	46	72	61	67	01	02	01	00	46	72	65	65Frag....Free
9a4481c8	01	02	c7	0b	4c	53	74	72	d0	e1	64	88	00	00	00	00LStr..d.....
9a4481d8	0c	03	24	0e	00	00	00	00	58	90	48	85	70	9c	45	85	..\$.X.H.p.E.
9a4481e8	58	8b	33	99	d8	1d	5a	88	28	c0	44	9a	20	30	44	9a	X.3...Z.(.D. 0D.
9a4481f8	00	00	00	00	00	00	02	00	78	82	44	9a	f8	d9	00	00x.D.....
9a448208	ff	ff	ff	ff	00	00	00	00	7c	82	44	9a	00	00	00	00D.....
9a448218	7c	82	44	9a	fc	8f	44	9a	7c	82	44	9a	fc	8f	44	9a	.D...D. .D...D.
9a448228	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

dataDisp goes out of bounds!

EternalChampion

CVE-2017-0146 – Race condition with Transaction requests

Several packets follow next:

- A QueryPathInformation packet to have transaction->InData point to a stack address:
- A bunch of secondary transactions following to reach that memcpy and trigger a stack corruption in SrvSmbTransactionSecondary:

```
if ( informationLevel == SMB_INFO_IS_NAME_VALID ) {  
  
    transaction->InData = (PVOID)&objectName;  
  
    //  
    // Get the Share root handle.  
    //  
    smbStatus = SrvGetShareRootHandle( WorkContext->TreeConnect->Share );  
}
```

```
if ( dataCount != 0 ) {  
    RtlMoveMemory(  
        transaction->InData + dataDisplacement,  
        (PCHAR)header + dataOffset,  
        dataCount  
    );  
}
```

EternalChampion

CVE-2017-0146 – Race condition with Transaction requests

That results in a 4-byte overwrite in the stack:

```
0: kd> g
Breakpoint 15 hit
srv!SrvSmbTransactionSecondary+0x1eb:
96c6b2d0 8b4648      mov     eax,dword ptr [esi+48h]
3: kd> !address poi(@esi+48)
Mapping user range ...
Mapping system range ...
Mapping page tables...
Mapping hyperspace...
Mapping HAL reserved range...
Mapping User Probe Area...
Mapping system shared page...
Mapping VAD regions...
Mapping module regions...
Mapping process, thread, and stack regions...
Mapping system cache regions...

Usage:
Base Address: 955c1000
End Address: 955c4000
Region Size: 00003000
VA Type: SystemPTEs
```

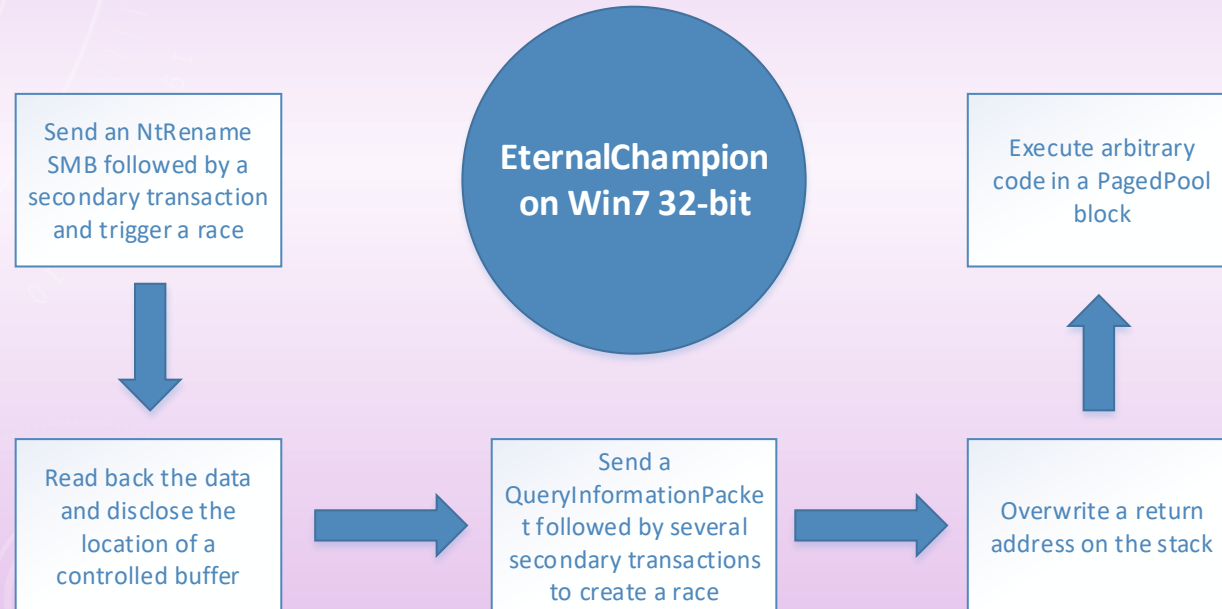
Leading to execute arbitrary code from a Paged pool:

```
3: kd> !address @eip
Mapping user range ...
Mapping system range ...
Mapping page tables...
Mapping hyperspace...
Mapping HAL reserved range...
Mapping User Probe Area...
Mapping system shared page...
Mapping VAD regions...
Mapping module regions...
Mapping process, thread, and stack regions...
Mapping system cache regions...

Usage:
Base Address: 9b200000
End Address: 9b400000
Region Size: 00200000
VA Type: PagedPool
```

EternalChampion

CVE-2017-0146 – Race condition with Transaction requests



EternalChampion in action

EternalRomance – 2 different bugs



Aug. 25-26 2017 in Taipei



EternalRomance

CVE-2017-0143 – Type confusion between WriteAndX and Transaction requests

CVE-2017-0147 – Info disclosure in SrvPeekNamedPipe

RestartPeekNamedPipe and SrvCompleteExecuteTransaction could be abused together to return uninitialized memory after calling SrvPeekNamedPipe:

```
VOID SRVFASTCALL  
RestartPeekNamedPipe (  
    IN OUT PWORK_CONTEXT WorkContext  
)
```

```
transaction->SetupCount = 0;  
transaction->ParameterCount = 6;  
transaction->DataCount = (ULONG)WorkContext->Irp->IoStatus.Information -  
    (4 * sizeof(ULONG));
```

```
VOID  
SrvCompleteExecuteTransaction (  
    IN OUT PWORK_CONTEXT WorkContext,  
    IN SMB_TRANS_STATUS ResultStatus  
)
```

```
if ( paramLength != 0 ) {  
    RtlMoveMemory( paramPtr, transaction->OutParameters, paramLength );  
}  
  
if ( dataLength != 0 ) {  
    RtlMoveMemory( dataPtr, transaction->OutData, dataLength );  
}
```

dataLength was not
checked against OutData,
thus leaking memory

EternalRomance

CVE-2017-0143 – Type confusion between WriteAndX and Transaction requests

CVE-2017-0147 – Info disclosure in SrvPeekNamedPipe

An initial request with Max Parameter Count = 0x5400 would lead to allocate 0x54A8 in the PagedPool bytes in SrvAllocateTransaction...

Max Parameter Count: 21504		
Max Data Count: 1		
Max Setup Count: 0		
Reserved: 00		
Flags: 0x0000		
..... = One Way Transaction: Two way transaction		
0000	00 15 5d 63 72 32 00 15 5d 63 72 05 08 00 45 00	..]cr2..]cr...E.
0010	00 6f 6c 59 40 00 80 06 00 00 21 00 00 0f 21 00	.oLY@... ..!...!
0020	00 02 c3 9a 01 bd 5a bc a4 cc 7e 33 da dd 50 18Z. ..~3..P.
0030	20 11 42 72 00 00 00 00 00 43 ff 53 4d 42 25 00	.Br.... .C.SMB%.
0040	00 00 00 18 07 c0 00 00 00 00 00 00 00 00 00
0050	00 00 00 08 dc 89 00 08 40 00 10 00 00 00 00 @.....
0060	54 01 00 00 00 00 00 00 10 00 00 00 00 00 00	T.....
0070	00 00 00 00 02 00 23 00 00 40 00 00# ..@..

Reserved: 0000		
Parameter Count: 6		
Parameter Offset: 56		
Parameter Displacement: 0		
Data Count: 580		
Data Offset: 64		
0060	00 06 00 38 00 00 00 44 02 40 00 00 00 00 00 4d	...8...D.@.....M
0070	02 00 44 02 44 02 00 00 00 00 00 00 00 00 00	..D.D... ..
0080	00 00 b8 54 00 00 00 06 01 06 46 72 61 67 01 06	...T.... ..Frag..
0090	68 01 46 72 65 65 38 90 24 85 38 90 24 85 00 00	h.Free8. \$.8.\$...
00a0	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

...which would lead to leak the bytes
at @buffer + 0x54A8 in the response

178	405.882513	33.0.0.15	33.0.0.2	RPC_BR...	706 NetrBrowserStatisticsGet request[
179	405.883017	33.0.0.2	33.0.0.15	SMB	105 Write AndX Response, FID: 0x4000,
181	407.390234	33.0.0.15	33.0.0.2	SMB	125 Trans Request
232	542.046955	33.0.0.2	33.0.0.15	SMB	702 Trans Response

EternalRomance

CVE-2017-0143 – Type confusion between WriteAndX and Transaction requests

CVE-2017-0147 – Info disclosure in SrvPeekNamedPipe

2.2.4.3 SMB_COM_WRITE_ANDX (0x2F)

2.2.4.3.1 Client Request Extensions

An SMB_COM_WRITE_ANDX request is sent by a client to write data to a file or **named pipe** on a server. These extensions allocate the **SMB_Parameters.Words.Reserved** field for use as the **DataLengthHigh** field. This field is used when the CAP_LARGE_WRITEX capability has been negotiated to allow for file writes larger than 0xFFFF bytes in length. All other fields are defined as specified in [\[MS-CIFS\]](#) section 2.2.4.43.1.

```
SMB_Parameters
{
    UCHAR WordCount;
    Words
    {
        UCHAR AndXCommand;
        UCHAR AndXReserved;
        USHORT AndXOffset;
        USHORT FID;
        ULONG Offset;
        ULONG Timeout;
        USHORT WriteMode;
        USHORT Remaining;
        USHORT DataLengthHigh;
        USHORT DataLength;
        USHORT DataOffset;
        ULONG OffsetHigh (optional);
    }
}
SMB_Data
{
    USHORT ByteCount;
    Bytes
    {
        UCHAR Pad;
        UCHAR Data[variable];
    }
}
```

[MS-SMB].pdf

```
if ( (SmbGetUshort( &request->WriteMode ) &
      SMB_WMODE_WRITE_RAW_NAMED_PIPE) != 0 ) {

    //
    // This is a multipiece named pipe write, is this the first
    // piece?
    //

    if ( (SmbGetUshort( &request->WriteMode ) &
          SMB_WMODE_START_OF_MESSAGE) != 0 ) ...

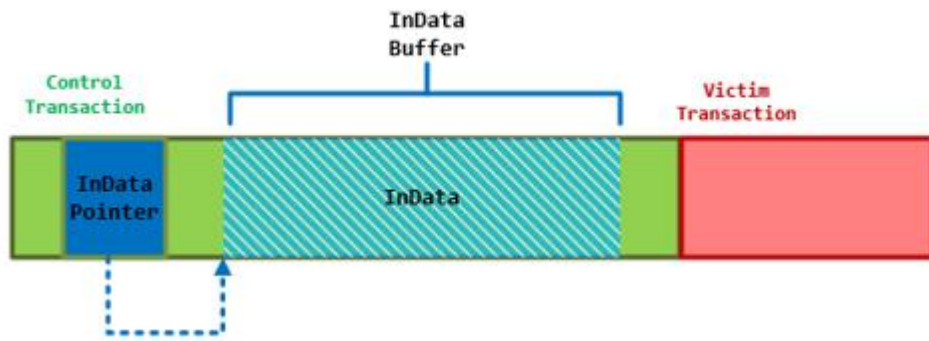
    transaction = SrvFindTransaction(
        connection,
        header,
        fid
    );
}
```

EternalRomance

CVE-2017-0143 – Type confusion between WriteAndX and Transaction requests

CVE-2017-0147 – Info disclosure in SrvPeekNamedPipe

Multiple WriteAndX requests can follow to fill the
Transaction buffer when WriteMode is in RAW_MODE...



```
RtlCopyMemory(transaction->InData, writeAddress, writeLength );  
  
//  
// Update the transaction data pointer to where the next  
// WriteAndX data buffer will go.  
//  
  
transaction->InData += writeLength;  
transaction->DataCount += writeLength;  
  
} // secondary piece of multipart write
```

...leading to increment InData and
DataCount in SrvSmbWriteAndX

EternalRomance

CVE-2017-0143 – Type confusion between WriteAndX and Transaction requests

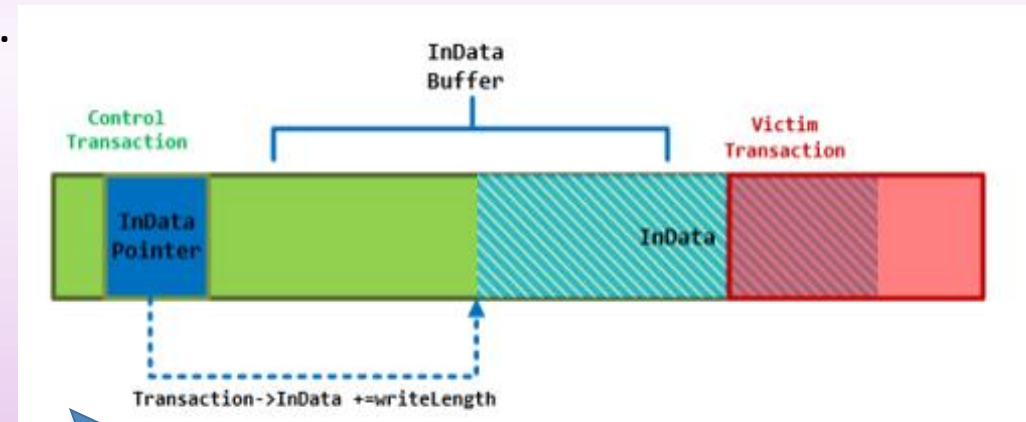
CVE-2017-0147 – Info disclosure in SrvPeekNamedPipe

But SrvSmbTransactionSecondary also calls SrvFindTransaction...

```
transaction = SrvFindTransaction( connection, header, 0 );
```

...and writes data to transaction->InData:

```
if ( dataCount != 0 ) {  
    RtlMoveMemory(  
        transaction->InData + dataDisplacement,  
        (PCHAR)header + dataOffset,  
        dataCount  
    );  
}
```



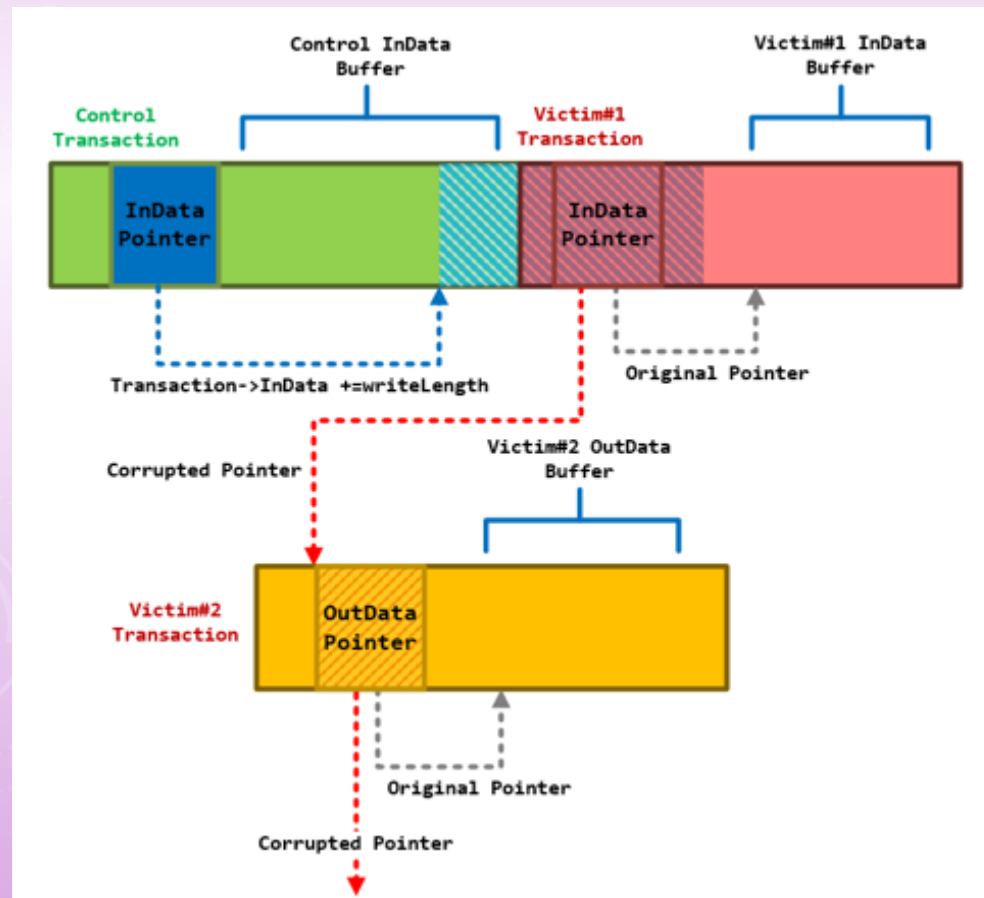
...leading to a buffer overflow on an adjacent Transaction object

EternalRomance

CVE-2017-0143 – Type confusion between WriteAndX and Transaction requests

CVE-2017-0147 – Info disclosure in SrvPeekNamedPipe

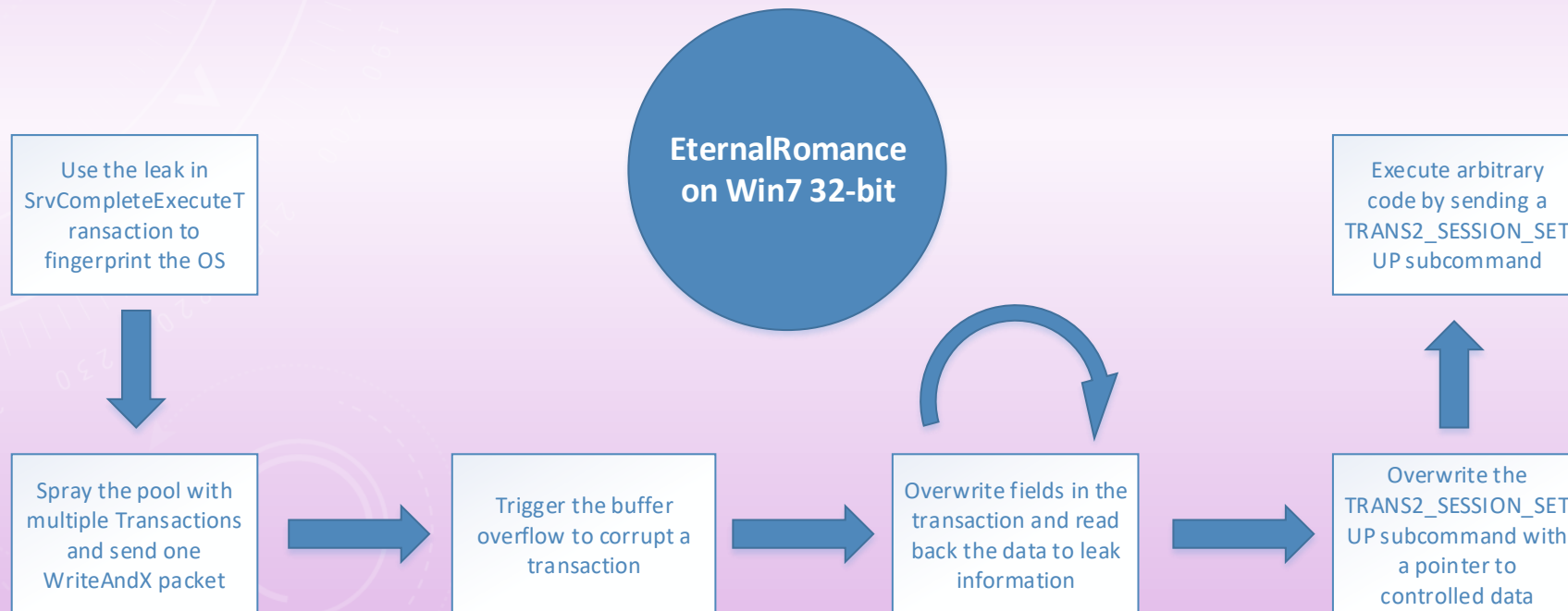
How to build an arbitrary read?



EternalRomance

CVE-2017-0143 – Type confusion between WriteAndX and Transaction requests

CVE-2017-0147 – Info disclosure in SrvPeekNamedPipe



EternalRomance in action

EternalSynergy

= [EternalChampion + EternalRomance]



Aug. 25-26 2017 in Taipei



EternalSynergy

CVE-2017-0143 – Type confusion between WriteAndX and Transaction requests

CVE-2017-0146 – Race condition with Transaction requests

29	1.520218	33.0.0.15	33.0.0.52	SMB	5382 Trans Request
34	1.520650	33.0.0.52	33.0.0.15	SMB	1494 NT Trans Response, NT RENAME
36	1.520953	33.0.0.52	33.0.0.15	SMB	93 Trans Response
37	1.520954	33.0.0.52	33.0.0.15	SMB	93 Trans Response
38	1.520955	33.0.0.52	33.0.0.15	SMB	93 Trans Response
39	1.520955	33.0.0.52	33.0.0.15	SMB	93 Trans Response
40	1.520956	33.0.0.52	33.0.0.15	SMB	93 Trans Response
41	1.520956	33.0.0.52	33.0.0.15	SMB	93 Trans Response
42	1.520957	33.0.0.52	33.0.0.15	SMB	93 Trans Response
43	1.520957	33.0.0.52	33.0.0.15	SMB	93 Trans Response
45	1.543609	33.0.0.52	33.0.0.15	SMB	474 NT Trans Response, <unknown>[Unassembled Packet]
47	1.545361	33.0.0.15	33.0.0.52		125 Trans Request

```
0020 00 0f 01 bd c4 0c 03 2d 7b d9 40 74 58 dc 50 18 .....- {.@tX.P.
0030 01 fd df 99 00 00 00 00 01 a0 ff 53 4d 42 a0 00 ..... ..SMB..
0040 00 00 00 98 03 c0 00 00 00 00 00 00 00 00 00 .....
0050 00 00 00 08 2d 51 00 08 6c a3 12 00 00 00 04 00 ....-Q.. l.....
0060 00 00 c0 10 00 00 00 00 00 00 48 00 00 00 04 00 ..... ..H....
0070 00 00 58 01 00 00 48 00 00 00 b8 10 00 00 00 59 ..X...H. ....Y
0080 01 00 96 b3 6c e9 d9 00 fa 07 00 00 00 00 00 00 ....l... ..
0090 00 00 00 00 00 00 00 00 00 00 04 02 03 46 72 ..... ..Fr
00a0 61 67 00 00 00 00 00 00 00 00 20 51 00 00 00 00 ag..... .. Q....
00b0 00 00 00 00 00 00 00 00 00 00 02 04 01 00 46 72 ..... ..Fr
00c0 65 65 00 00 00 00 00 00 00 00 01 04 eb 03 4c 53 ee..... ..LS
00d0 74 72 f0 c4 07 00 83 fa ff ff 8c 0e 00 00 00 00 tr..... ..
```

Race condition to leak a transaction object
Same exploit in EternalChampion

CVE-2017-0143 – Type confusion between WriteAndX and Transaction requests

CVE-2017-0146 – Race condition with Transaction requests

998	707.241651	33.0.0.15	33.0.0.52	SMB	630 Write AndX Request, FID: 0x4000,
999	707.241743	33.0.0.52	33.0.0.15	SMB	105 Write AndX Response, FID: 0x4000,
1001	708.257619	33.0.0.15	33.0.0.52	SMB	126 Trans Secondary Request
1003	708.305569	33.0.0.15	33.0.0.52	SMB	125 Trans Secondary Request
1004	708.305883	33.0.0.52	33.0.0.15	SMB	93 Trans Secondary Response, Error:
1005	708.308095	33.0.0.15	33.0.0.52	SMB	132 Trans Secondary Request
1007	708.372613	33.0.0.15	33.0.0.52	SMB	754 Trans Secondary Request

Parameter Displacement: 0
Data Count: 40
Data Offset: 66
Data Displacement: 136
Byte Count (BCC): 55
Padding: 00000000000000000000000000000000
Extra byte parameters: 80b82f0183faffff0400000000

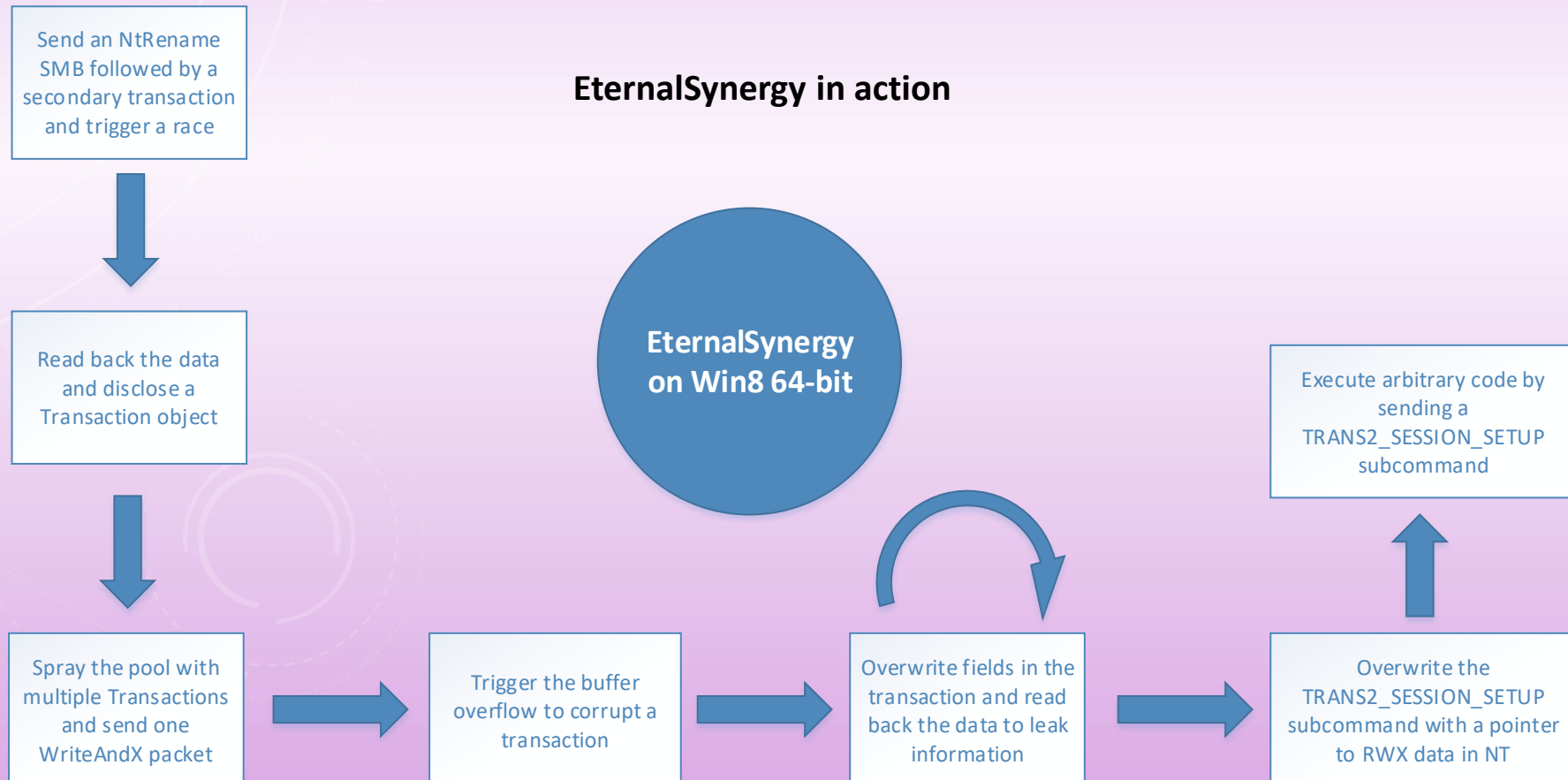
Several Transactions exchanged
Same exploit in EternalRomance

The main difference is the choice of an RWX section in NT to write the payload...

EternalSynergy

CVE-2017-0143 – Type confusion between WriteAndX and Transaction requests

CVE-2017-0146 – Race condition with Transaction requests



None of these worked on Windows 10...




Aug. 25-26 2017 in Taipei



Mitigations blocking these exploits

- Null sessions have disappeared by default since Windows 8
- Full ASLR, the HAL heap region is now randomized starting from RS2
- Kernel NX since Windows 8, making the HAL heap region and the non paged pool allocations used in srv.sys non executable
- Kernel CFG (kCFG) with HVCI enabled since RS2 prevents indirect calls to hijack the flow

```
movzx    eax, si
mov      rcx, rdi
call     rva SrvTransaction2DispatchTable[rdx+rax*8]
```



```
movzx    eax, di
mov      rax, ds:rva SrvTransaction2DispatchTable[rcx+rax*8]
mov      rcx, rbx
call     cs:__guard_dispatch_icall_ptr
```

- RWX areas in ntoskrnl have now disappeared
- **Windows 10 RS3 (Fall Creators Update) and Windows Server 2016 RS3 have SMB1 uninstalled by default under most circumstances**

And what else?



Aug. 25-26 2017 in Taipei



Pool overflow in SrvSmbCreateWithExtraOptions

```
SMB_TRANS_STATUS  
SrvSmbCreateWithExtraOptions (  
    IN OUT PWORK_CONTEXT WorkContext  
)
```

```
if (transaction->Function != NT_TRANSACTION_CREATE ||  
    transaction->ParameterCount < sizeof(REQ_CREATE_WITH_SD_OR_EA) ||  
    transaction->MaxParameterCount < sizeof(RES_CREATE_WITH_SD_OR_EA) ) {
```

```
if ( extendedRequested ) {  
    NTSTATUS ExtendedResponseStatus;  
  
    PRESP_EXTENDED_CREATE_WITH_SD_OR_EA ExtendedResponse;  
  
    ExtendedResponse = (PRESP_EXTENDED_CREATE_WITH_SD_OR_EA)response;  
  
    RtlZeroMemory( ((PVOID)&ExtendedResponse->VolumeGuid[0]), sizeof(ExtendedResponse->VolumeGuid)+sizeof(ExtendedResponse->FileId) );
```

**Extended Requests in SrvSmbCreateWithExtraOptions were not properly checked
leading to pool overflow**

Pool overflow in SrvCompleteExecuteTransaction

```
outputBufferSize = ((maxParameterCount * sizeof(CHAR) + 3) & ~3) +  
                    ((maxDataCount * sizeof(CHAR) + 3) & ~3);  
  
if ( sizeof(SMB_HEADER) +  
      sizeof (RESP_TRANSACTION) +  
      sizeof(USHORT) * request->SetupCount +  
      sizeof(USHORT) +  
      outputBufferSize  
      <= MIN( (ULONG)session->MaxBufferSize, SrvReceiveBufferLength ) ) {  
    outputBufferSize = 0;  
}
```

```
if ( transaction->OutParameters == NULL ) {  
  
    //  
    // Parameters will go into the SMB buffer. Calculate the pointer  
    // then round it up to the next DWORD address.  
    //  
  
    transaction->OutParameters = (PCHAR)(transaction->OutSetup +  
                                          transaction->MaxSetupCount);  
}
```

Confusion due to using SetupCount at one point, and MaxSetupCount at another, leading to pool overflow

OOB Write in RestartWriteNamedPipe

```
transaction = WorkContext->Parameters.Transaction;

transaction->SetupCount = 0;
transaction->ParameterCount = 2;
transaction->DataCount = 0;

SmbPutUshort( (PSMB_USHORT)transaction->OutParameters,
              (USHORT)iosb->Information
              );

SrvCompleteExecuteTransaction(WorkContext, SmbTransStatusSuccess);

IF_DEBUG(TRACE2) SrvPrint0( "RestartWriteNamedPipe complete\n" );
return;

} // RestartWriteNamedPipe
```

RestartWriteNamedPipe was not checking whether there was enough space in OutParameters to write iosb->Information

Abusing SrvSmbNtRename

```
{  
    IF_DEBUG	TRACE2) KdPrint(( "SrvSmbNtRename complete.\n" ));  
  
    //  
    // Dead code path. Fail to prevent use in exploits  
    //  
    SrvSetSmbError(WorkContext, STATUS_SMB_BAD_COMMAND);  
  
    return SmbTransStatusErrorWithoutData;  
} // SrvSmbNtRename
```

SrvSmbNtRename was a noop, making exploits easier (especially combined with the next issue). It now returns an error.

Uninitialized memory in SrvSmbTransactionSecondary

```
if ( parameterCount != 0 ) {  
    RtlMoveMemory(  
        transaction->InParameters + parameterDisplacement,  
        (PCHAR)header + parameterOffset,  
        parameterCount  
    );  
}  
  
if ( dataCount != 0 ) {  
    RtlMoveMemory(  
        transaction->InData + dataDisplacement,  
        (PCHAR)header + dataOffset,  
        dataCount  
    );  
}
```

Abusing dataDisplacement in SrvSmbTransactionSecondary or SrvSmbNtTransactionSecondary could lead to create a Transaction with dataCount = TotalDataCount but leaving the buffer uninitialized

Uninitialized memory in SrvSnapEnumerateSnapShots

```
PSRV_SNAPSHOT_ARRAY SnapShotArray = (PSRV_SNAPSHOT_ARRAY)transaction->OutData;

SnapShotArray->NumberOfSnapShots = SnapShotCount;
SnapShotArray->SnapShotArraySize = SNAPSHOT_NAME_LENGTH*SnapShotArray->NumberOfSnapShots+sizeof(WCHAR);
if( (SnapShotCount == 0) || (transaction->MaxDataCount < SnapShotArray->SnapShotArraySize + FIELD_OFFSET(SRV_SNAPSHOT_ARRAY, SnapShotMultisZ)) )
{
    // The buffer is not big enough. Return the required size
    SnapShotArray->NumberOfSnapShotsReturned = 0;
    transaction->DataCount = sizeof(SRV_SNAPSHOT_ARRAY);
    Status = STATUS_SUCCESS;
}
else
```

```
typedef struct _SRV_SNAPSHOT_ARRAY
{
    ULONG NumberOfSnapShots;           //
    ULONG NumberOfSnapShotsReturned;   //
    ULONG SnapShotArraySize;           //
    WCHAR SnapShotMultisZ[1];          //
} SRV_SNAPSHOT_ARRAY, *PSRV_SNAPSHOT_ARRAY;
```

SrvSnapEnumerateSnapShots was leaking bytes when SnapShotCount = 0

Uninitialized memory in ProcessOs2Ioctl

```
if ( ansiShare.Length >= LM20_NNLEN ) {  
    RtlCopyMemory(  
        buffer,  
        ansiShare.Buffer,  
        LM20_NNLEN  
    );  
} else {  
    RtlCopyMemory(  
        buffer,  
        ansiShare.Buffer,  
        ansiShare.Length  
    );  
    buffer[ansiShare.Length] = '\\0';  
}
```

0030	02 01 e3 ce 00 00 00 00	00 58 ff 53 4d 42 32 00X.SMB2.
0040	00 00 00 98 01 48 00 00	00 00 00 00 00 00 00 00H..
0050	00 00 00 08 e0 2f 00 08	41 41 0a 00 00 20 00 00/.. AA.... ..
0060	00 00 00 38 00 00 00 20	00 38 00 00 00 00 00 21	...8... .8.....!
0070	00 00 00 00 4a 4f 48 41	20 50 43 20 20 20 20 20	...JOHN -PC
0080	20 20 20 00 49 50 43 21	00 65 48 79 24 85 48 79	.IPC\$.eHy\$.Hy
0090	00 85		..

ProcessOs2Ioctl was leaking bytes when ansiShare.length < LM20_NNLEN

Return values not correctly initialized in SrvSmbQuerySecurityDescriptor

```
SmbPutUlong( &response->LengthNeeded, lengthNeeded );
transaction->ParameterCount = sizeof( RESP_QUERY_SECURITY_DESCRIPTOR );

//
// If an error occurred, return an appropriate response.
//

if ( !NT_SUCCESS(status) ) {

    transaction->ParameterCount = transaction->MaxParameterCount;
    transaction->DataCount = 0;
    SrvSetSmbError2( WorkContext, status, TRUE );
    return SmbTransStatusErrorWithData;
} else {
    transaction->DataCount =
        RtlLengthSecurityDescriptor( transaction->OutData );
}

return SmbTransStatusSuccess;
} // SrvSmbQuerySecurityDescriptor
```

SrvSmbQuerySecurityDescriptor was not resetting some fields before returning

Return values not correctly initialized in RestartNtloctl

```
if ( transaction->MaxSetupCount > 0 ) {  
    transaction->SetupCount = 1;  
    SmbPutUshort( transaction->OutSetup, (USHORT)length );  
}  
  
transaction->ParameterCount = transaction->MaxParameterCount;  
transaction->DataCount = length;  
  
if ( !NT_SUCCESS(status) ) {
```

RestartNtloctl was not resetting some fields before returning

Return values not correctly initialized in RestartCallNamedPipe

```
if ( status == STATUS_BUFFER_OVERFLOW ) {  
  
    //  
    // Down level clients, expect us to return STATUS_SUCCESS.  
    //  
  
    if ( !IS_NT_DIALECT( WorkContext->Connection->SmbDialect ) ) {  
        status = STATUS_SUCCESS;  
  
    } else {  
  
} else if ( !NT_SUCCESS(status) ) ...  
  
    //  
    // Success. Prepare to generate and send the response.  
    //  
  
    transaction->SetupCount = 0;  
    transaction->ParameterCount = 0;  
    transaction->DataCount = (ULONG)WorkContext->Irp->IoStatus.Information;  
  
}
```

RestartCallNamedPipe

Is that all?



Aug. 25-26 2017 in Taipei



Enjoyed the talk? Join us!



Axel Souchet @Overcl0k · Aug 10

The MSRC Vulnerabilities & Mitigations team is hiring a security engineer, come join us! careers.microsoft.com/jobdetails.asp...



1



45



36



Nicolas Joly @n_joly · May 23

MSRC-UK is expanding again! If you want to play with bugz and 0dayz that's definitely the place to be. DM for fancy detailz :-)



29



22

HITCON 

Aug. 25-26 2017 in Taipei

THANKS!

