MS17-010?

Evolution of Exploitation and Windows Security

zerosum0x0

RISKSENSE

DerbyCon VIII EVOLUTION

LOUISVILLE, KY - 2018

MS17-010 in 2018?

- … “in 2017?”
  - 30-day, reliable, remote exploits for Windows SMB
- It still works! (like MS08-067/MS09-050/MS10-061)
  - Sophistication of worms
  - 0xAmit: Last month, 919k external vulnerable hosts
- Case studies
  - Old/new Windows kernel exploitation techniques
  - Bypassing mitigations (ASLR/DEP/Cookies/KPP/etc)
  - As well as, pwning exploited driver’s own structures

https://www.cybereason.com/blog/wannamine-cryptominer-eternalblue-wannacry

Outline

- Background
  - SMBv1 Internals
  - ~~Shadow Brokers/Equation Group Saga~~
- Exploit Chains (XP, Win7, Win10)
  - EternalBlue
  - ~~EternalChampion~~
  - ~~EternalRomance~~
  - ~~EternalSynergy~~
- Patch Info
- Payloads
  - DoublePulsar
  - ~~DanderSpritz/PeddleCheap/KillSuit~~
- Defeated and Future Mitigations

msuiche - Lessons from TheShadowBrokers One Year Later, *SANS DFIR Prague*
https://www.sans.org/event/dfir-prague-2018/summit-agenda

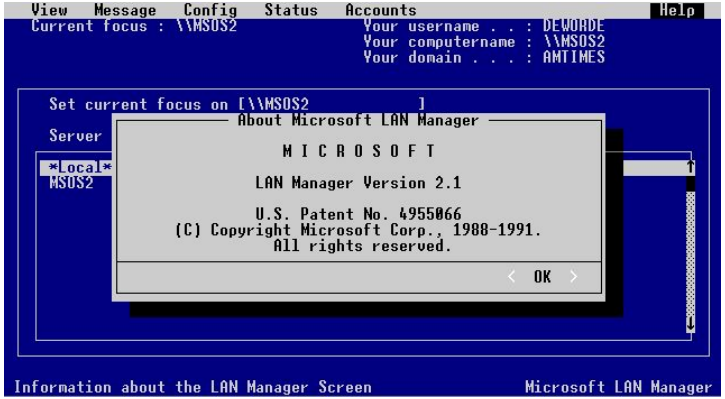
zerosum0x0 - Eternal Exploits, DEF CON 26
https://preview.tinyurl.com/eedefcon

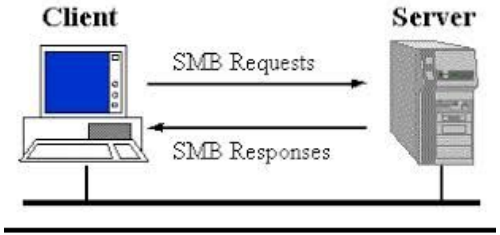
franciscrs - DerbyCon VIII - Sunday 11:00 Track 3

Background

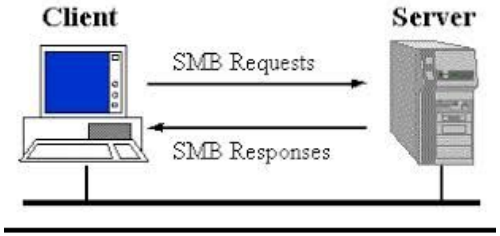
A Brief History of SMBv1

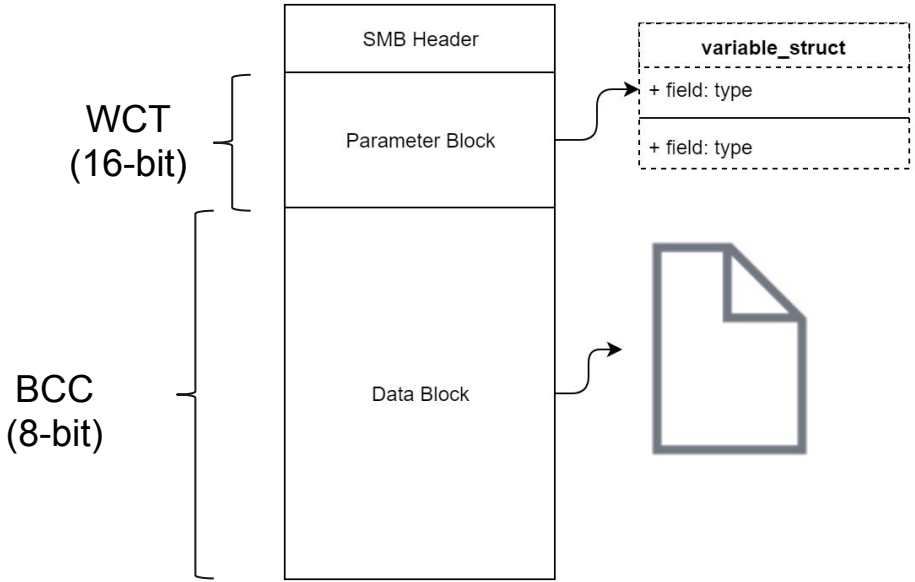
- 1983
  - BAF/SMB (Barry Feigenbaum, IBM)
- 1988
  - LAN Manager for OS/2 (Microsoft/IBM)
  - NT kernel (Dave Cutler, Microsoft)
- 1989-1991
  - Srv.sys (Microsoft)
    - Most ETERNAL mechanics introduced
- 1997-1999?
  - ETERNALBLUE vuln introduced
    - Win2000 Pre-release RC 2?
    - Not in NT4 SP6a SRP
- 2017
  - MS17-010/FuzzBunch





Server Message Block

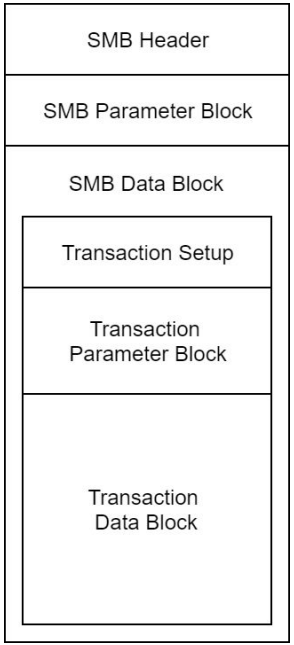


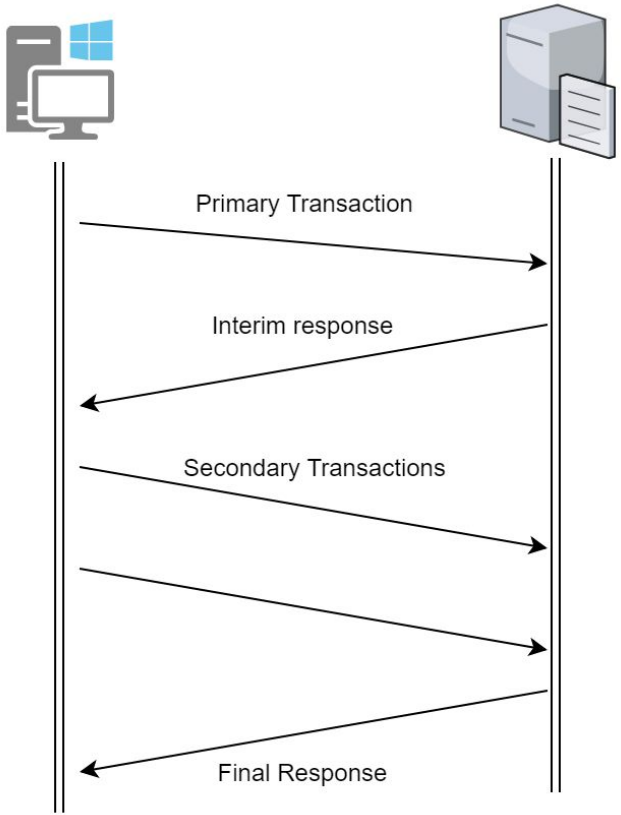



```
SMB_Header
{
    UCHAR       Protocol[4];

    UCHAR       Command;

    SMB_ERROR   Status;

    UCHAR       Flags;
    USHORT      Flags2;

       /* … */
}
```

Transaction Lifecycle

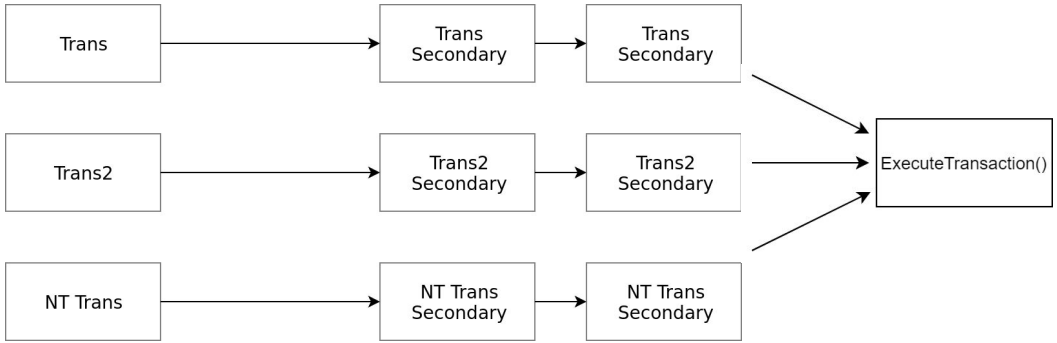
- SMB in an SMB over many SMB
  - Primary Transaction SMB
  - Secondary Transaction SMB(s)
- Transactions IOCTL
  - Mostly filesystem
- Think "database transactions"
  - Performed mostly atomic
  - Waits for **all** parts/data



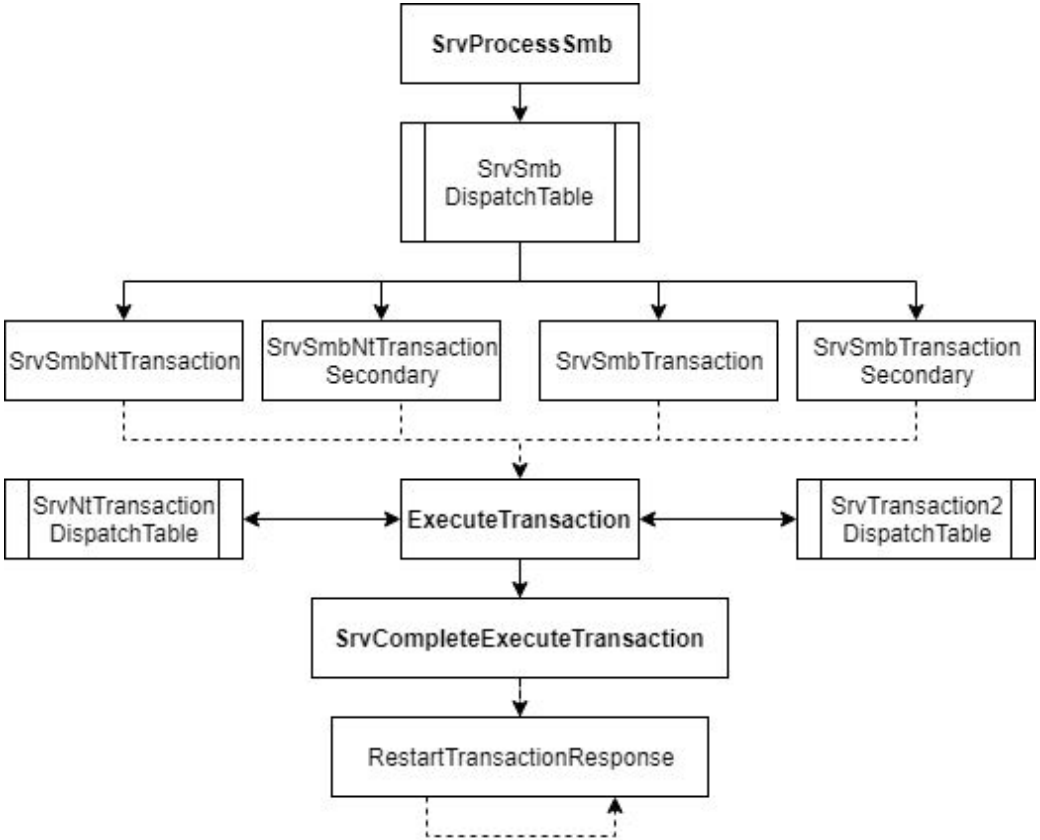


Transaction Types

- Trans (Trans1)
  - Mailslots
  - MS-RAP
- Trans2
  - >8.3 DOS shortnames
  - OS/2 to NT file stuff
- NT Trans
  - Transaction Parameter/Data sizes
  - USHORT -> ULONG



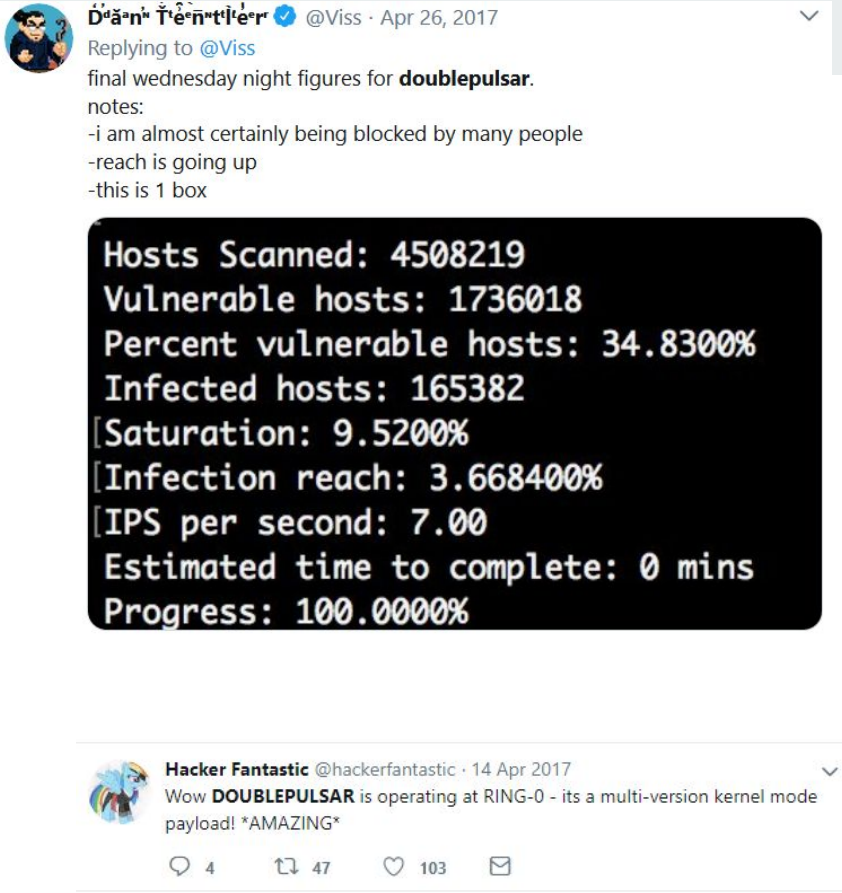
Srv.sys
Dispatch Tables



DOUBLEPULSAR

- Ping
- Exec
- Kill

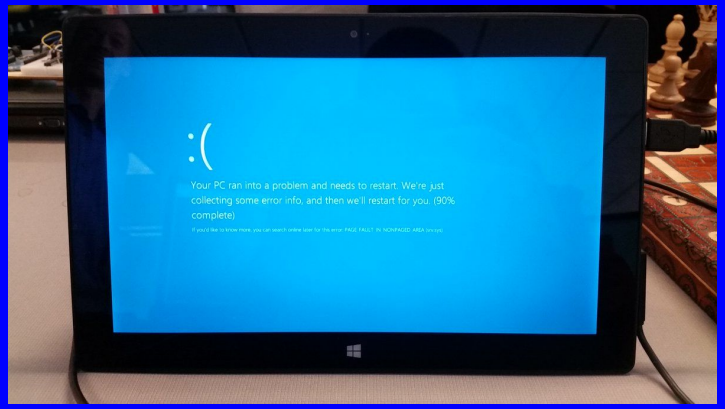
```
kd> dps srv!SrvTransaction2DispatchTable
fffff880`03ad8760  fffff880`03b42780 srv!SrvSmbOpen2
fffff880`03ad8768  fffff880`03b09b20 srv!SrvSmbFindFirst2
fffff880`03ad8770  fffff880`03b3ff40 srv!SrvSmbFindNext2
fffff880`03ad8778  fffff880`03b0b650 srv!SrvSmbQueryFsInformation
fffff880`03ad8780  fffff880`03b34d20 srv!SrvSmbSetFsInformation
fffff880`03ad8788  fffff880`03b09670 srv!SrvSmbQueryPathInformation
fffff880`03ad8790  fffff880`03b42cb0 srv!SrvSmbSetPathInformation
fffff880`03ad8798  fffff880`03b07420 srv!SrvSmbQueryFileInformation
fffff880`03ad87a0  fffff880`03b08080 srv!SrvSmbSetFileInformation
fffff880`03ad87a8  fffff880`03b29660 srv!SrvSmbFsctl
fffff880`03ad87b0  fffff880`03b42ae0 srv!SrvSmbIoctl2
fffff880`03ad87b8  fffff880`03b29660 srv!SrvSmbFsctl
fffff880`03ad87c0  fffff880`03b29660 srv!SrvSmbFsctl
fffff880`03ad87c8  fffff880`03b354f0 srv!SrvSmbCreateDirectory2
fffff880`03ad87d0  fffffa80`04682060
fffff880`03ad87d8  fffff880`03b29460 srv!SrvTransactionNotImplemented
```



https://www.countercept.com/blog/analyzing-the-doublepulsar-kernel-dll-injection-technique/
https://zerosum0x0.blogspot.com/2017/04/doublepulsar-initial-smb-backdoor-ring.html

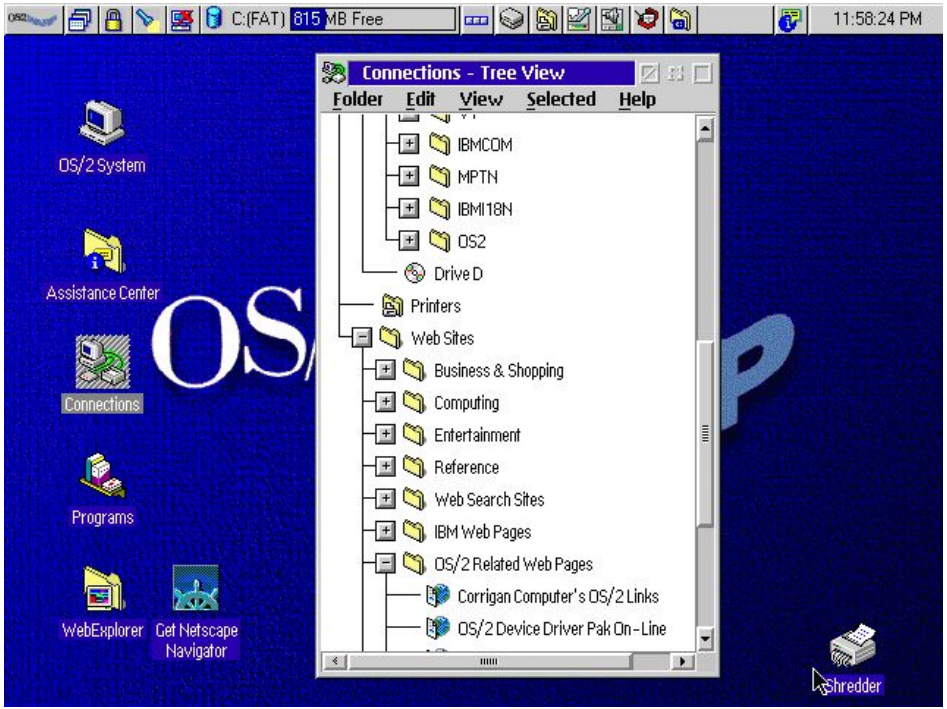
ETERNALBLUE

v2.2.0



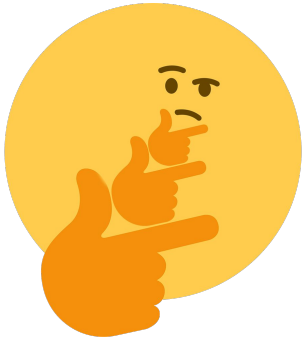
Extended Attributes

- Filesystem hidden Name/Value key-pair
  - Metadata attached to files
- OS/2 v1.2
  - Joint Microsoft/IBM OS
  - HPFS
- Windows NT
  - NTFS
  - Alternate Data Streams
  - WSL
- FEA vs. GEA
  - FEA = name+value
  - GEA = name



Alternative Fact

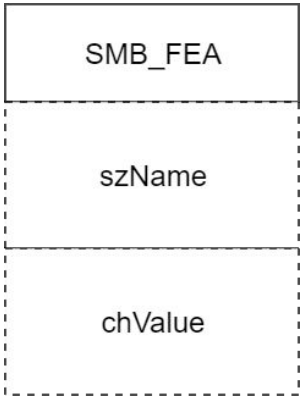
- Most theories suggest EternalBlue = BSOD
- But, what if: a reference to IBM?
    - OS/2 FEA -> NT FEA



OS/2 FEA

```
struct SMB_FEA
{
     BYTE ExtendedAttributeFlag;            // 0x0 or 0x80
     BYTE AttributeNameLengthInBytes;
     WORD AttributeValueLengthInBytes;



};
```

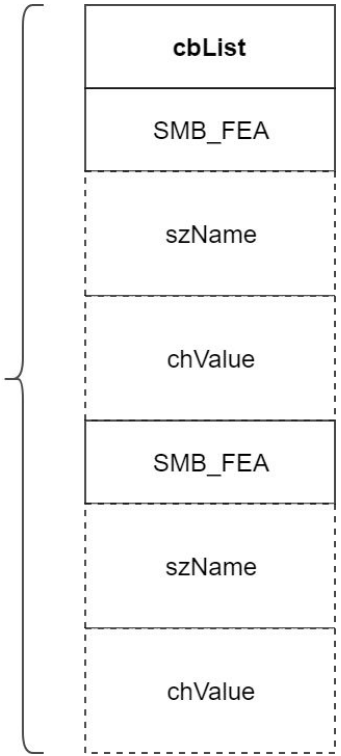
OS/2 FEA




```
struct SMB_FEA
{
     BYTE ExtendedAttributeFlag;           // 0x0 or 0x80
     BYTE AttributeNameLengthInBytes;
     WORD AttributeValueLengthInBytes;

     // CHAR   szName[AttributeNameLengthInBytes + 1];   // C str
     // BYTE   chValue[AttributeValueLengthInBytes];// no null-terminator
};
```

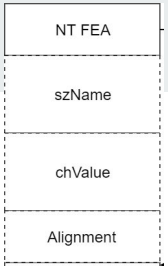
OS/2 FEA List

```
struct SMB_FEA_LIST
{
     ULONG     SizeOfListInBytes;          // cbList
     SMB_FEA   List[];
};
```



NT FEA

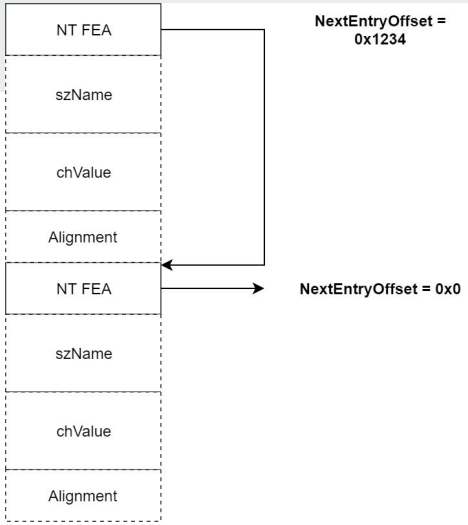
```
struct FILE_FULL_EA_INFORMATION
{
     ULONG     NextEntryOffset;
     UCHAR     Flags;                    // 0x0 or 0x80
     UCHAR     EaNameLength;
     USHORT    EaValueLength;



};
```



NT FEA

```
struct FILE_FULL_EA_INFORMATION
{
     ULONG     NextEntryOffset;
     UCHAR     Flags;                          // 0x0 or 0x80
     UCHAR     EaNameLength;
     USHORT    EaValueLength;

     // CHAR   EaName[EaNameLength + 1];       // C str
     // BYTE   EaValue[EaValueLength];         // no null-terminator
     // BYTE   Alignment[+3 & ~3];             // align DWORD
};
```

NT FEA(LIST)




```
struct FILE_FULL_EA_INFORMATION
{
    ULONG     NextEntryOffset;          // Parse list until == 0
    UCHAR     Flags;                    // 0x0 or 0x80
    UCHAR     EaNameLength;
    USHORT    EaValueLength;

    // CHAR   EaName[EaNameLength + 1];  // C str
    // BYTE   EaValue[EaValueLength];    // no null-terminator
    // BYTE   Alignment[+3 & ~3];        // align DWORD
};
```

(NULL) FEA Conversion OS/2 -> NT

```
struct SMB_FEA
{
      BYTE  Flags;
      BYTE  cbName;
      WORD  cbValue;
      CHAR  szName[cbName + 1];
      BYTE  chValue[cbValue];
};
```

```
Flags         =     0x00
cbName        =     0x00
cbValue       =     0x0000
szName        =     '\0'
```

"\x00" * 5

(NULL) FEA Conversion OS/2 -> NT

```
struct SMB_FEA
{
      BYTE   Flags;
      BYTE   cbName;
      WORD   cbValue;
      CHAR   szName[cbName + 1];
      BYTE   chValue[cbValue];
};
```

```
Flags         =      0x00
cbName        =      0x00
cbValue       =      0x0000
szName        =      '\0'
```

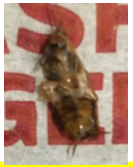
"\x00" * 5

```
struct FILE_FULL_EA_INFORMATION
{
      ULONG        NextEntryOffset;
      UCHAR        Flags;
      UCHAR        EaNameLength;
      USHORT       EaValueLength;
      CHAR         EaName[EaNameLength];
      BYTE         EaValue[EaValueLength];
      BYTE         Alignment[+3 & ~3];
};
```

```
NextEntryOffset      =      0x00000000
Flags                =      0x00
EaNameLength         =      0x00
EaValueLength        =      0x0000
EaName               =      '\0'
Alignment            =      9 + 3 & ~3 = 0x000000
```

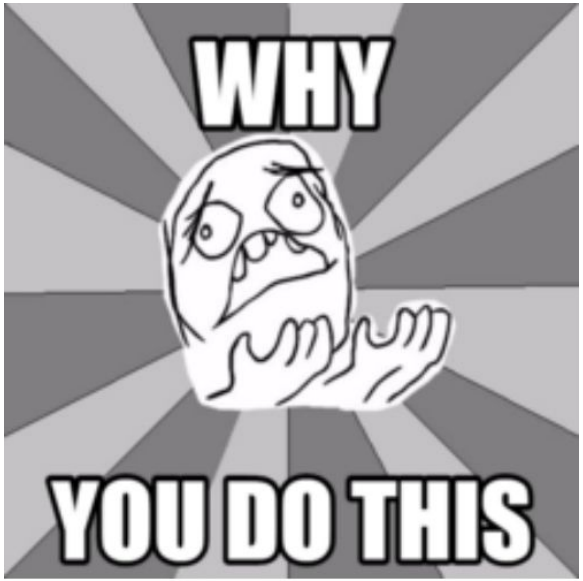
"\x00" * 12

```
ULONG SrvOs2FeaListSizeToNt(FEALIST *FeaList)
{
    lastValidLocation = FeaList + FeaList->cbList;
    fea = FeaList->list;
    ntBufferSize = 0;

    while (fea < lastValidLocation) {
        feaSize = fea->cbName + 1 + fea->cbValue;
        if (fea + feaSize > lastValidLocation) {
            SmbPutUshort(&FeaList->cbList, PTR_DIFF_SHORT(fea, FeaList));
            break;
        }
        ntBufferSize += FEA_SIZE(fea);
        fea = NEXT_FEA(fea);
    }

    return ntBufferSize;
}
```



SrvOs2FeaListSizeToNt()

- Iterate over each SMB_FEA
  - Accumulate size needed for NT buffer (conversion)
- If (`start(FEA) + size(FEA) > cbList`)
  - “Correct” `FEALIST.cbList`
    - **WHY NOT JUST REJECT THE SMB??**
- 2 outputs
  - Updated `FEALIST.cbList`
  - Returns accumulated NT buffer size



Bug #1 - CVE-2017-0144

- Incorrect pointer cast, store of WORD (16-bit) into DWORD (32-bit) value

```
ULONG FEALIST.cbList;
SmbPutUshort(&FeaList->cbList, PTR_DIFF_SHORT(fea, FeaList));
```

Bug #1 - CVE-2017-0144

- Incorrect pointer cast, store of WORD (16-bit) into DWORD (32-bit) value

```
ULONG FEALIST.cbList;
SmbPutUshort(&FeaList->cbList, PTR_DIFF_SHORT(fea, FeaList));
```

	HIDWORD	LODWORD
Attacker	0001	0000
Valid Size		
Vuln Size		
NT Buffer Size		

Bug #1 - CVE-2017-0144

- Incorrect pointer cast, store of WORD (16-bit) into DWORD (32-bit) value

```
ULONG FEALIST.cbList;
SmbPutUshort(&FeaList->cbList, PTR_DIFF_SHORT(fea, FeaList));
```

	HIDWORD	LODWORD
Attacker	0001	0000
Valid Size	0000	e627
Vuln Size		
NT Buffer Size		

Bug #1 - CVE-2017-0144

- Incorrect pointer cast, store of WORD (16-bit) into DWORD (32-bit) value

```
ULONG FEALIST.cbList;
SmbPutUshort(&FeaList->cbList, PTR_DIFF_SHORT(fea, FeaList));
```

	HIDWORD	LODWORD
Attacker	0001	0000
Valid Size	0000	e627
Vuln Size	0001	e627
NT Buffer Size		

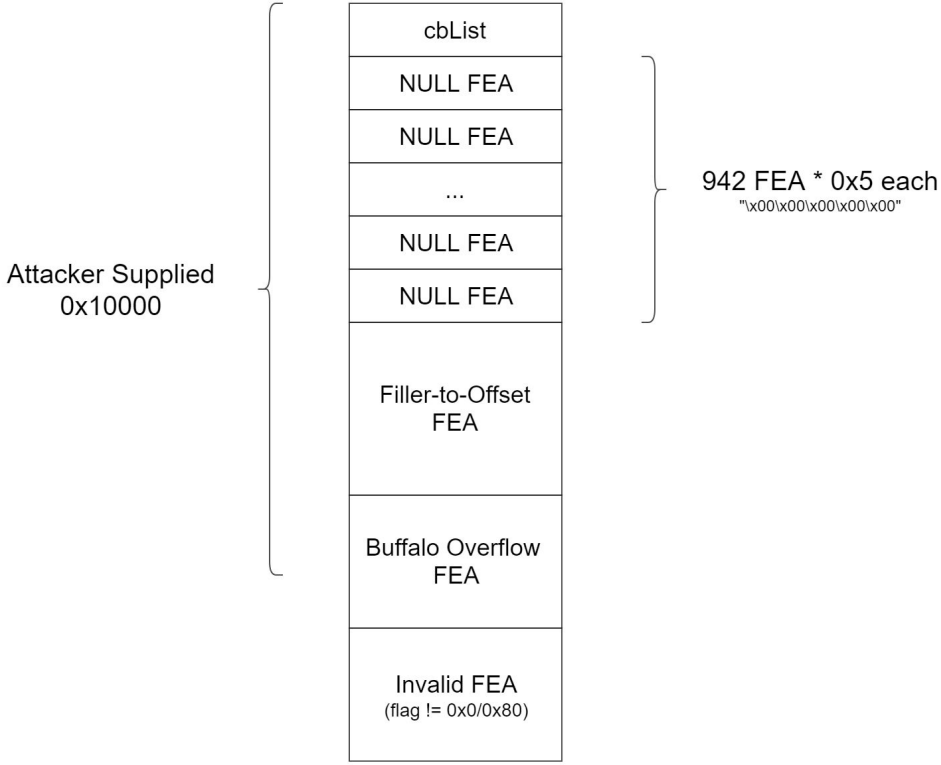
Bug #1 - CVE-2017-0144

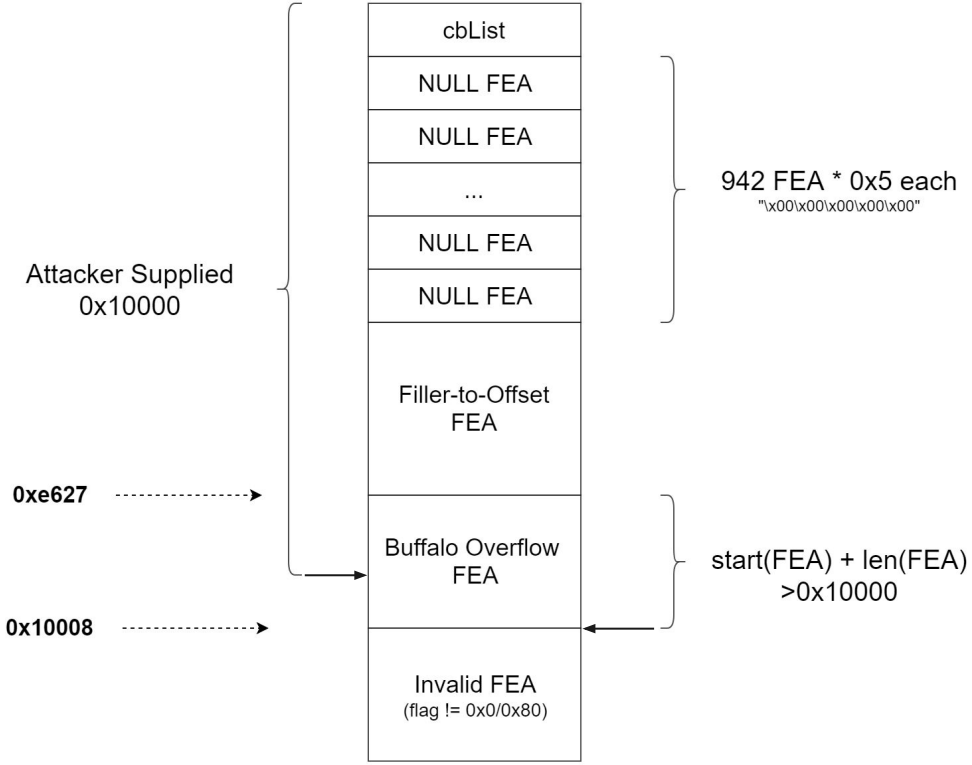
- Incorrect pointer cast, store of WORD (16-bit) into DWORD (32-bit) value

```
ULONG FEALIST.cbList;
SmbPutUshort(&FeaList->cbList, PTR_DIFF_SHORT(fea, FeaList));
```

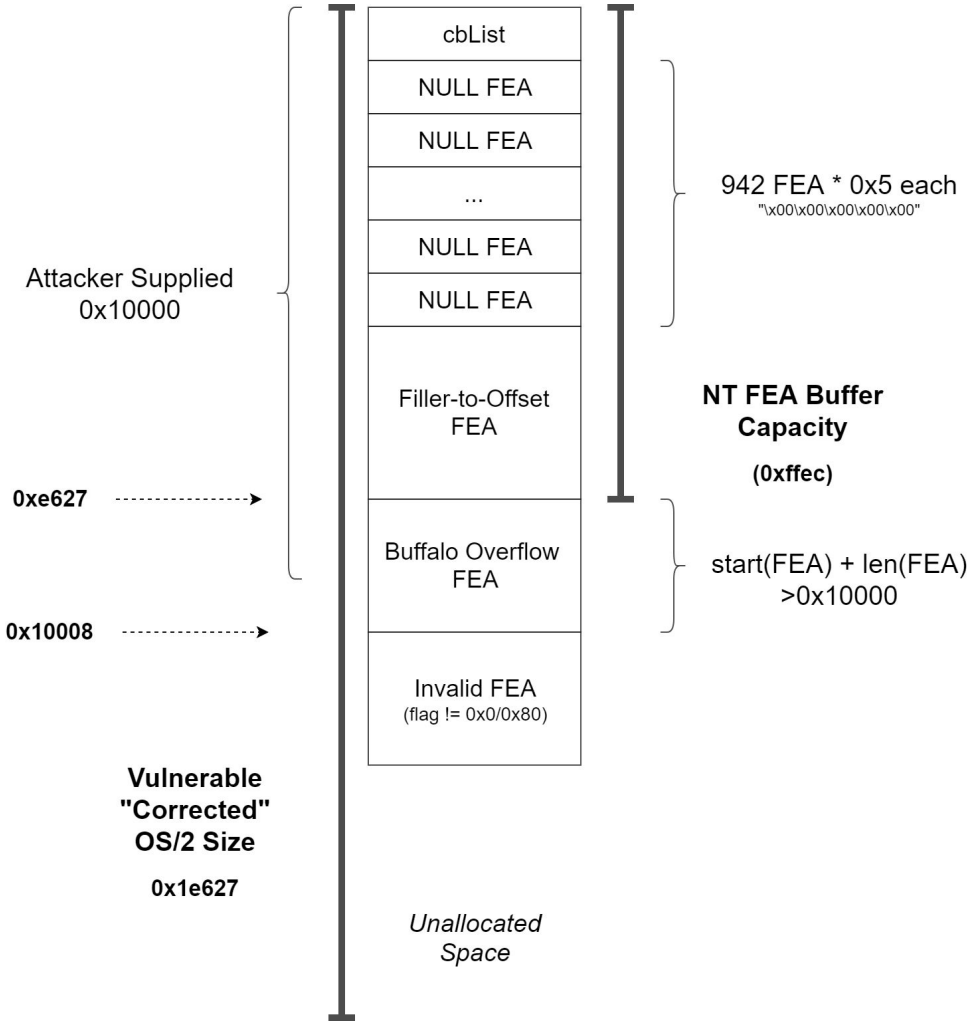
	HIDWORD	LODWORD
Attacker	0001	0000
Valid Size	0000	e627
Vuln Size	0001	e627
NT Buffer Size	0000	ffec

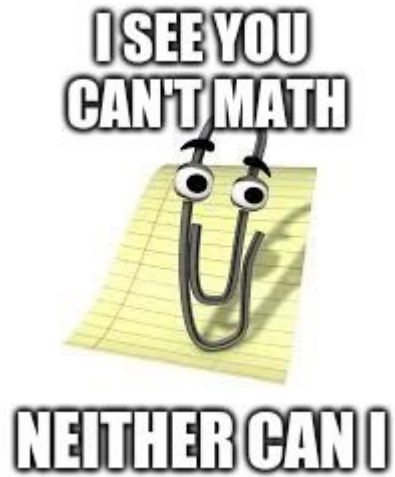
```
E627  ==  FFEC

1E627  >  FFEC
```

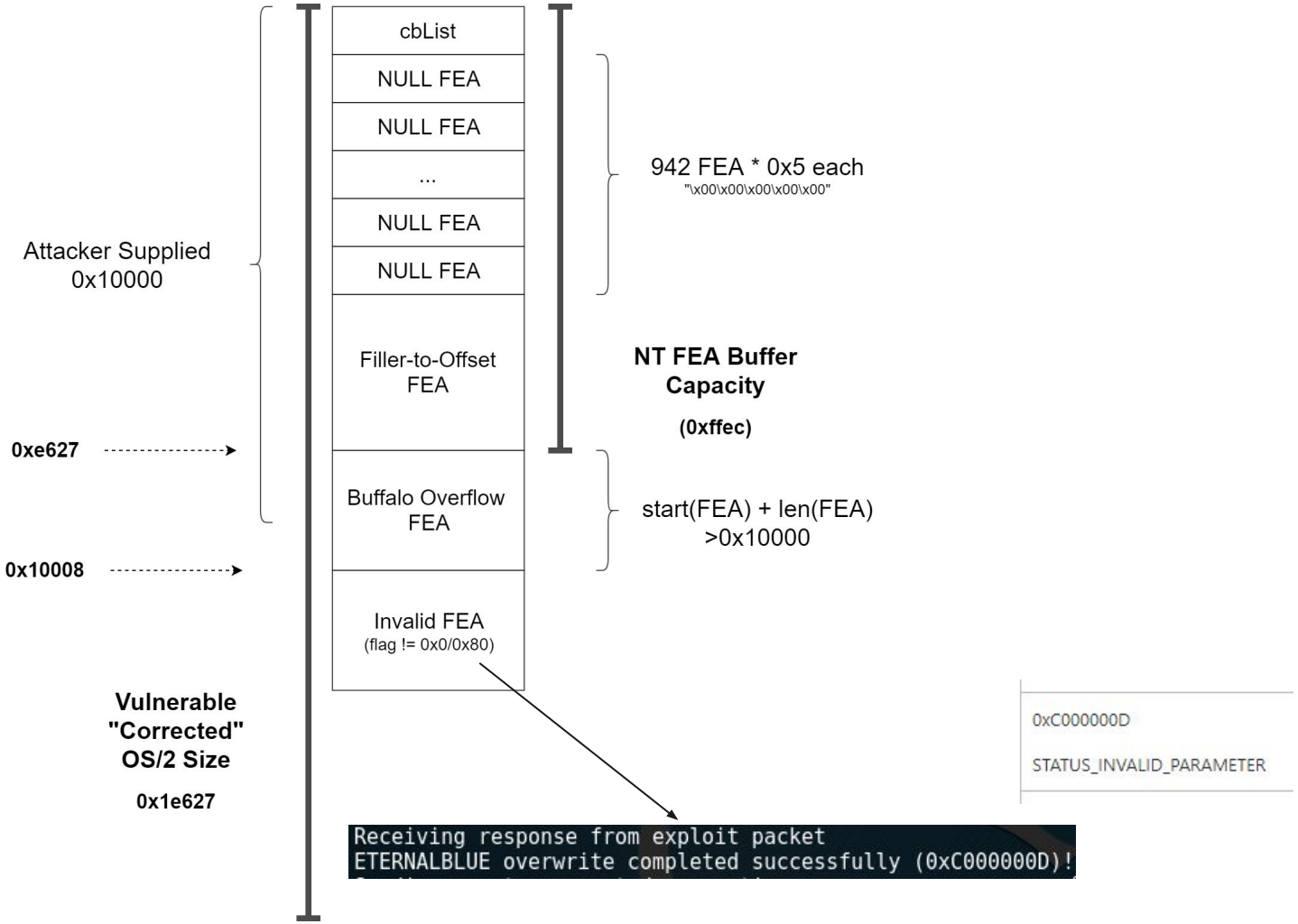

cbList
NULL FEA
NULL FEA
...
NULL FEA
NULL FEA
Filler-to-Offset FEA
Buffalo Overflow FEA
Invalid FEA
(flag != 0x0/0x80)
Attacker Supplied 0x10000
942 FEA * 0x5 each
"\x00\x00\x00\x00\x00"
Unallocated Space



Unallocated Space


cbList
NULL FEA
NULL FEA
...
NULL FEA
NULL FEA
Filler-to-Offset FEA
Buffalo Overflow FEA
Invalid FEA
(flag != 0x0/0x80)
Attacker Supplied 0x10000
0xe627
0x10008
Vulnerable "Corrected" OS/2 Size
0x1e627
Unallocated Space
942 FEA * 0x5 each
"\x00\x00\x00\x00\x00"
NT FEA Buffer Capacity
(0xffec)
start(FEA) + len(FEA) >0x10000


I SEE YOU
CAN'T MATH
NEITHER CAN I


cbList
NULL FEA
NULL FEA
...
NULL FEA
NULL FEA
Filler-to-Offset FEA
Buffalo Overflow FEA
Invalid FEA
(flag != 0x0/0x80)
Attacker Supplied 0x10000
0xe627
0x10008
Vulnerable "Corrected" OS/2 Size
0x1e627
942 FEA * 0x5 each
"\x00\x00\x00\x00\x00"
NT FEA Buffer Capacity
(0xffec)
start(FEA) + len(FEA) >0x10000
Receiving response from exploit packet
ETERNALBLUE overwrite completed successfully (0xC000000D)!
0xC000000D
STATUS_INVALID_PARAMETER

Assembly Analysis

x86/x64

```
srv!SrvOs2FeaListSizeToNt+0x47:
b6bb3c53 2bcf            sub     ecx,edi
b6bb3c55 66890f          mov     word ptr [edi],cx
```

```
srv!SrvOs2FeaListSizeToNt+0x71:
fffff880`039d5931 662bde          sub     bx,si
fffff880`039d5934 66891e          mov     word ptr [rsi],bx
```

ARM

```
loc_2BC9E
SUBS            R3, R6, R5
STRH            R3, [R5]
```

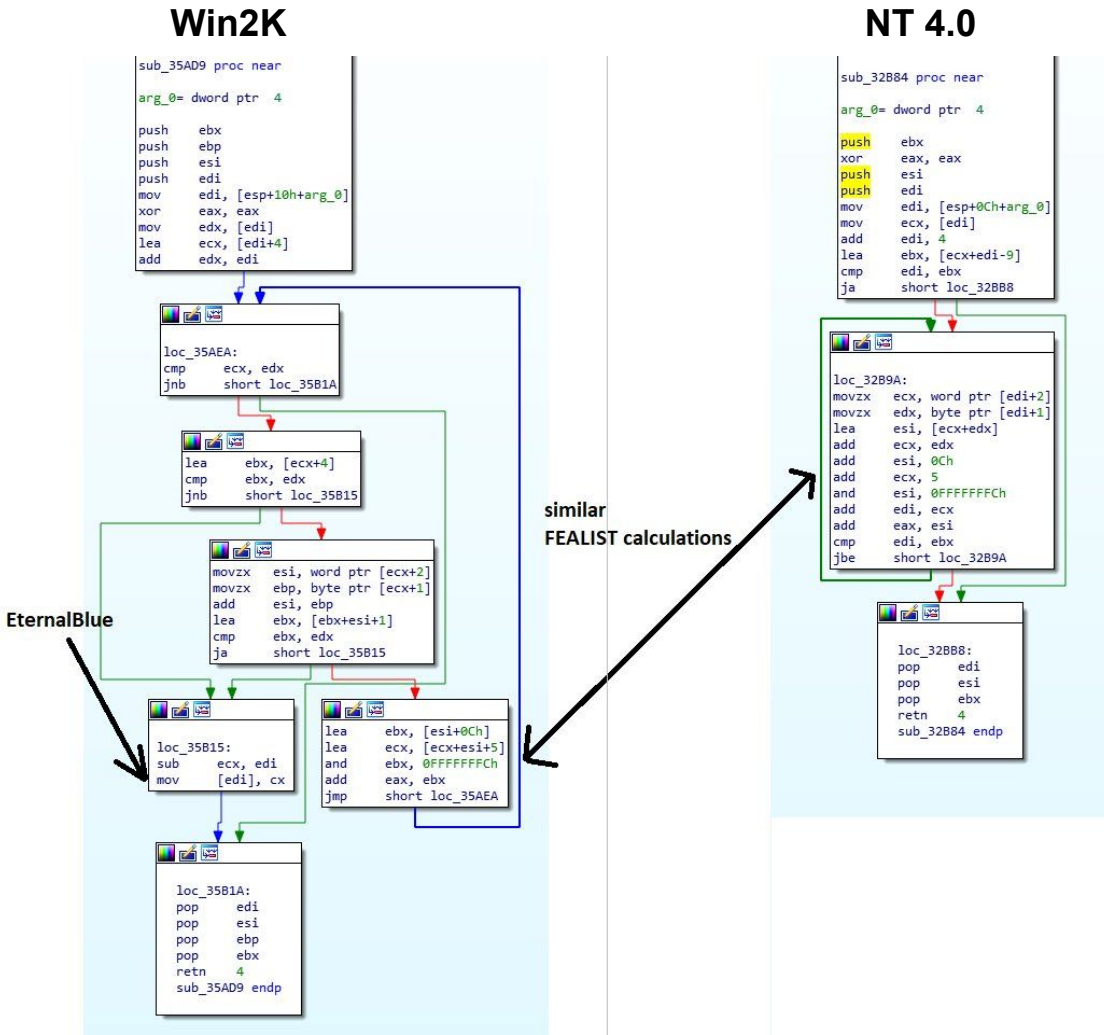
Itanium

```
loc_A4100:
sub r21 = r35, r32
adds r20 = 1, r32;;
shr.u r19 = r21, 8
st1 [r32] = r21;;
st1 [r20] = r19
nop.i 0;;
```

DEC Alpha

```
loc_5EBC0:
andnot  $11, 2, $17
subl    $13, $9, $9
ldl     $19, 0($17)
and     $11, 2, $18
inswl   $9, $18, $20
mskwl   $19, $18, $19
```

An Odd Observation...



An Odd Observation...

Win2K SP0

c:\users\nsa\desktop\samples\win2ksp0\srv.sys

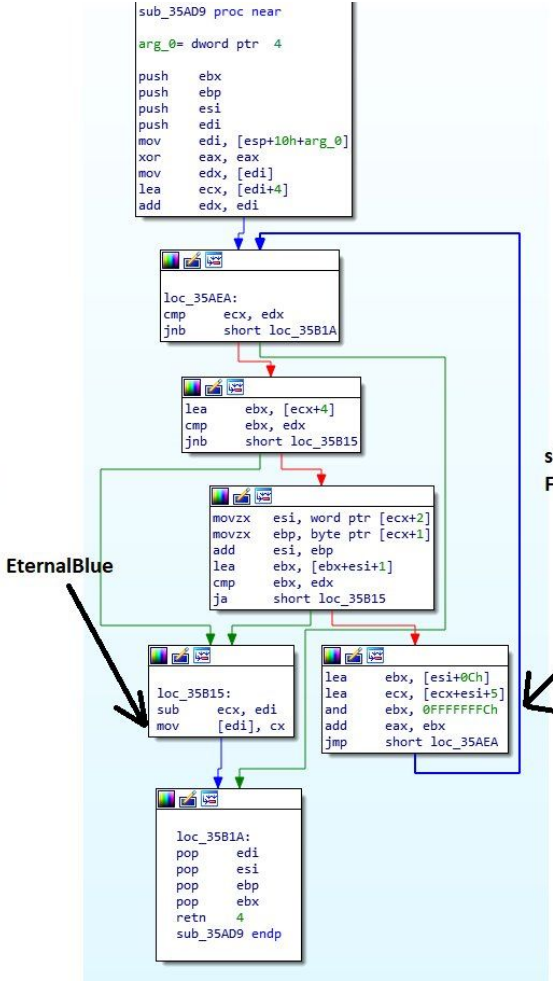
compiler-stamp	0x38437EED (Tue Nov 30 00:38:21 1999)

NT 4.0 SP6a SRP

c:\users\nsa\desktop\samples\nt4sp7\srv.sys

compiler-stamp	0x3A382E16 (Wed Dec 13 19:19:02 2000)

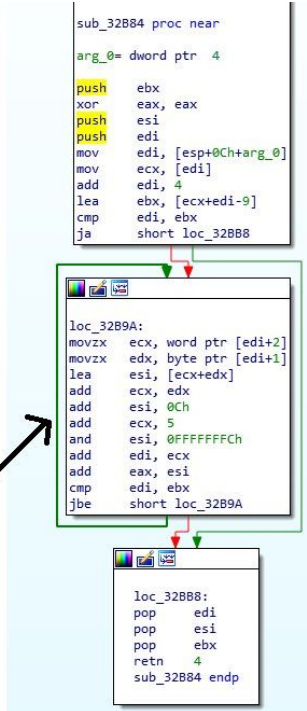
Win2K



EternalBlue

similar
FEALIST calculations

NT 4.0



(technically)
Newer, but no
EternalBlue

Least Resistance

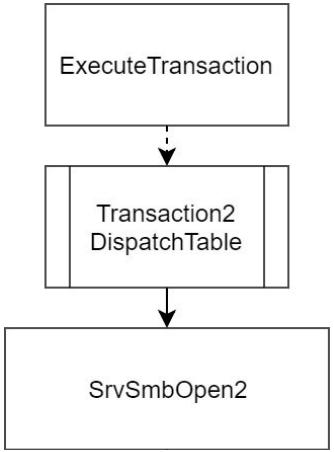
```
pkt Trans2_Open2_Parameters
{
     USHORT                Flags;
     USHORT                AccessMode;
     USHORT                Reserved1;
     SMB_FILE_ATTRIBUTES   FileAttributes;
     UTIME                 CreationTime;
     USHORT                OpenMode;
     ULONG                 AllocationSize;
     USHORT                Reserved[5];
     SMB_STRING            FileName;
};
pkt Trans2_Open2_Data
{
     SMB_FEA_LIST ExtendedAttributeList;
};
```

Requires a tree
e.g. IPC$

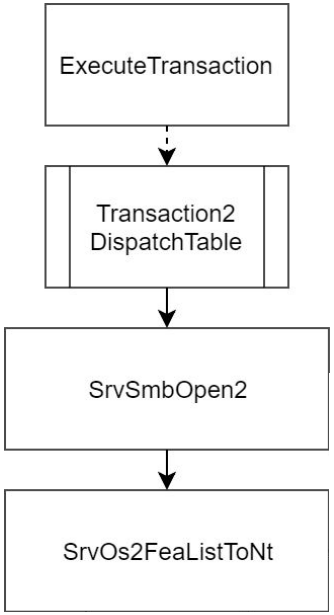
Side Note

- TRANS2_OPEN2?...
- CVE-2003-0201
  - Stack Buffer Overflow in Samba
- EchoWrecker
  - FuzzBunch, also dumped by Shadow Brokers

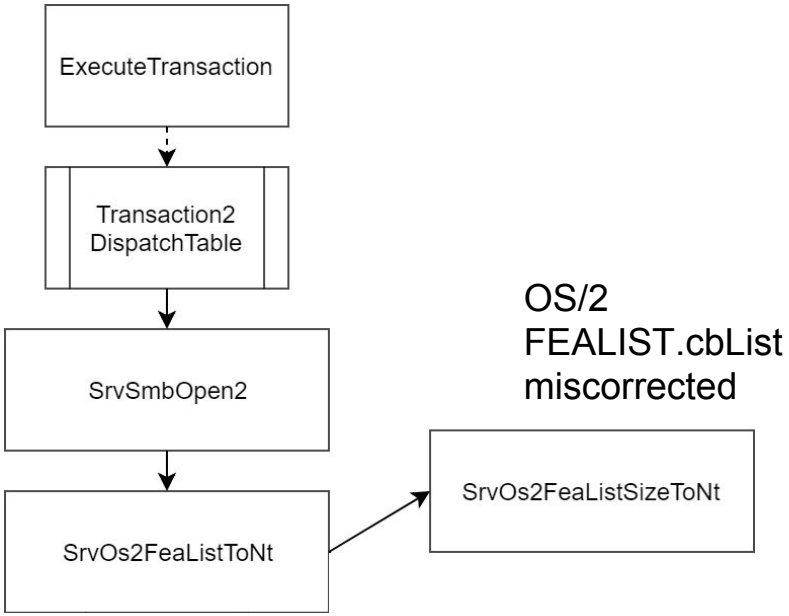
Basic Exploit Process



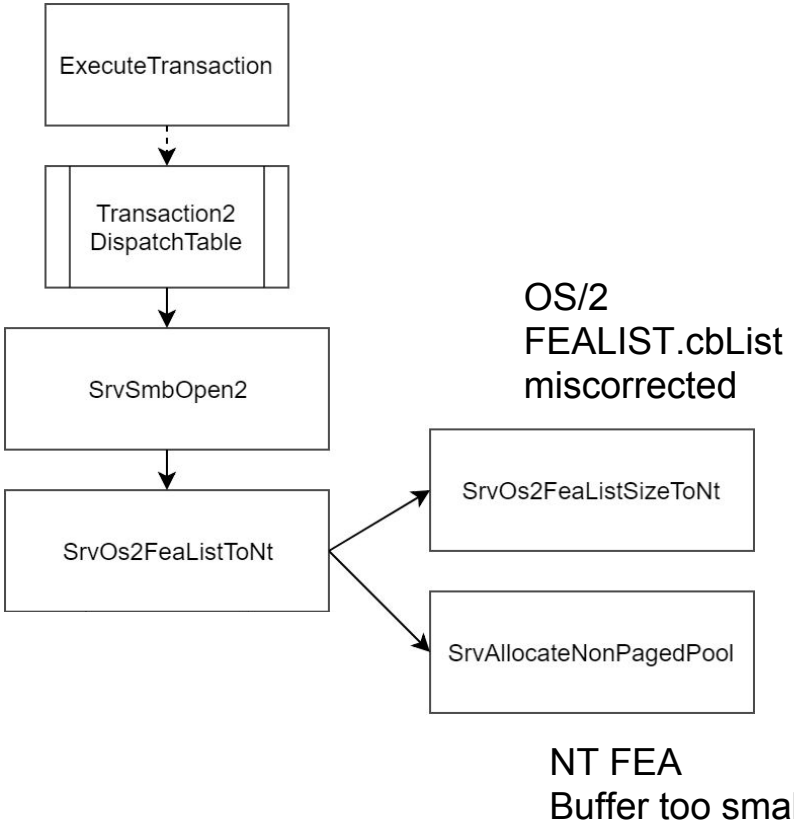
Basic Exploit Process



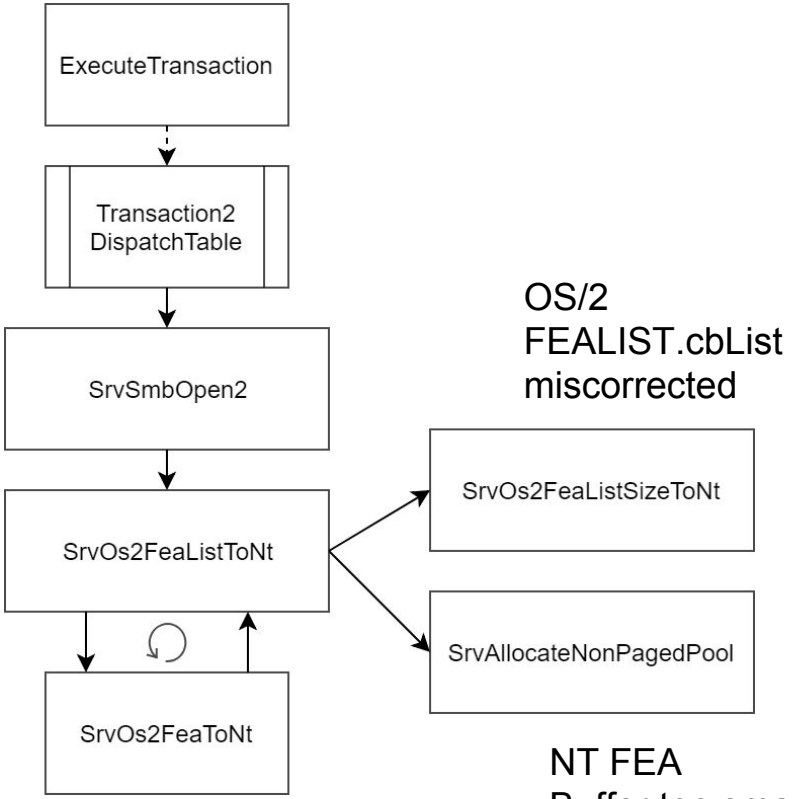
Basic Exploit Process

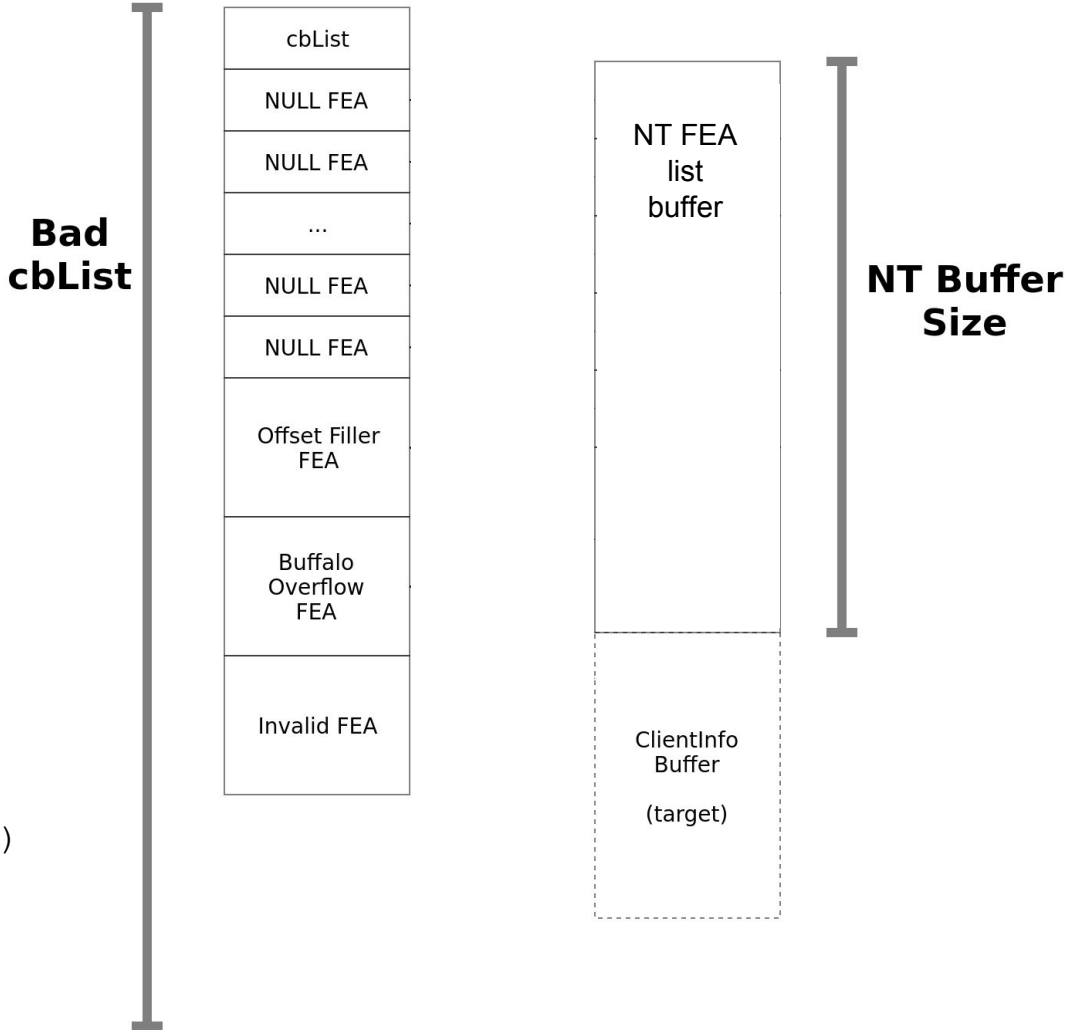


Basic Exploit Process



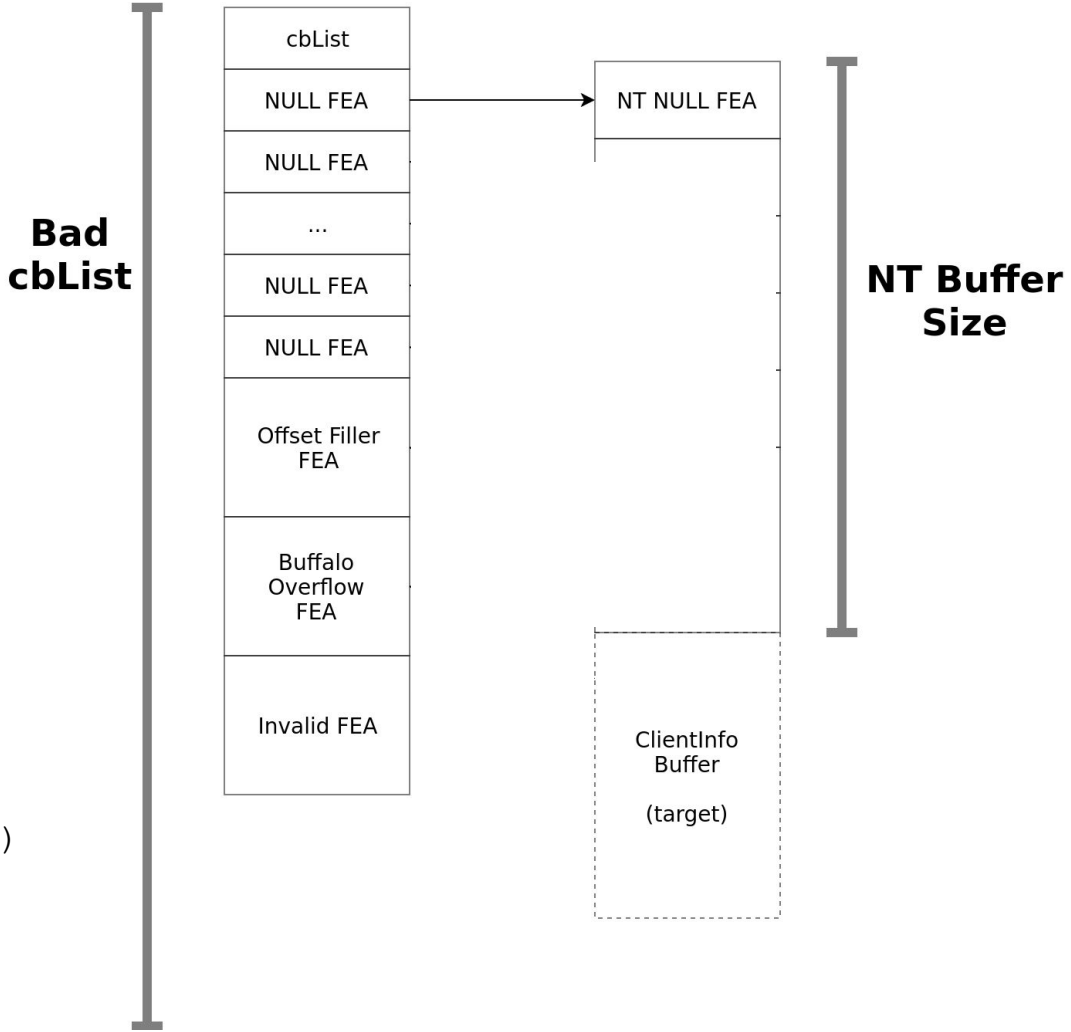
Basic Exploit Process





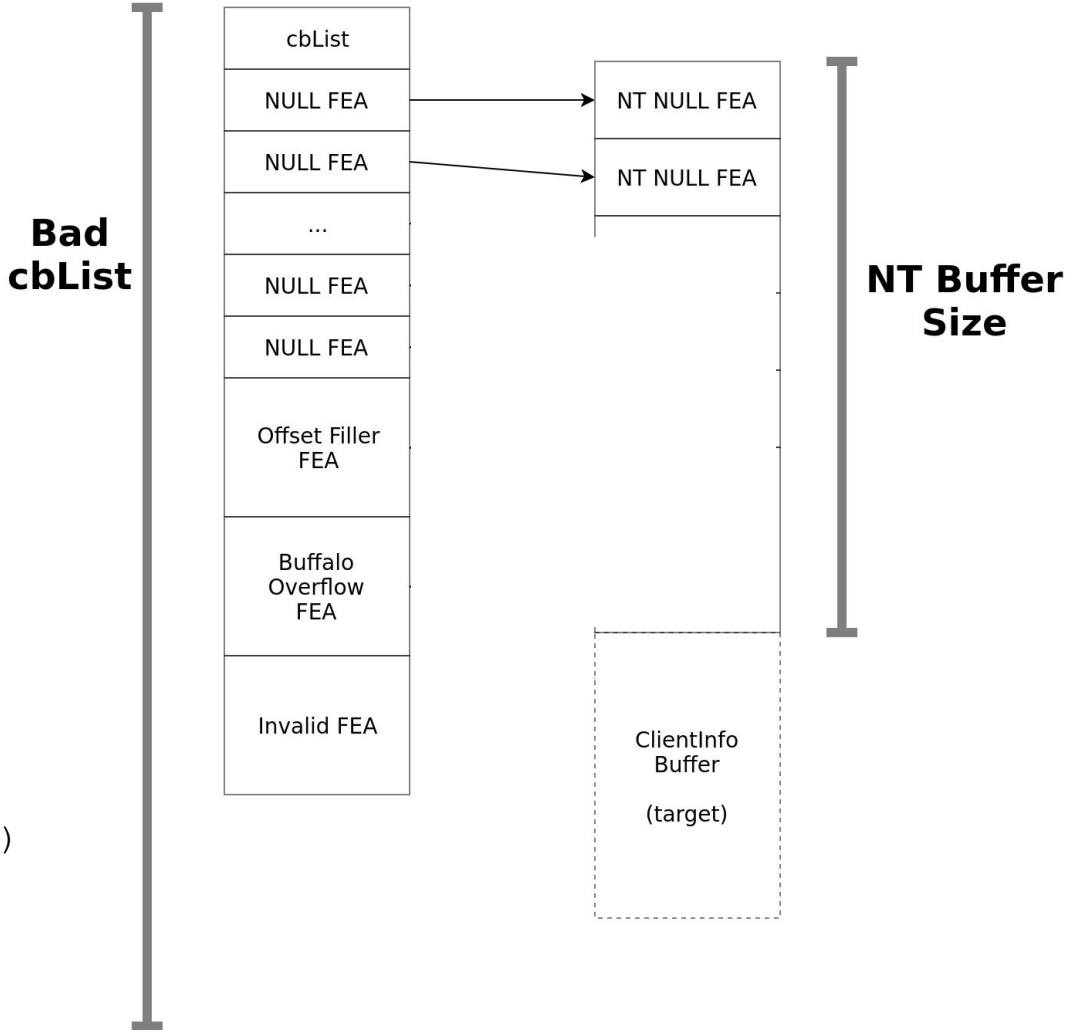
SrvOs2FeaListToNt()

```
    foreach:
        SrvOs2FeaToNt()
```



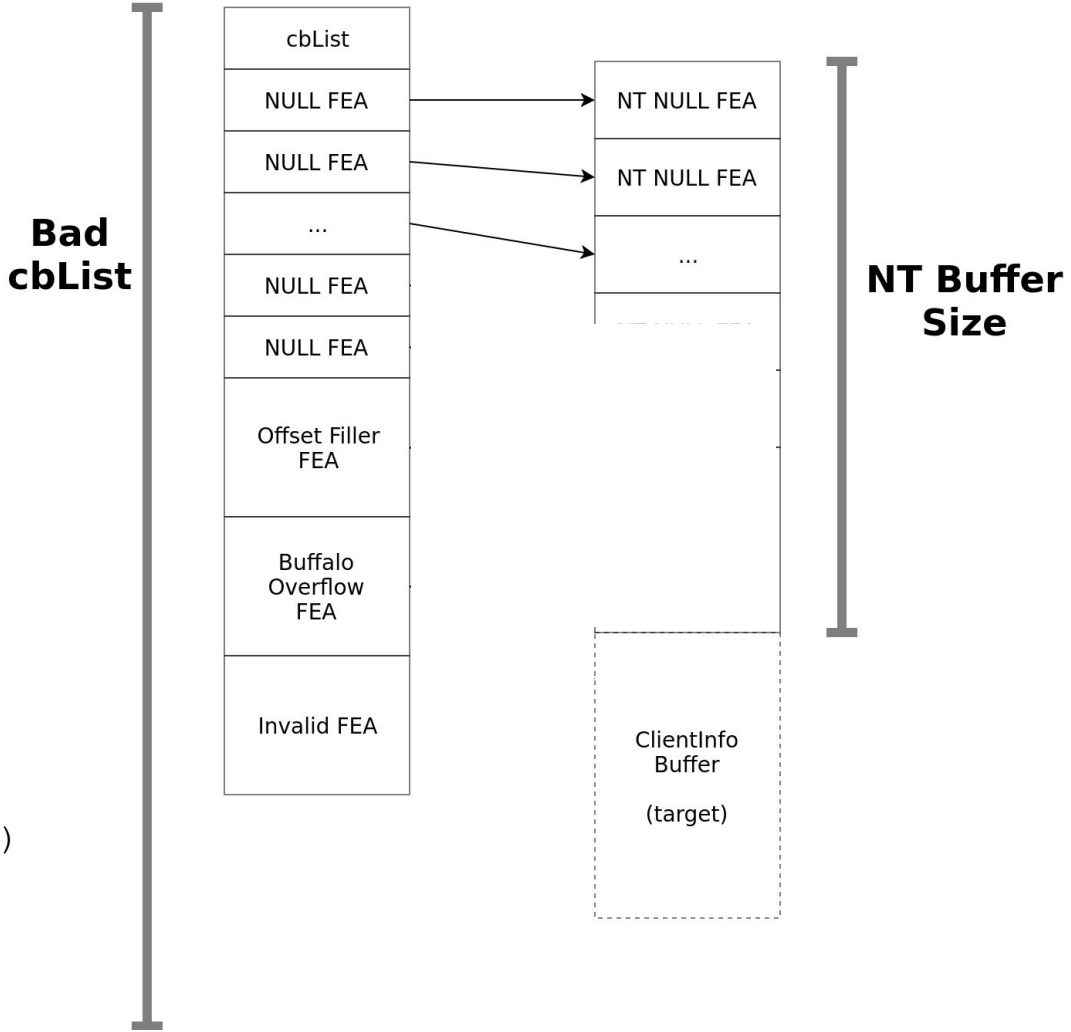
SrvOs2FeaListToNt()

```
    foreach:
        SrvOs2FeaToNt()
```



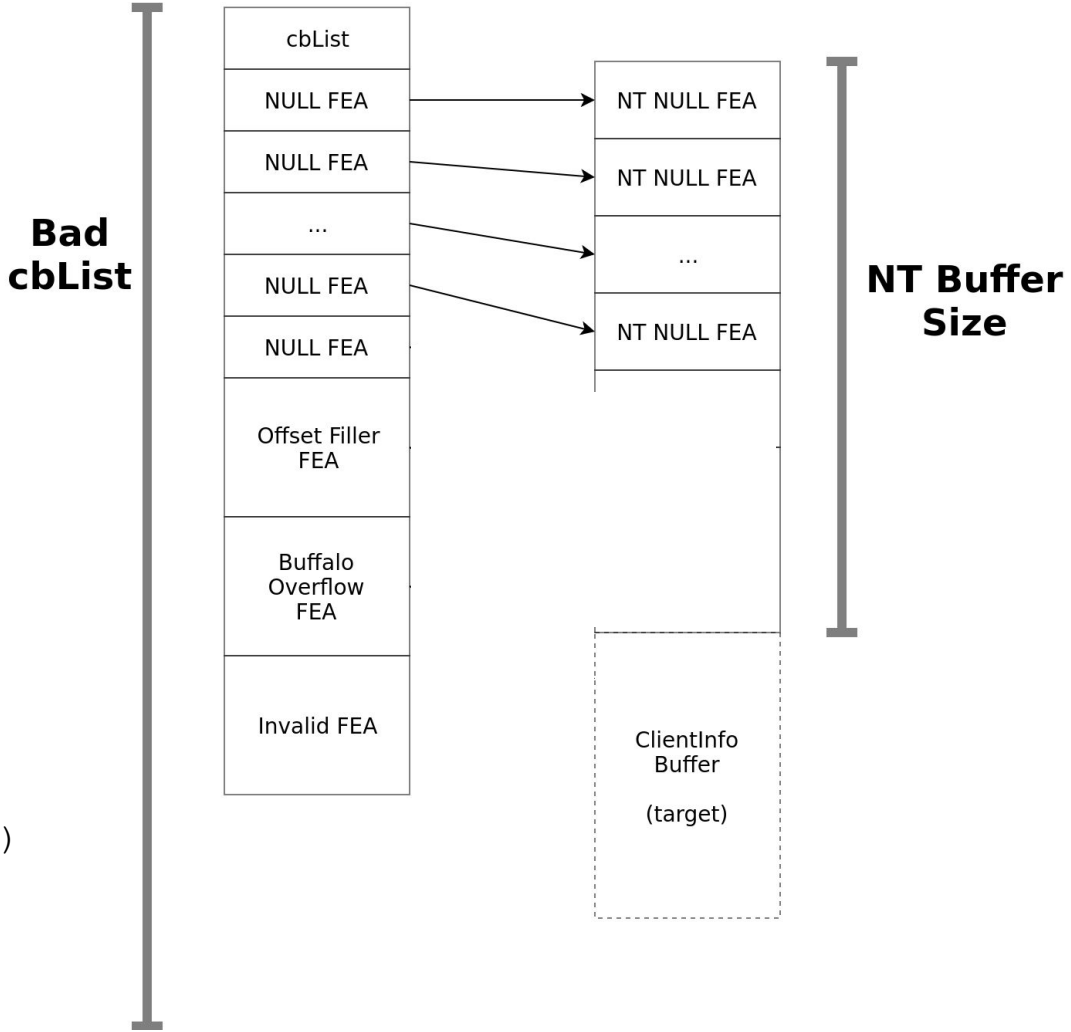
SrvOs2FeaListToNt()

```
    foreach:
        SrvOs2FeaToNt()
```



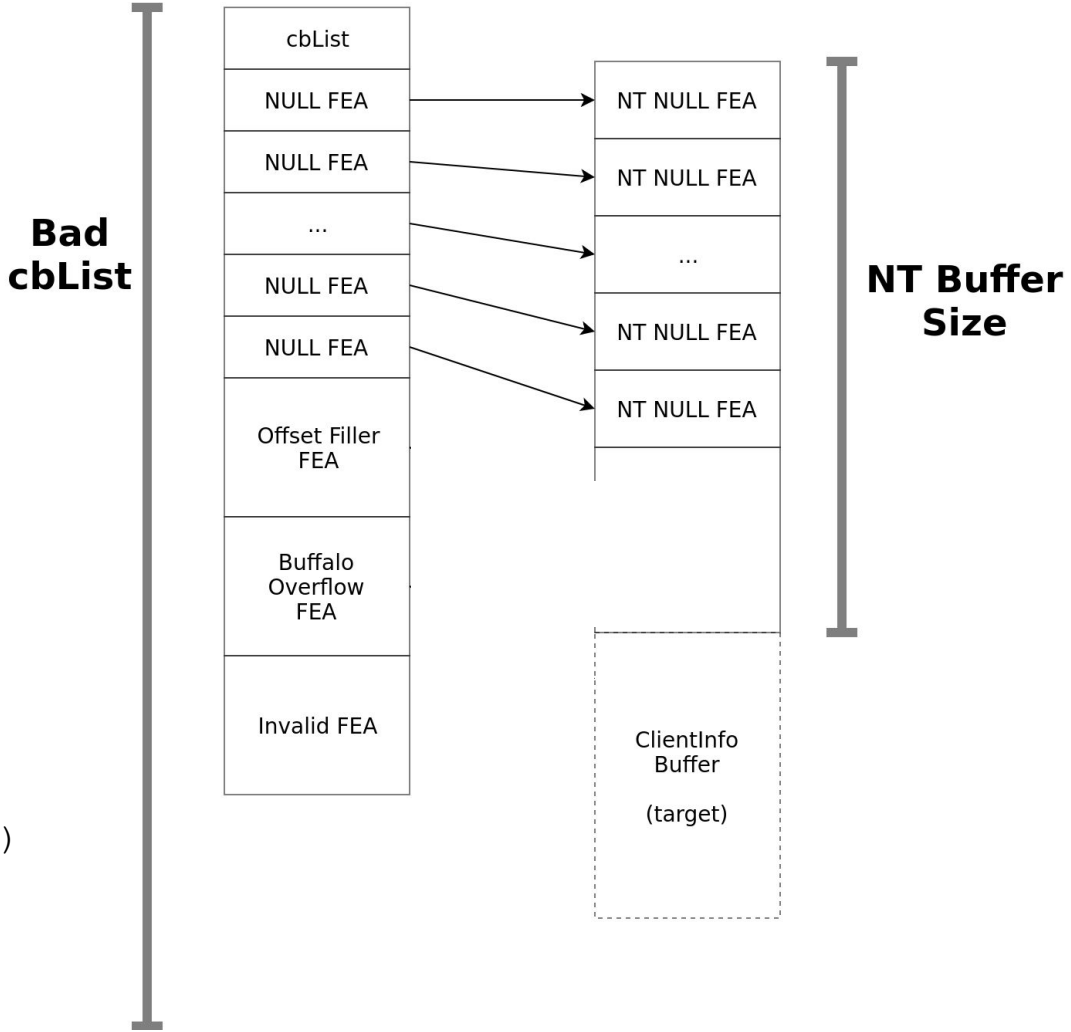
SrvOs2FeaListToNt()

```
    foreach:
        SrvOs2FeaToNt()
```



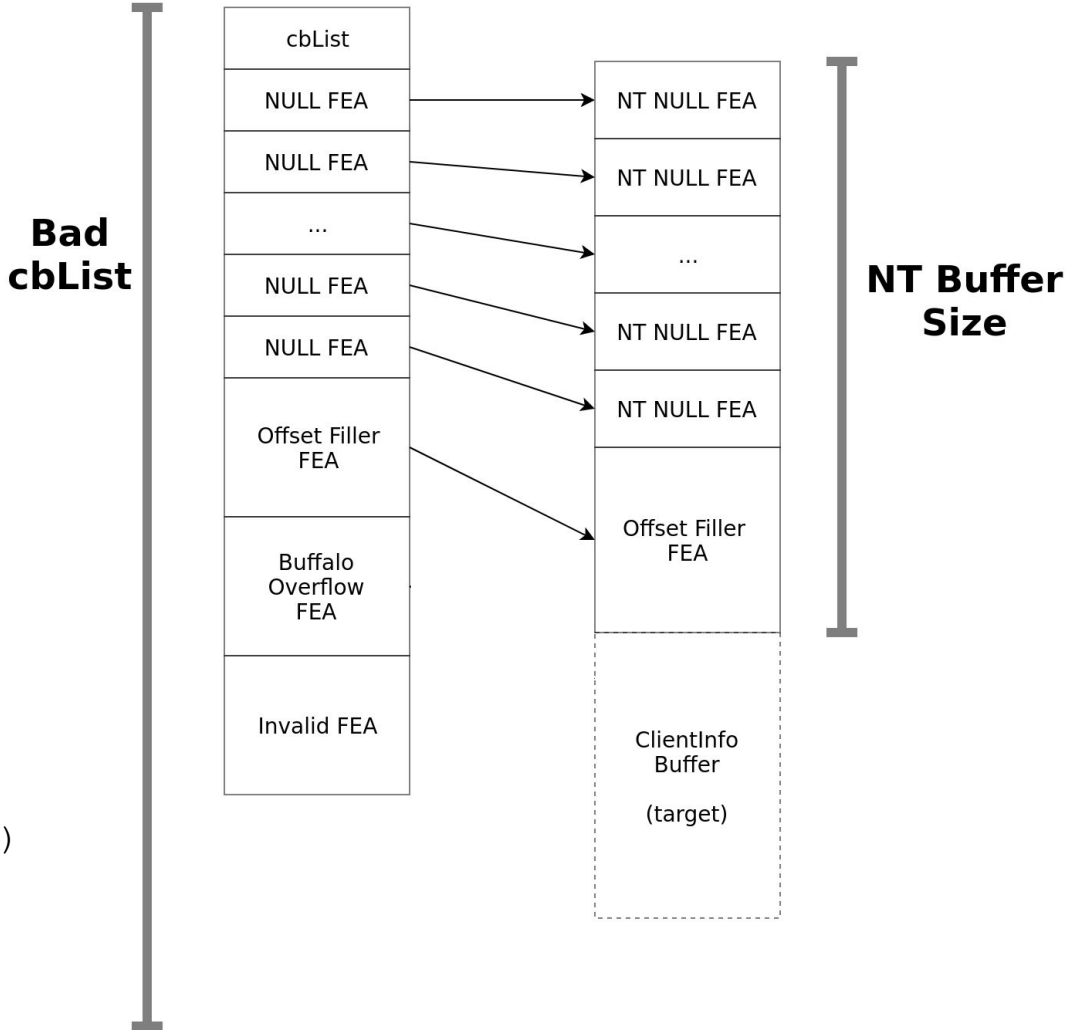
SrvOs2FeaListToNt()

```
    foreach:
        SrvOs2FeaToNt()
```



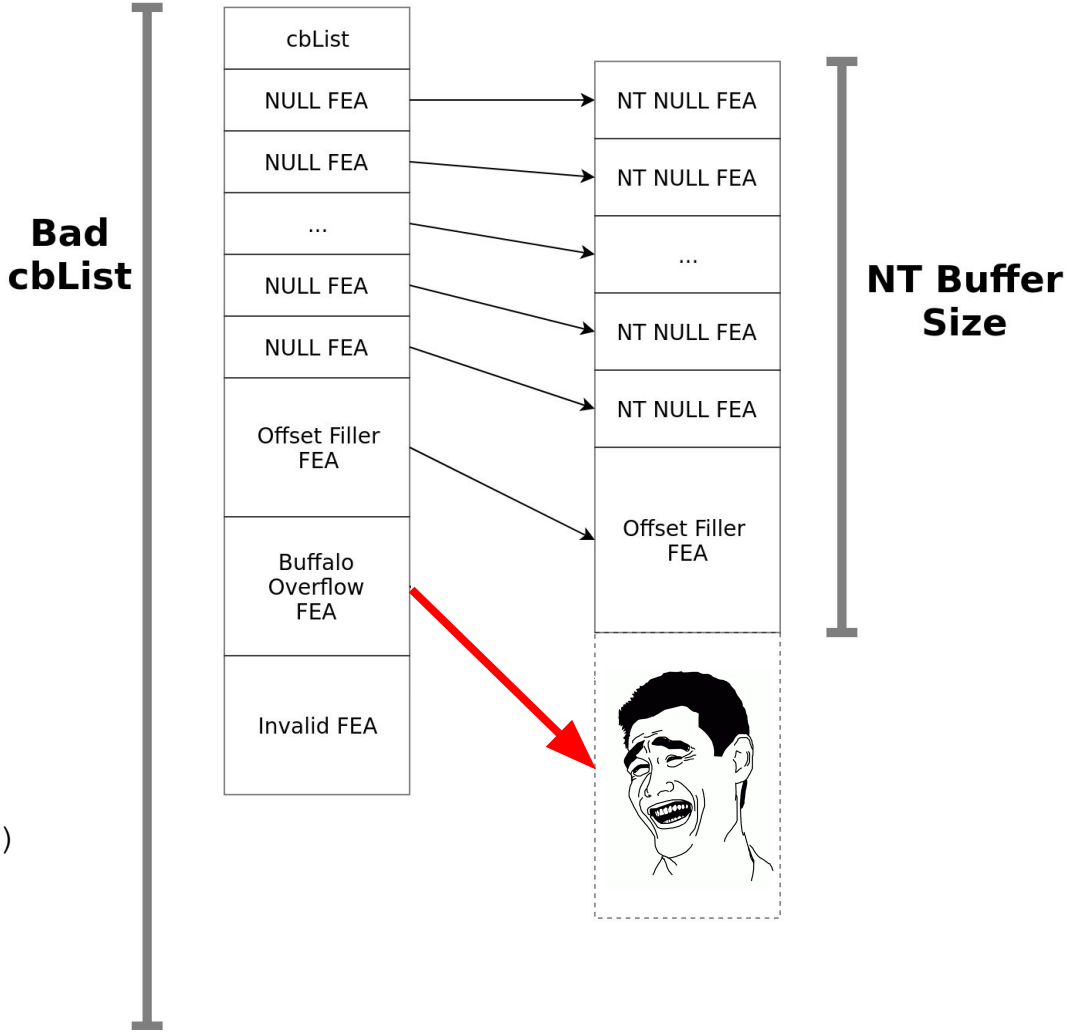
SrvOs2FeaListToNt()

```
    foreach:
        SrvOs2FeaToNt()
```



SrvOs2FeaListToNt()

```
    foreach:
        SrvOs2FeaToNt()
```

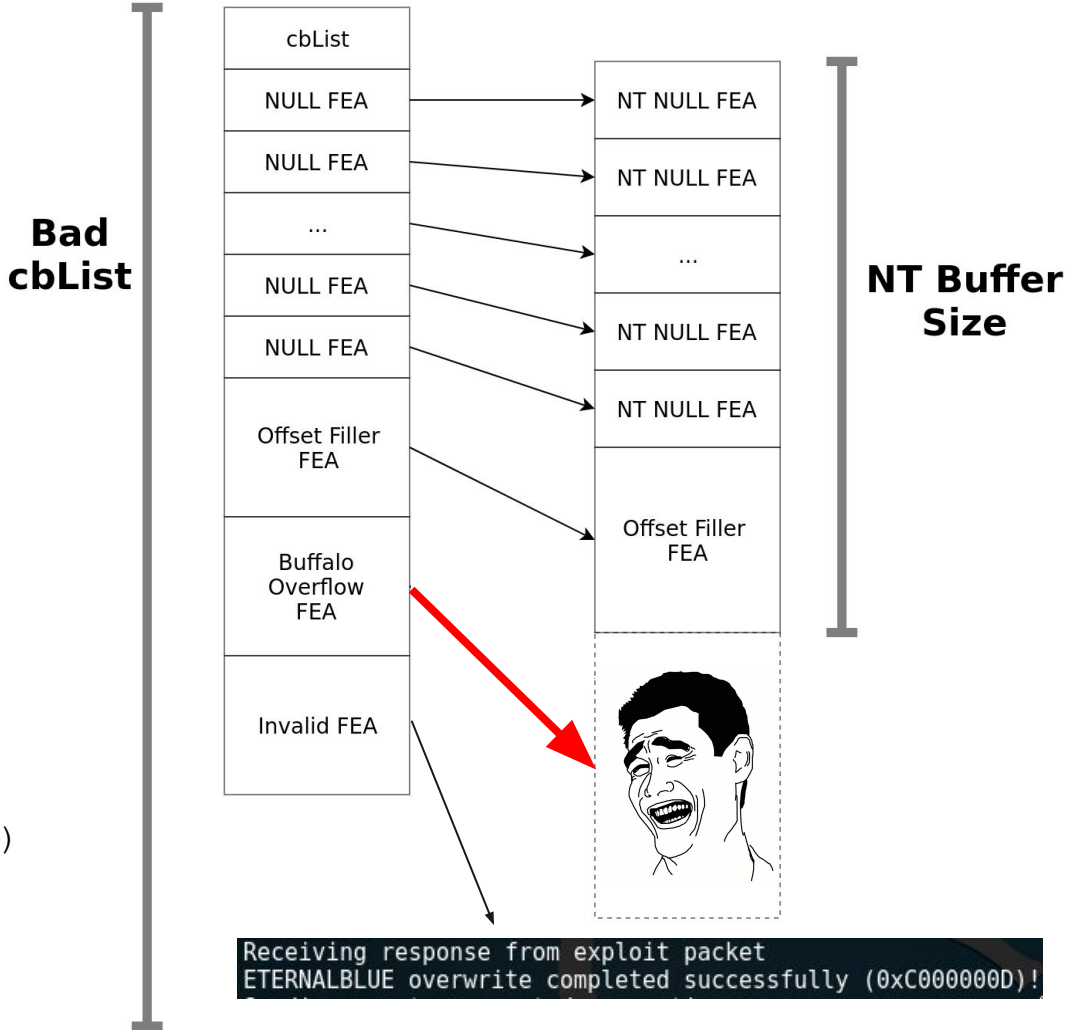


SrvOs2FeaListToNt()

```
    foreach:
        SrvOs2FeaToNt()
```

SrvOs2FeaListToNt()

```
    foreach:
        SrvOs2FeaToNt()
```



Bug #2

Session Setup Type Confusion

Session Setup Types

(SMB Param Block)

Offset	NT Security Parameter	Extended Security Parameter
0x0	WordCount	WordCount
0x1	AndXCommand	AndXCommand
0x2	AndXReserved	AndXReserved
0x3	AndXOffset	AndXOffset
0x4		
0x5	MaxBufferSize	MaxBufferSize
0x6		
0x7	MaxMpxCount	MaxMpxCount
0x8		
0x9	VcNumber	VcNumber
0xa		
0xb	SessionKey	SessionKey
0xc		
0xd		
0xe		
0xf	OEMPasswordLen	SecurityBlobLength
0x10		
0x11	UnicodePasswordLen	Reserved
0x12		
0x13	Reserved	
0x14		
0x15		Capabilities
0x16		
0x17	Capabilities	
0x18		
0x19		—
0x1a		—

13 words
(26 bytes)

12 Words to
SMB Data Block
(BCC, validated)

srv!BlockingSessionSetupAndX()

Scenario	WordCount	Capabilities*	Flags2†	Deduced Type
1	12	✓	✓	Extended Security
2	12	✓	x	NT Security
3	12	x	✓	Invalid
4	12	x	x	Invalid
5	13	✓	✓	Extended Security
6	13	✓	x	NT Security
7	13	x	✓	NT Security
8	13	x	x	NT Security

* CAP_EXTENDED_SECURITY

† FLAGS2_EXTENDED_SECURITY

Post BCC validation

Bug #2 - Session Setup Type Confusion

```
SMB (Server Message Block Protocol)
  SMB Header
      Server Component: SMB
      [Response in: 40]
      SMB Command: Session Setup AndX (0x73)
      NT Status: STATUS_SUCCESS (0x00000000)
    ▸ Flags: 0x18, Canonicalized Pathnames, Case Sensitivity
    ▾ Flags2: 0xc007, Unicode Strings, Error Code Type, Security Signatures, Extended Attributes, Long Names Allowed
        1... .... .... .... = Unicode Strings: Strings are Unicode
        .1.. .... .... .... = Error Code Type: Error codes are NT error codes
        ..0. .... .... .... = Execute-only Reads: Don't permit reads if execute-only
        ...0 .... .... .... = Dfs: Don't resolve pathnames with Dfs
        .... 0... .... .... = Extended Security Negotiation: Extended security negotiation is not supported
        .... .0.. .... .... = Reparse Path: The request does not use a @GMT reparse path
        .... .... .0.. .... = Long Names Used: Path names in request are not long file names
        .... .... ...0 .... = Security Signatures Required: Security signatures are not required
        .... .... .... 0... = Compressed: Compression is not requested
        .... .... .... .1.. = Security Signatures: Security signatures are supported
        .... .... .... ..1. = Extended Attributes: Extended attributes are supported
        .... .... .... ...1 = Long Names Allowed: Long file names are allowed in the response
      Process ID High: 0
      Signature: 0000000000000000
      Reserved: 0000
      Tree ID: 0
      Process ID: 65279
      User ID: 0
      Multiplex ID: 64
  Session Setup AndX Request (0x73)
      Word Count (WCT): 12
      AndXCommand: No further commands (0xff)
      Reserved: 00
      AndXOffset: 0
      Max Buffer: 4356
      Max Mpx Count: 10
      VC Number: 2
      Session Key: 0x00000000
      Security Blob Length: 0
      Reserved: 00000000
    ▸ Capabilities: 0x80000000, Extended Security
      Byte Count (BCC): 22
      Security Blob: <MISSING>
      Native OS: \377
      Native LAN Manager:
      Primary Domain:
      Extra byte parameters: 00000000000000000000000000
```

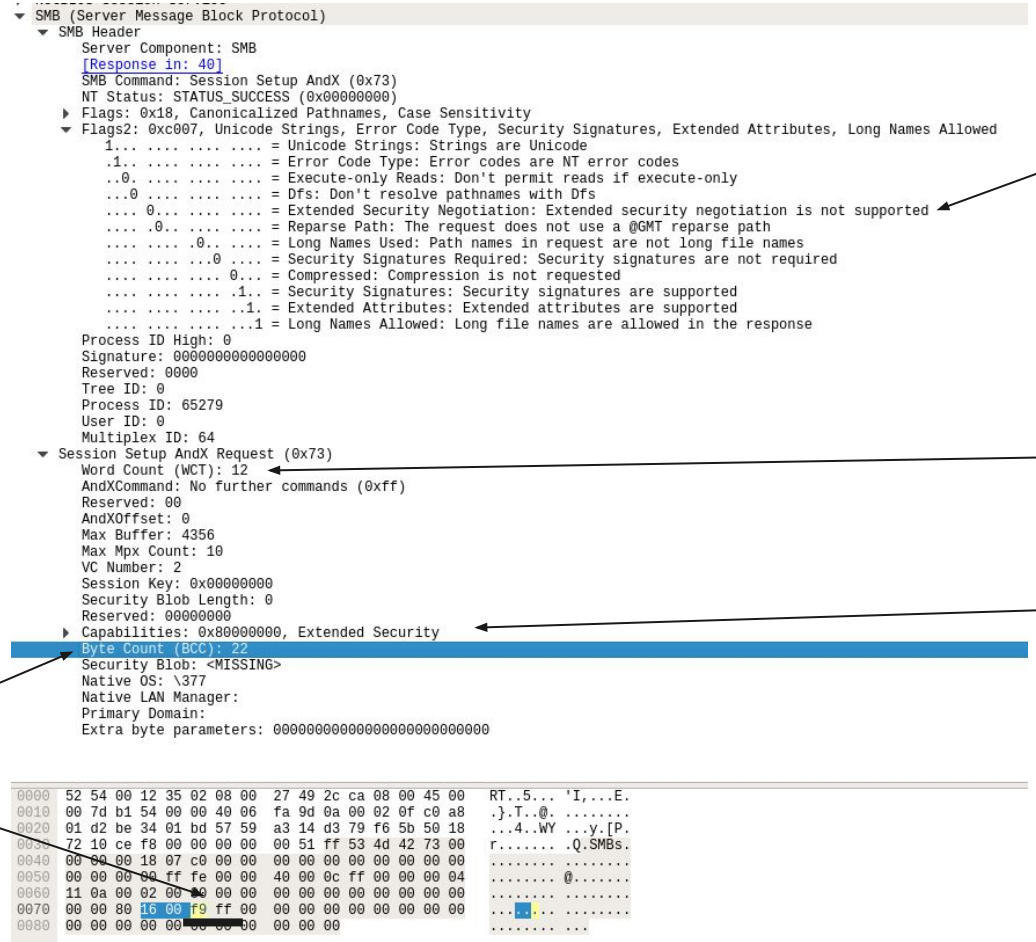
```
0000  52 54 00 12 35 02 08 00  27 49 2c ca 08 00 45 00   RT..5... 'I,...E.
0010  00 7d b1 54 00 00 40 06  fa 9d 0a 00 02 0f c0 a8   .}.T..@. ........
0020  01 d2 be 34 01 bd 57 59  a3 14 d3 79 f6 5b 50 18   ...4..WY ...y.[P.
0030  72 10 ce f8 00 00 00 00  00 51 ff 53 4d 42 73 00   r....... .Q.SMBs.
0040  00 00 00 18 07 c0 00 00  00 00 00 00 00 00 00 00   ........ ........
0050  00 00 00 00 ff fe 00 00  40 00 0c ff 00 00 00 04   ........ @.......
0060  11 0a 00 02 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0070  00 00 80 16 00 f9 ff 00  00 00 00 00 00 00 00 00   ........ ........
0080  00 00 00 00 00 00 00 00  00 00 00                 ........ ...
```

~FLAGS2_EXT_SEC (0)

WCT = 12

CAP_EXT_SEC (1)

Bug #2 - Session Setup Type Confusion



~FLAGS2_EXT_SEC (0)

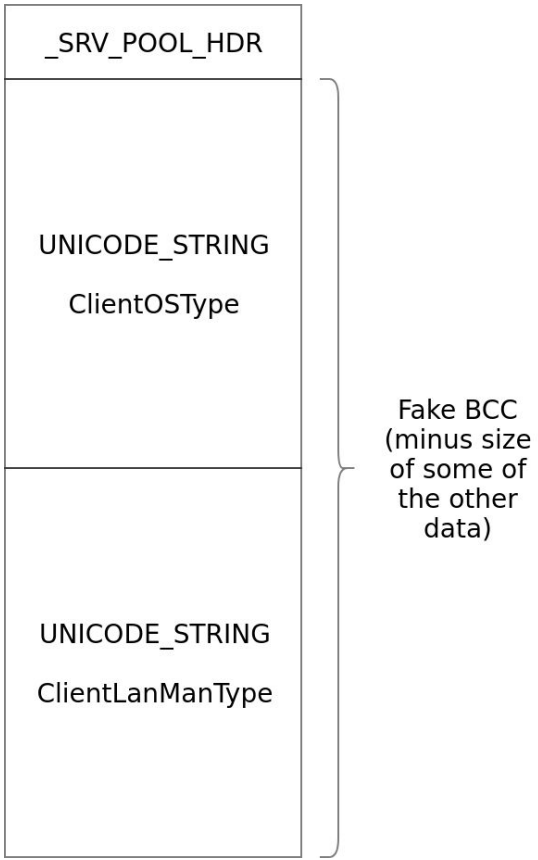
WCT = 12

CAP_EXT_SEC (1)

BCC wrong!
(WCT = 13)
BCC = 0xfff9

“ClientInfo” Buffer

```
struct _CONNECTION
{
    // ...

    UNICODE_STRING     ClientOSType;
    UNICODE_STRING     ClientLanManType;

    // ...
};
```



BUG #2 In a Nutshell

- Allows:
    1. Allocate large NPP memory blocks (pool feng shui)
    2. Free on demand (close socket)

Bug #3

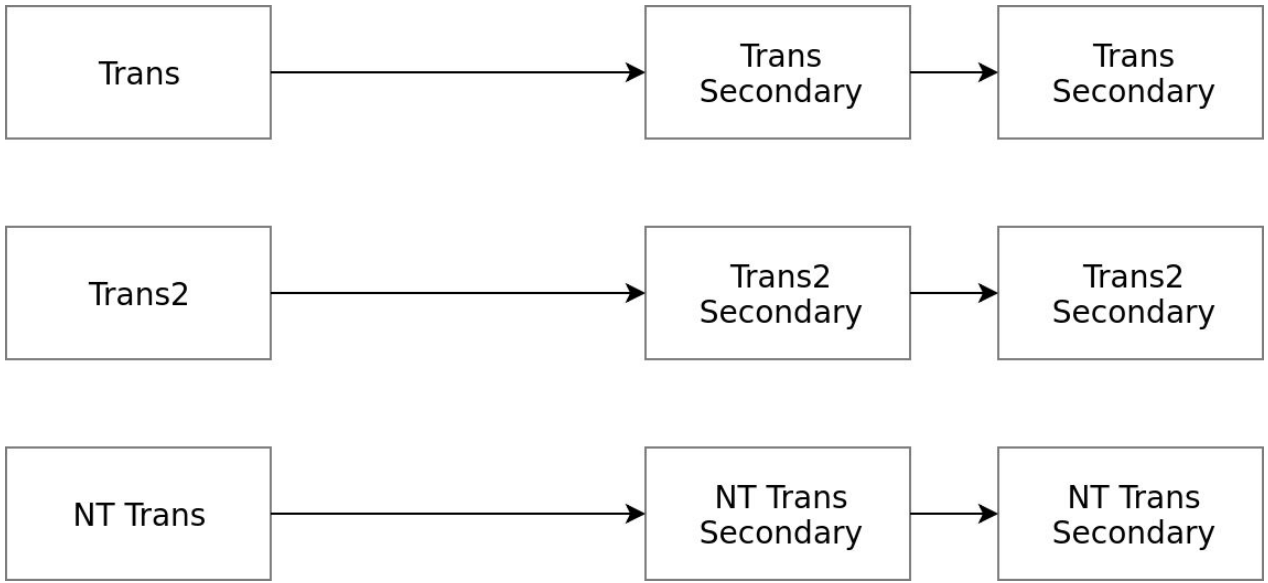
Oversized Trans2 Dispatch

Size Problem

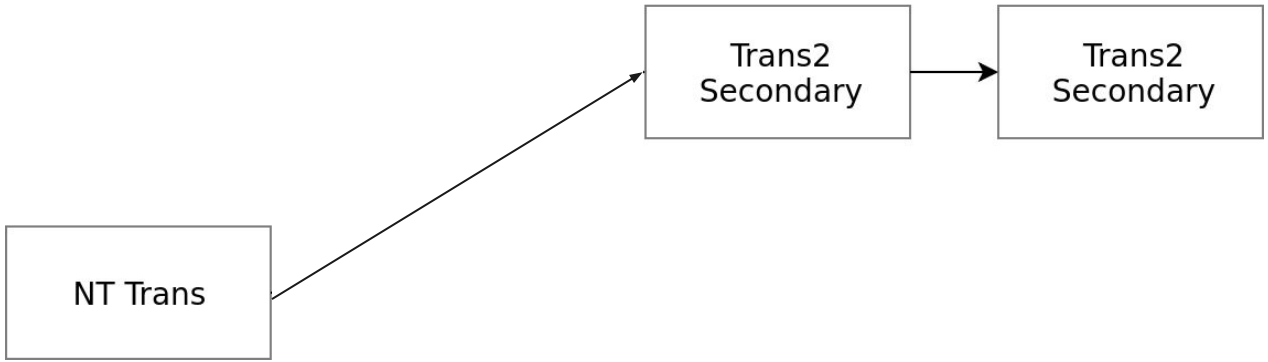
- Bug #1 = attacker FEALIST >= 0x10000
- Trans2 only allows WORD trans param/data

0x10000 > 0xFFFF

Expected SMB Client Behavior



But What If?



NT Trans

SrvSmbNtTransaction

SrvAllocateTransaction

_TRANSACTION

NT Trans

Trans2 Secondary

SrvSmbNtTransaction

SrvSmbTransaction2Secondary

SrvAllocateTransaction

SrvFindTransaction

_TRANSACTION

NT Trans

Trans2 Secondary

Trans2 Secondary

SrvSmbNtTransaction

SrvSmbTransaction2Secondary

WORK_CONTEXT
NextCommand =
Trans2Secondary

SrvAllocateTransaction

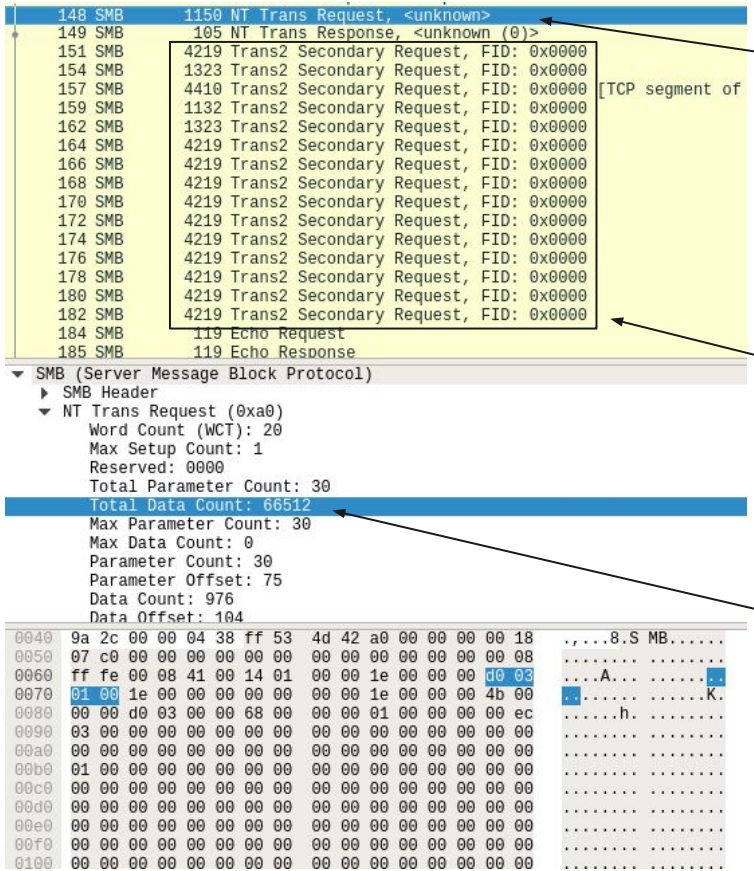
SrvFindTransaction

ExecuteTransaction

_TRANSACTION

SrvTransaction2
DispatchTable

Bug #3 - Oversized Trans2 Dispatch

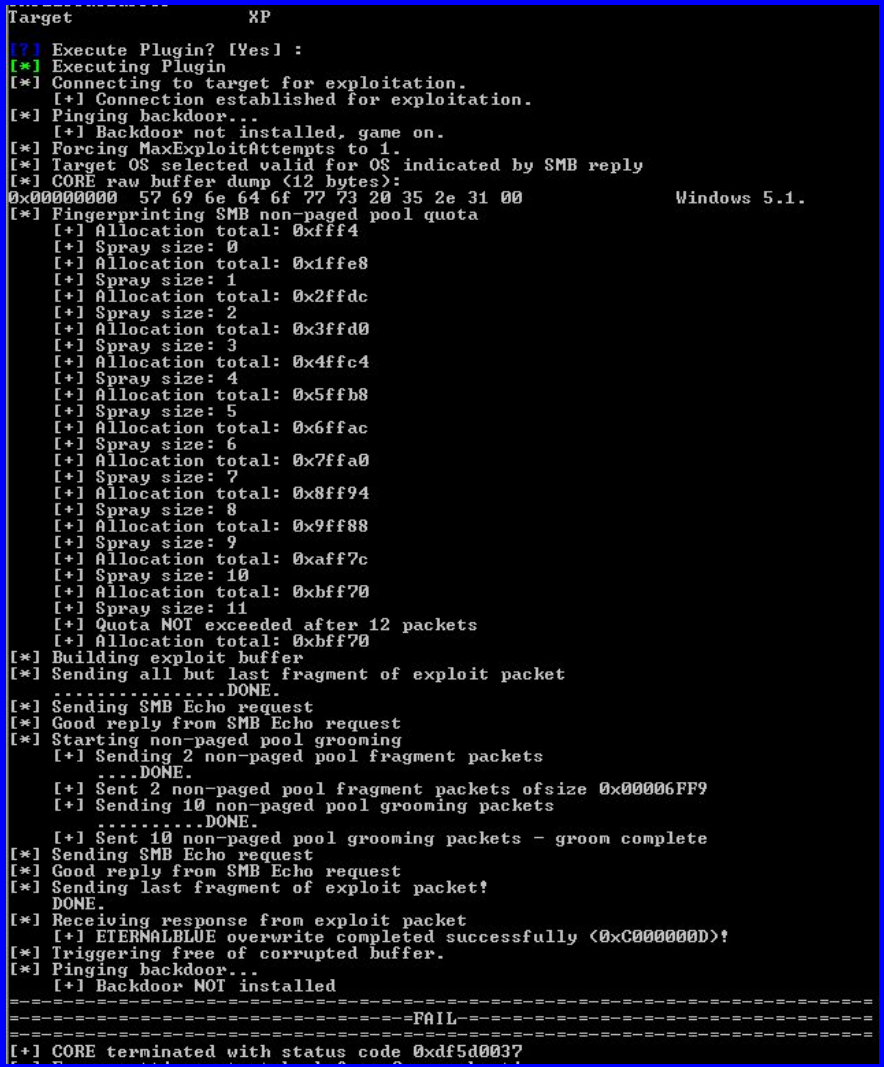


NT TRANS
primary
(allows DWORD)

TRANS2
secondaries

(data > 0xffff)

ETERNALBLUE
WinXP



Memory Quota Check

- WinXP Minimum requirements
  - 64 MB of RAM
- Pre-exploit check
  - Enough memory to even do a pool feng shui??
  - Failed allocs leave event logs

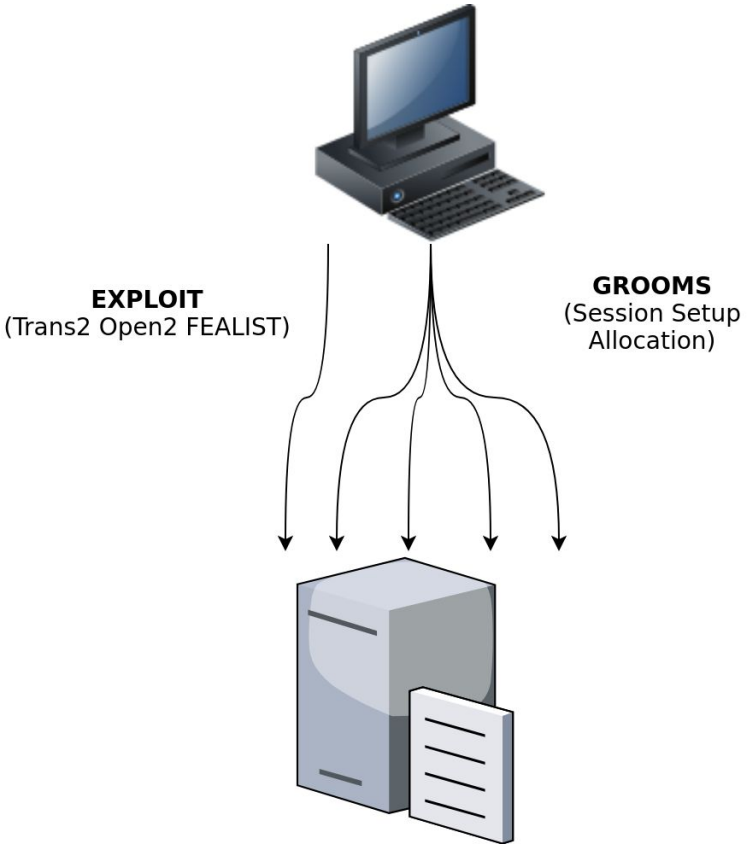
See Windows Internals Book 2

```
Target                XP

[?] Execute Plugin? [Yes] :
[*] Executing Plugin
[*] Connecting to target for exploitation.
    [+] Connection established for exploitation
[*] Pinging backdoor...
    [+] Backdoor not installed, game on.
[*] Forcing MaxExploitAttempts to 1.
[*] Target OS selected valid for OS indicated b
[*] CORE raw buffer dump (12 bytes):
0x00000000  57 69 6e 64 6f 77 73 20 35 2e 31 00
[*] Fingerprinting SMB non-paged pool quota
    [+] Allocation total: 0xfff4
    [+] Spray size: 0
    [+] Allocation total: 0x1ffe8
    [+] Spray size: 1
    [+] Allocation total: 0x2ffdc
    [+] Spray size: 2
    [+] Allocation total: 0x3ffd0
    [+] Spray size: 3
    [+] Allocation total: 0x4ffc4
    [+] Spray size: 4
    [+] Allocation total: 0x5ffb8
    [+] Spray size: 5
    [+] Allocation total: 0x6ffac
    [+] Spray size: 6
    [+] Allocation total: 0x7ffa0
    [+] Spray size: 7
    [+] Allocation total: 0x8ff94
    [+] Spray size: 8
    [+] Allocation total: 0x9ff88
    [+] Spray size: 9
    [+] Allocation total: 0xaff7c
    [+] Spray size: 10
    [+] Allocation total: 0xbff70
    [+] Spray size: 11
    [+] Quota NOT exceeded after 12 packets
    [+] Allocation total: 0xbff70
[*] Building exploit buffer
```

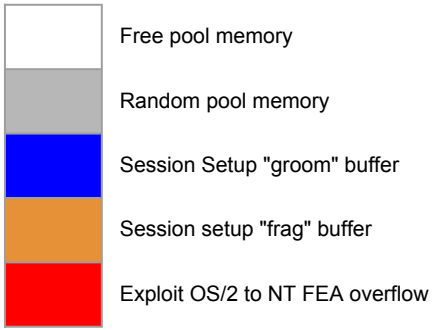
Information	4/8/2018	3:41:39 AM	Servic
Information	4/8/2018	3:40:38 AM	Servic
Information	4/8/2018	3:40:37 AM	Servic
Information	4/8/2018	3:40:37 AM	Servic
Information	4/8/2018	3:40:37 AM	Servic
Information	4/8/2018	3:40:37 AM	Servic
Information	4/8/2018	3:40:37 AM	Servic
Information	4/8/2018	3:40:37 AM	Servic
Information	4/8/2018	3:39:59 AM	Setup
Information	4/8/2018	3:40:34 AM	srserv
Error	6/15/2018	7:48:02 PM	Srv
Error	6/6/2018	5:38:33 PM	Srv
Error	4/10/2018	7:59:49 PM	Srv
Error	4/11/2018	5:14:40 PM	Syster
Error	4/11/2018	12:43:02 ...	Syster
Error	4/10/2018	9:31:58 PM	Syster

XP NPP Ingredients

- Exploit
  - Primary - NT transaction
  - Secondary - TRANS2_OPEN2
- Session Setup bug
  - Grooms
    - Overflow target
  - Frags
    - Smaller allocations using same method



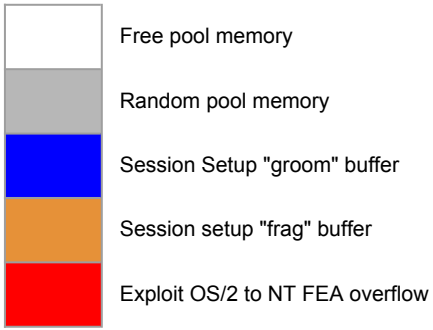
XP EternalBlue Grooming



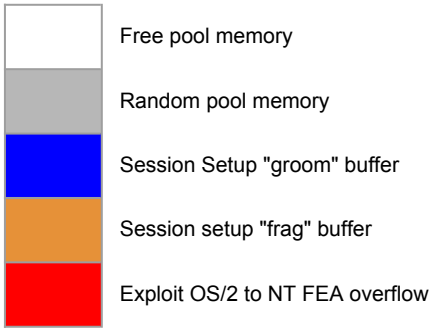
- **Step 0.** Pre-Exploitation Memory Layout
  - Random stuff in a fragmented pool

XP EternalBlue Grooming

- **Step 1.** Send all of FEALIST except last Trans2 Open2 secondary
  - NT Primary
  - All Trans2 (- 1)
  - The NT FEA Buffer will not be reserved yet

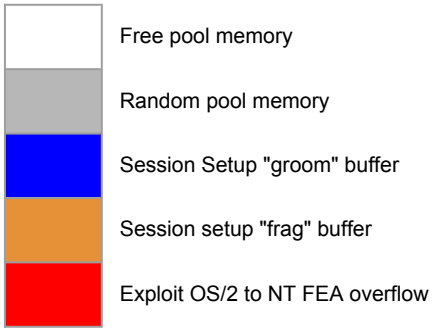


XP EternalBlue Grooming



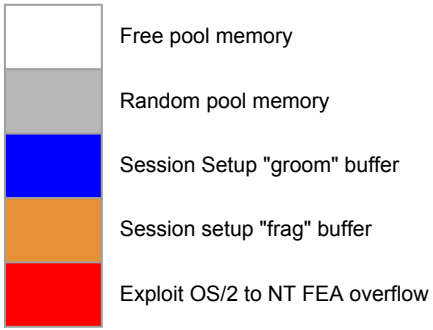
- **Step 2.** Send frag buffers
  - Fill up random problematic holes?
  - BugSize: 0x6ff9

XP EternalBlue Grooming



- **Step 2.** Send frag buffers
  - Fill up random problematic holes?
  - BugSize: 0x6ff9

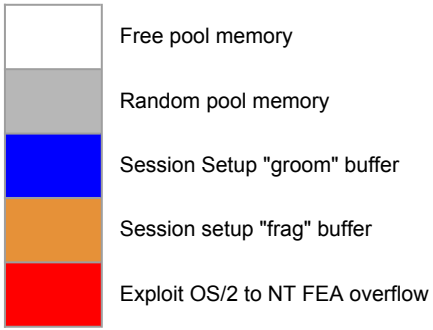
XP EternalBlue Grooming



- **Step 2.** Send frag buffers
  - Fill up random problematic holes?
  - BugSize: 0x6ff9

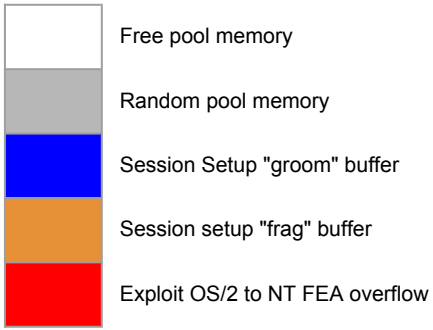
XP EternalBlue Grooming

- **Step 2.** Send frag buffers
  - Fill up random problematic holes?
  - BugSize: 0x6ff9



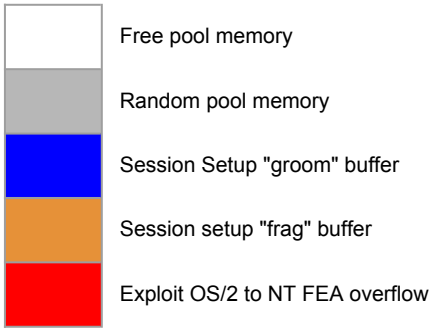
XP EternalBlue Grooming

- **Step 3.** Send groom buffers
  - Causes new allocations
  - BugSize: 0xfff9



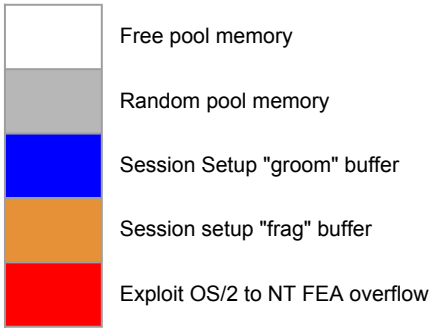
XP EternalBlue Grooming

- **Step 3.** Send groom buffers
  - Causes new allocations
  - BugSize: 0xfff9



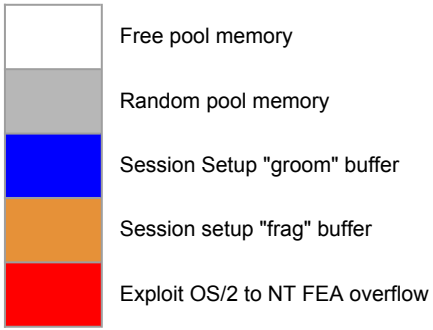
XP EternalBlue Grooming

- **Step 3.** Send groom buffers
  - Causes new allocations
  - BugSize: 0xfff9



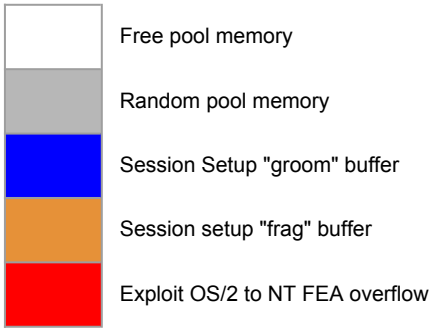
XP EternalBlue Grooming

- **Step 3.** Send groom buffers
  - Causes new allocations
  - BugSize: 0xfff9



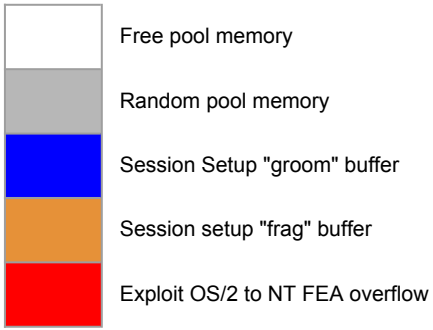
XP EternalBlue Grooming

- **Step 3.** Send groom buffers
  - Causes new allocations
  - BugSize: 0xfff9

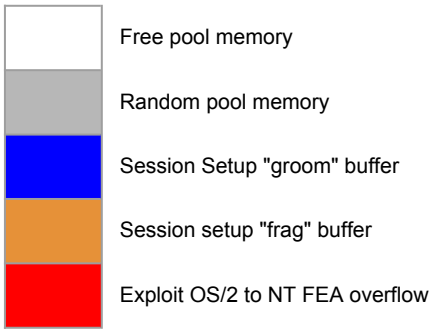


XP EternalBlue Grooming

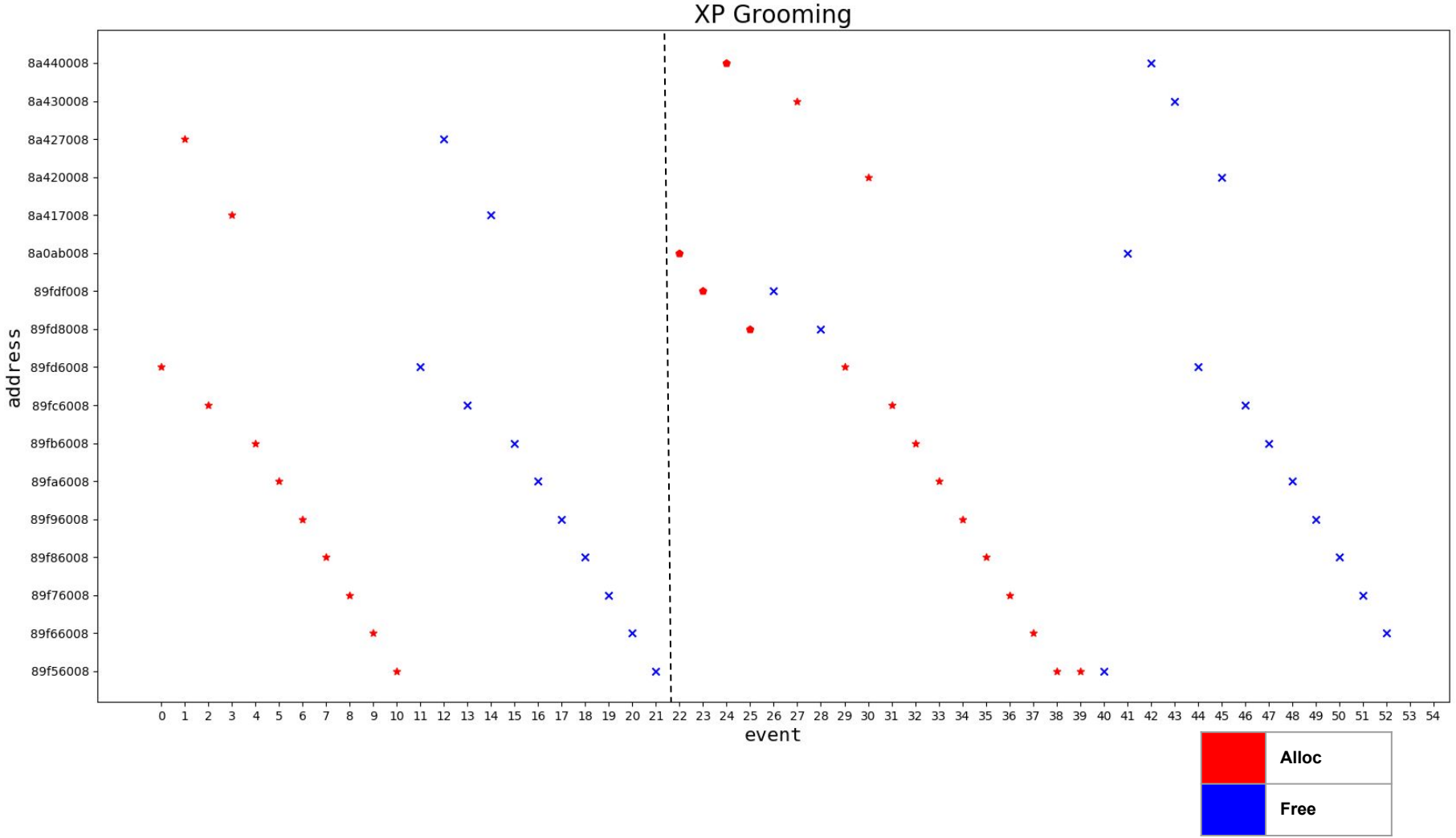
- **Step 3.** Send groom buffers
  - Causes new allocations
  - BugSize: 0xfff9

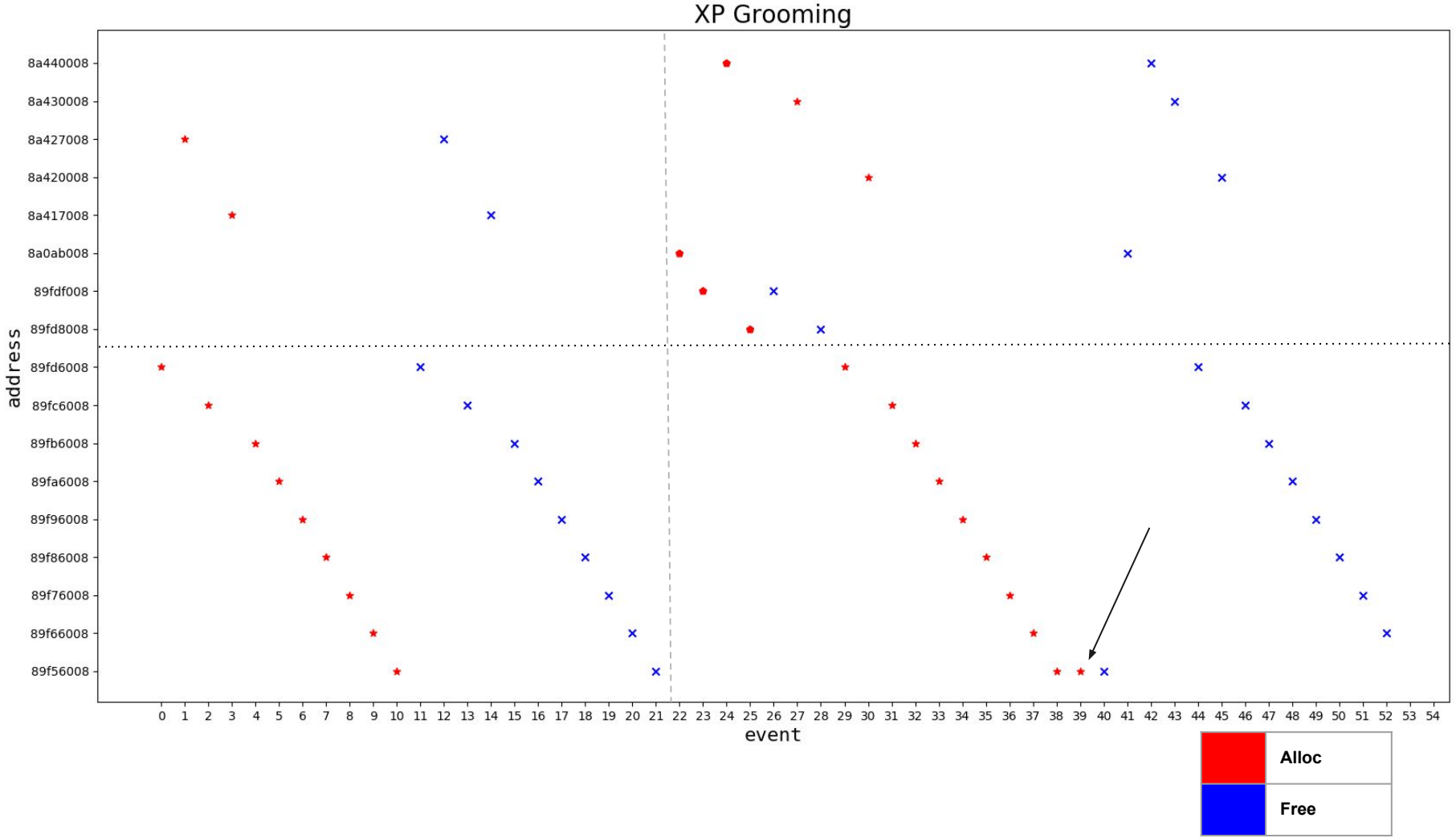


XP EternalBlue Grooming



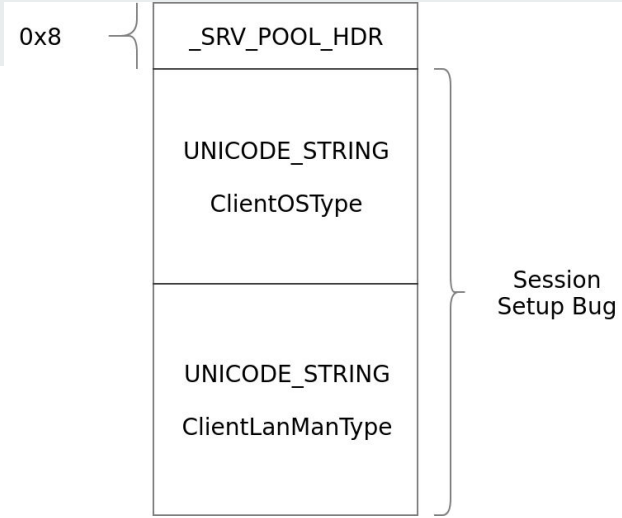
- **Step 4.** Send final TRANS2 OPEN2 packet
  - The OS/2 to NT FEA conversion will happen
  - AllocSize: 0xffec


XP Grooming
address
8a440008
8a430008
8a427008
8a420008
8a417008
8a0ab008
89fdf008
89fd8008
89fd6008
89fc6008
89fb6008
89fa6008
89f96008
89f86008
89f76008
89f66008
89f56008
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54
event
Alloc
Free


XP Grooming
address
8a440008
8a430008
8a427008
8a420008
8a417008
8a0ab008
89fdf008
89fd8008
89fd6008
89fc6008
89fb6008
89fa6008
89f96008
89f86008
89f76008
89f66008
89f56008
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54
event
Alloc
Free

Target of Buffalo Overflow

```
struct _SRV_POOL_HEADER
{
     ULONG     AllocationSize;              /* 0x0 */

     _SRV_POOL_HEADER   **LookAsideList;    /* 0x4 */      // lock xchg heap pattern

     /* 0x8 */
     /* Actual ClientInfo Buffer goes here */
};
```



http://illmatics.com/XP2003_Exploitation.pdf

Windows XP Power State

Type	Address/+Offset	Notes
`KPCR`	**`0xFFDFF000`**	Boot Processor Control Region (fixed addr)
`KPCR.KPRCB`	**`+0x120`**	All 32-bit NT
`KPRCB.PROCESSOR_POWER_STATE`	**`+0xB30`**	NT 5.1 (XP)

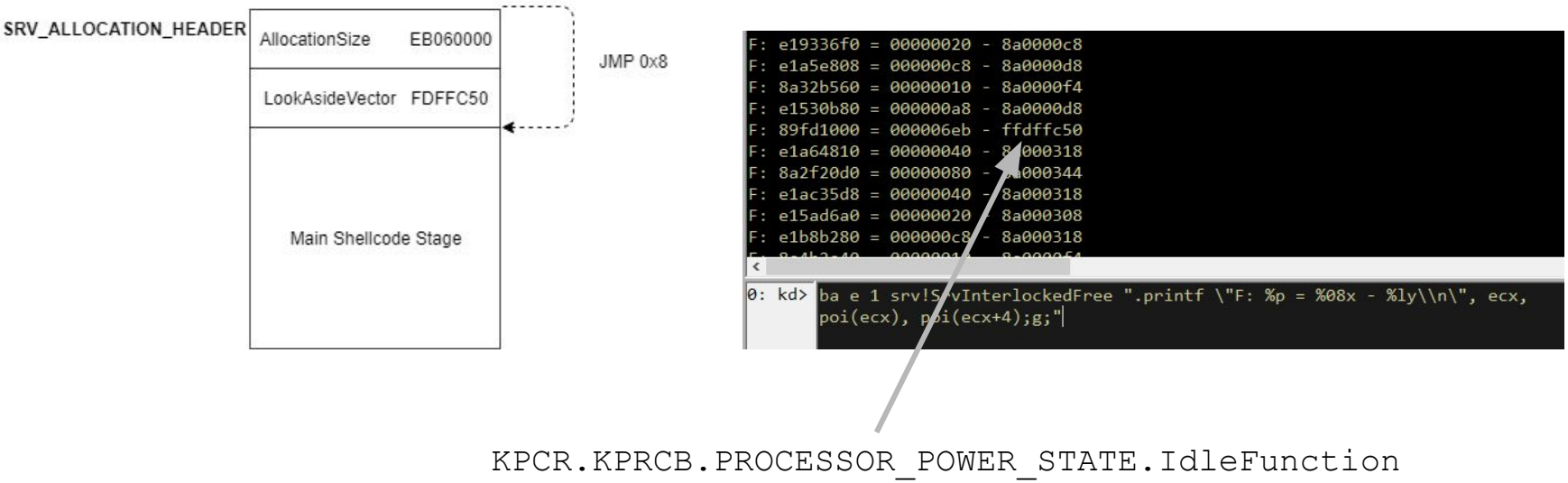
Thus:

`0xFFDFF000 + 0x120 + 0xB30 =` **`0xFFDFFC50`**

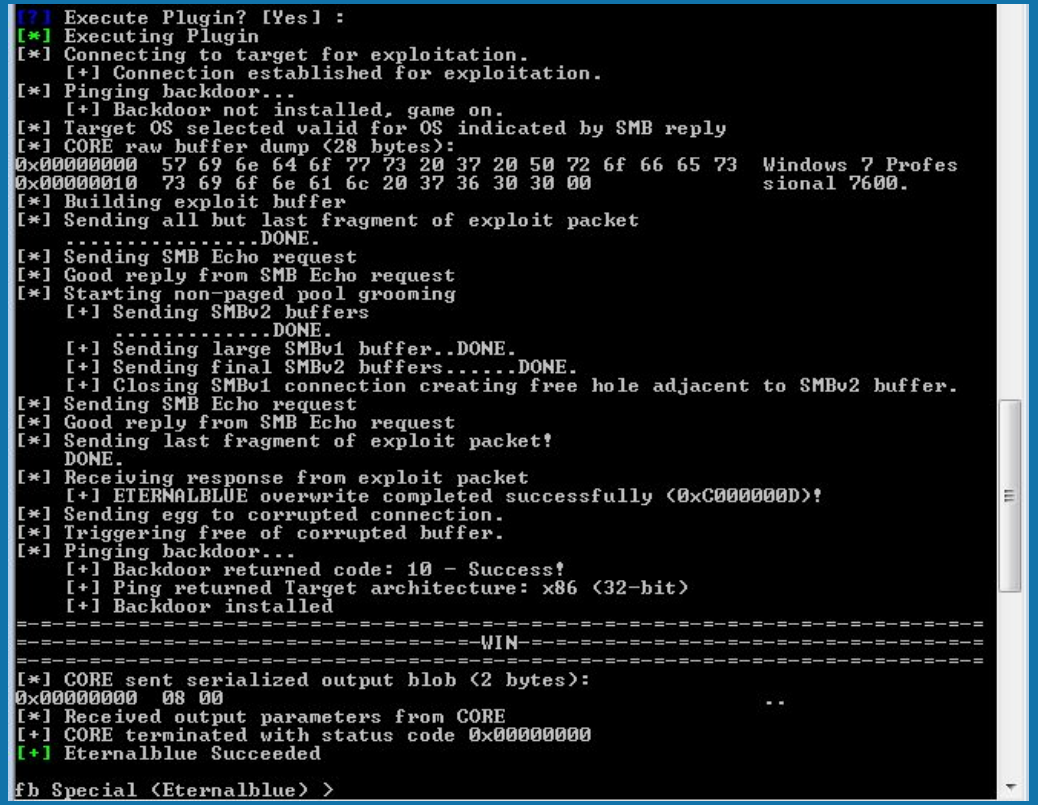
[1] https://www.geoffchappell.com/studies/windows/km/ntoskrnl/structs/kpcr.htm
[2] https://www.geoffchappell.com/studies/windows/km/ntoskrnl/structs/kprcb/i386.htm

```
struct PROCESSOR_POWER_STATE
{
     PVOID     IdleFunction;           /* 0x0 - 1 bajillion times/sec */

     /* ... */
};
```

Hijack Execution - InterlockedExchange()

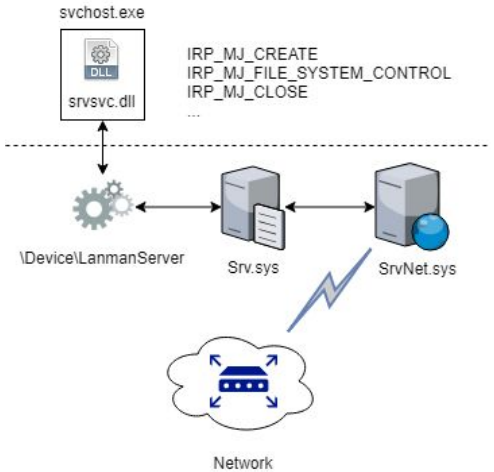


ETERNALBLUE Win7



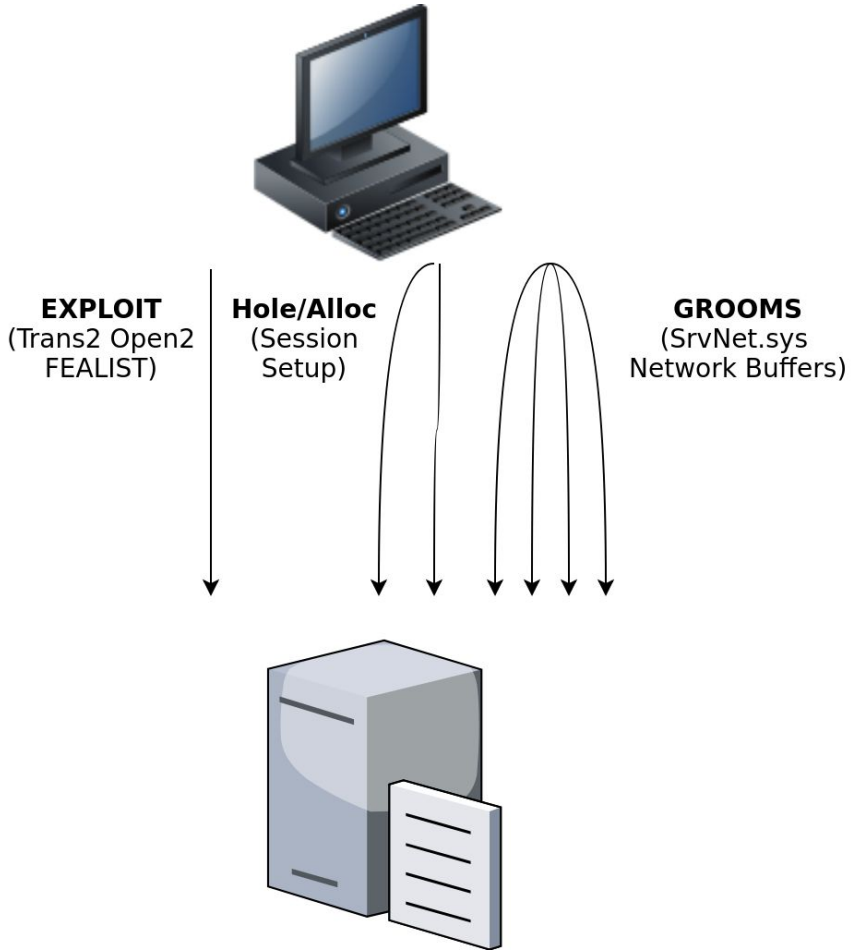
What has changed from XP?

- Simultaneous x86 and x64
- NPP Allocation direction
  - Sequential allocations go from low to high addresses
- Windows Firewall
  - Technically, XP SP2
  - Doesn't matter in many configs (e.g. domain)
- NT 6.0 - HAL heap still static, but:
  - Boot KPCR not static 0xffdff000
    - IdleFunction ASLR
- SrvNet.sys Library
  - Handles lookasides
  - Handles networking (WSK - ports 139/445)
    - Srv.sys
    - Srv2.sys (SMB2/3)

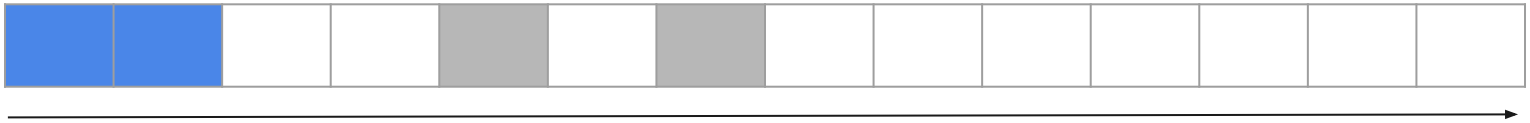


Win7 NPP Groom

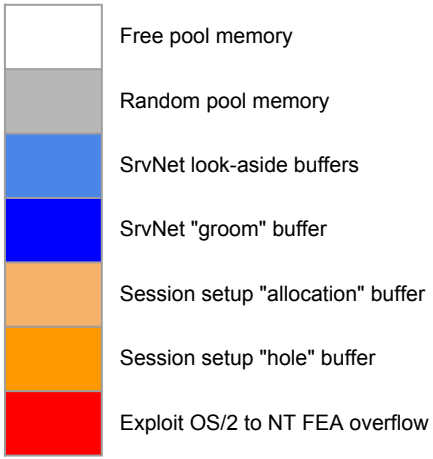
- “Exploit”
  - FEALIST overflow
- “Allocation” / “Hole”
  - Session Setup bug
- Primary / Secondary “Grooms”
  - SrvNet.sys network buffers/struct
  - FAKE SMB2
    - IDS bypass?



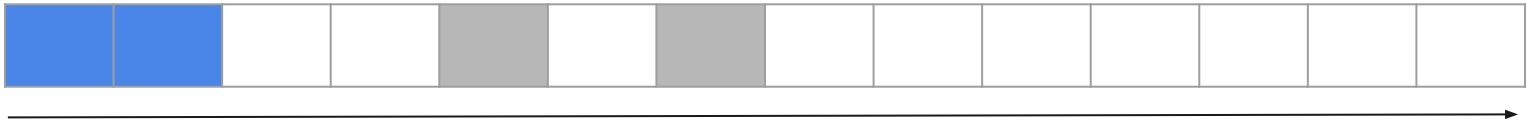
Win7 EternalBlue Grooming



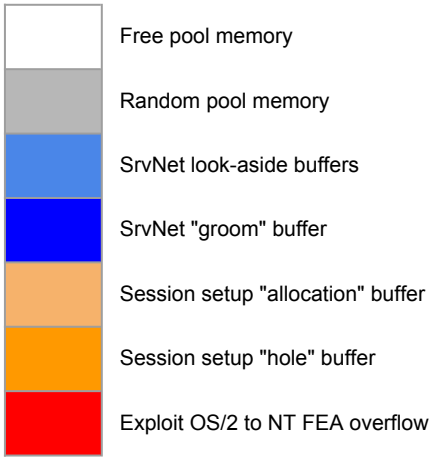
- **Step 0.** Pre-Exploitation Memory Layout
  - SrvNet has lookaside memory, random stuff is in the pool



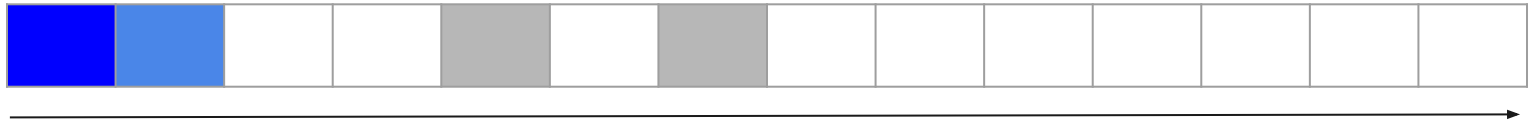
Win7 EternalBlue Grooming



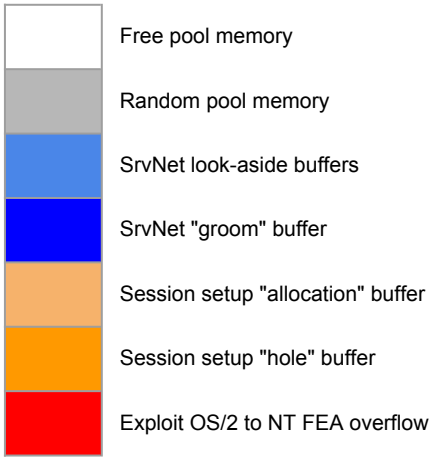
- **Step 1.** Send all of FEALIST except last Trans2 secondary
  - The NT FEA Buffer will not be reserved yet



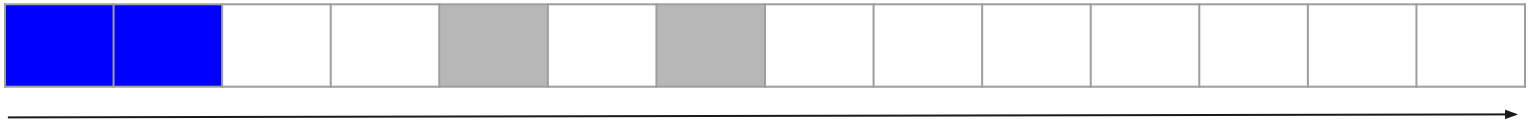
Win7 EternalBlue Grooming

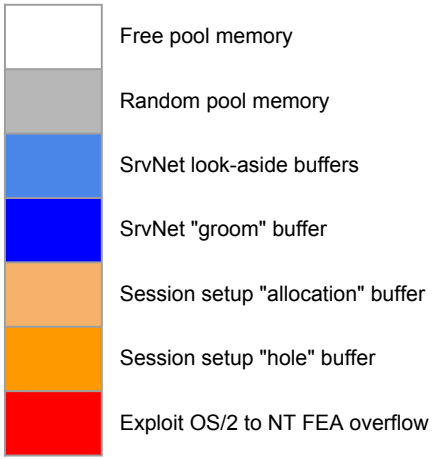


- **Step 2.** Send initial *N* grooms
  - Use up all of SrvNet look-aside, forcing new pool allocations



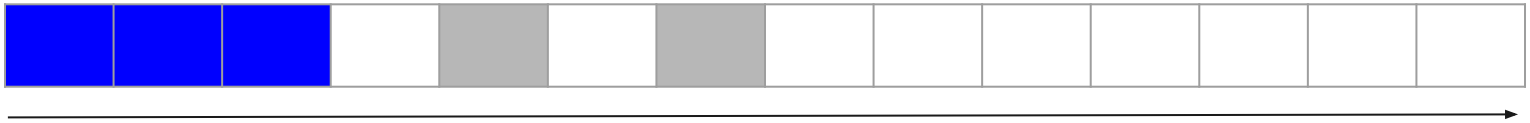
Win7 EternalBlue Grooming



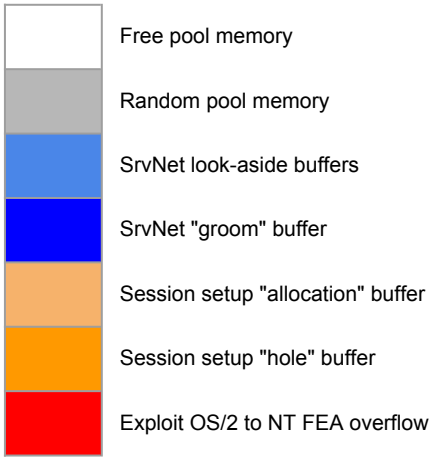


- **Step 2.** Send initial *N* grooms
  - Use up all of SrvNet look-aside, forcing new pool allocations

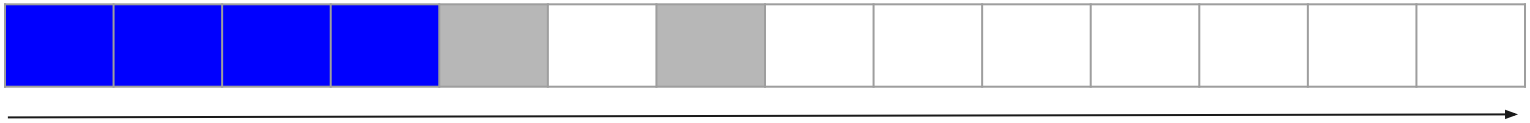
Win7 EternalBlue Grooming



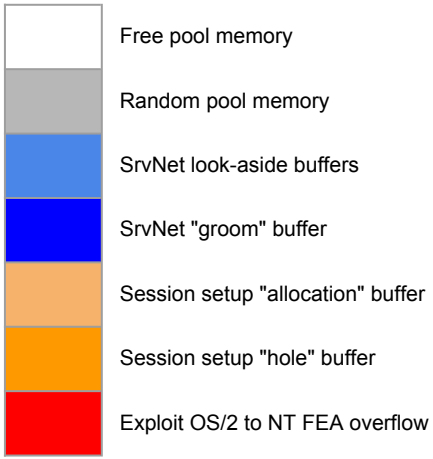
- **Step 2.** Send initial *N* grooms
  - Use up all of SrvNet look-aside, forcing new pool allocations



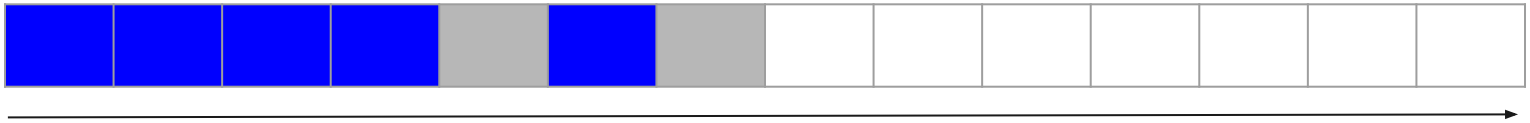
Win7 EternalBlue Grooming

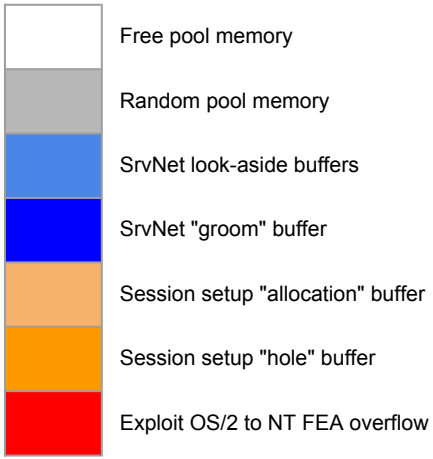


- **Step 2.** Send initial *N* grooms
  - Use up all of SrvNet look-aside, forcing new pool allocations



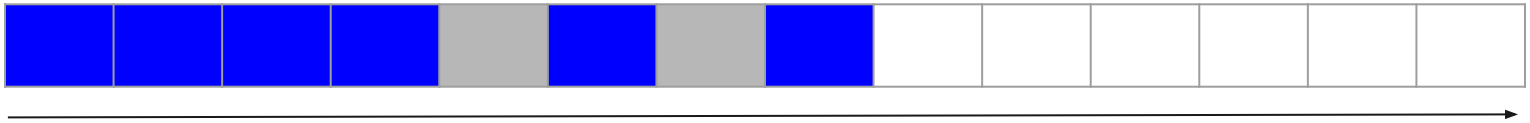
Win7 EternalBlue Grooming

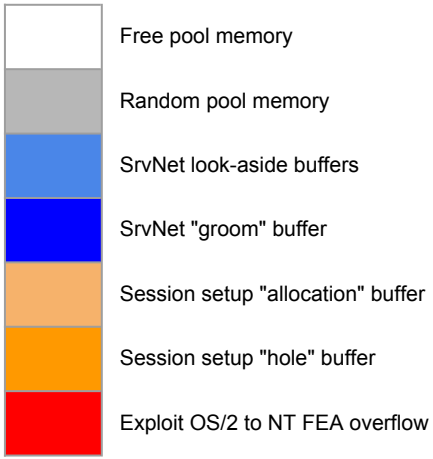




- **Step 2.** Send initial *N* grooms
  - Use up all of SrvNet look-aside, forcing new pool allocations

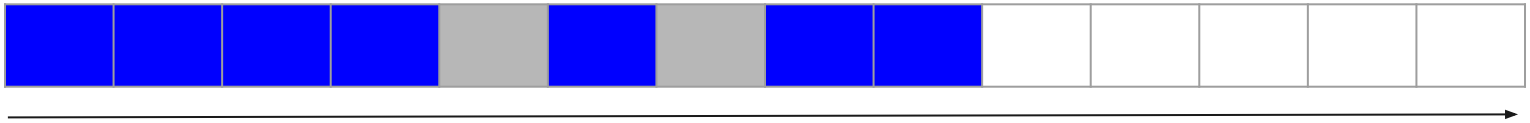
Win7 EternalBlue Grooming



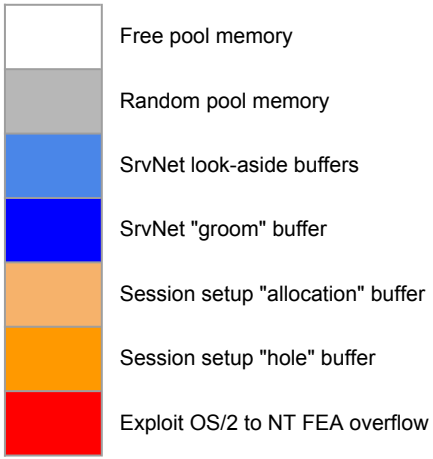


- **Step 2.** Send initial *N* grooms
  - Use up all of SrvNet look-aside, forcing new pool allocations

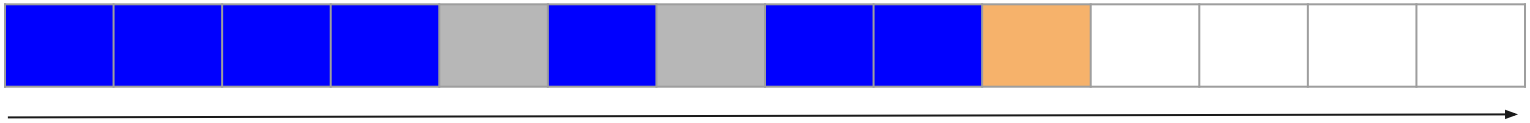
Win7 EternalBlue Grooming

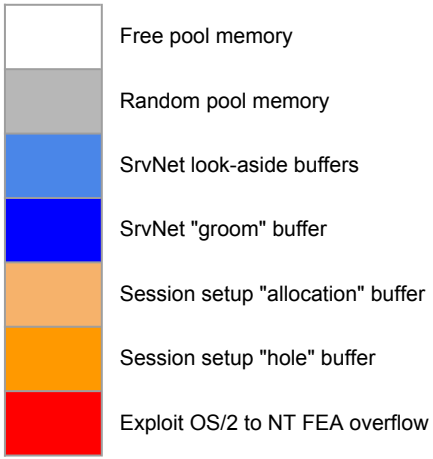


- **Step 2.** Send initial *N* grooms
  - Use up all of SrvNet look-aside, forcing new pool allocations



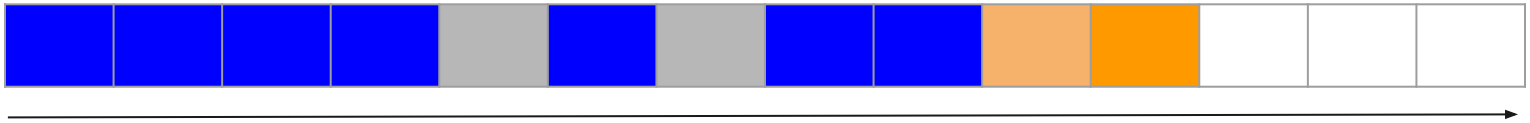
Win7 EternalBlue Grooming



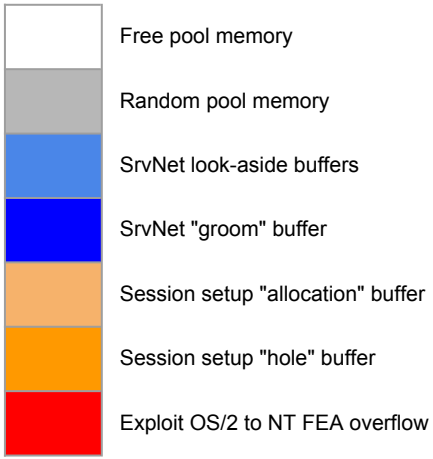


- **Step 3.** Send allocation connection
  - Session Setup bug SMALLER than NT FEA Buffer Size

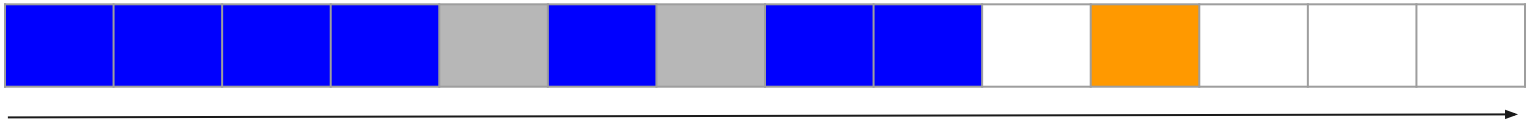
Win7 EternalBlue Grooming

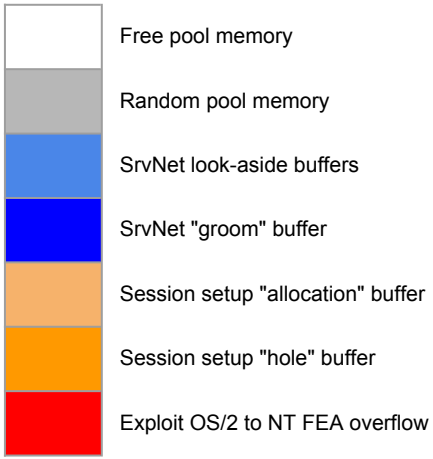


- **Step 4.** Send hole buffer connection
  - Session Setup bug SAME SIZE as NT FEA Buffer Size



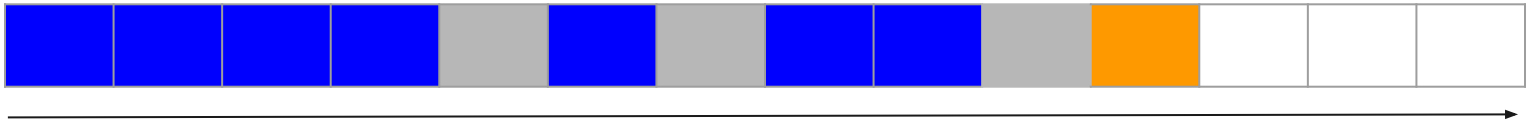
Win7 EternalBlue Grooming



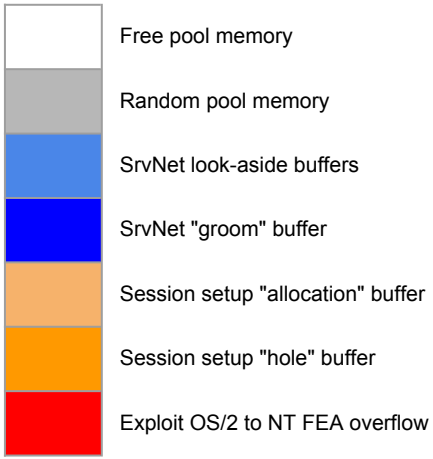


- **Step 5.** Close allocation connection
  - Memory slot can now hold smaller miscellaneous allocations

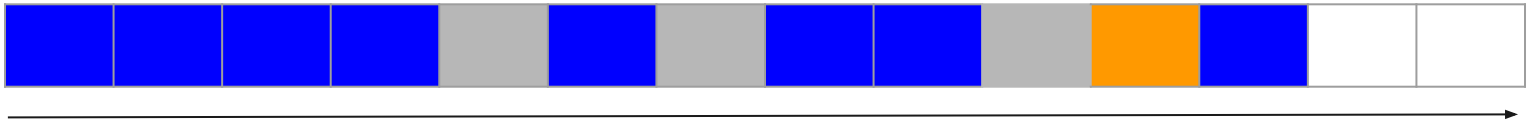
Win7 EternalBlue Grooming

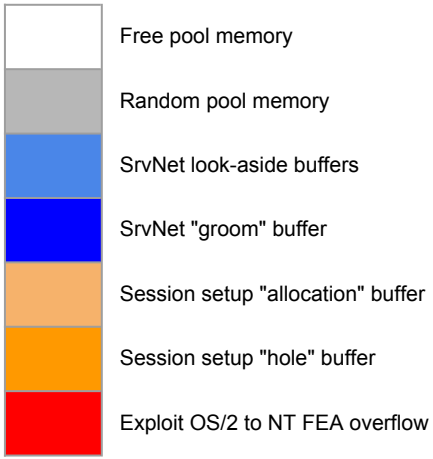


- **Step 5.** Close allocation connection
  - Memory slot can now hold smaller miscellaneous allocations



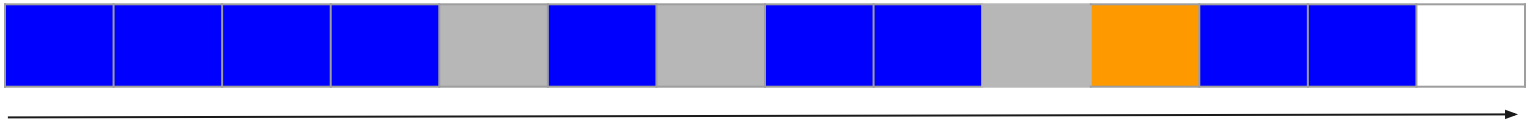
Win7 EternalBlue Grooming

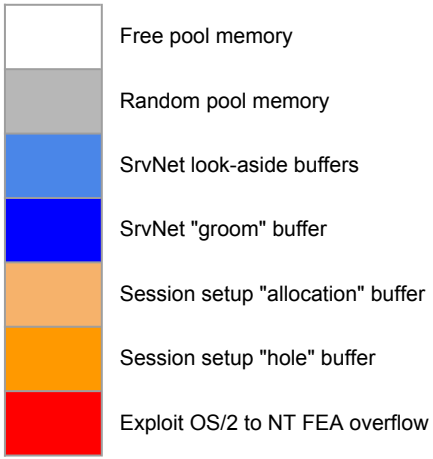




- **Step 6.** Send final groom packets
  - Hopefully a groom is after the Hole buffer

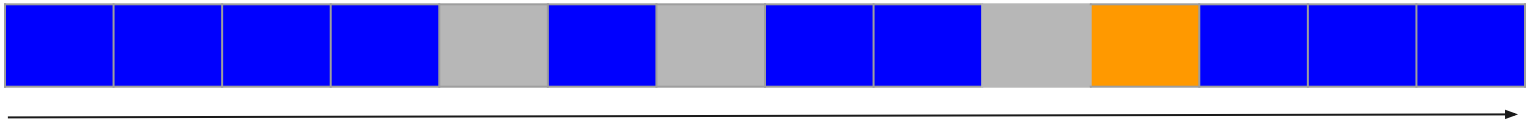
Win7 EternalBlue Grooming

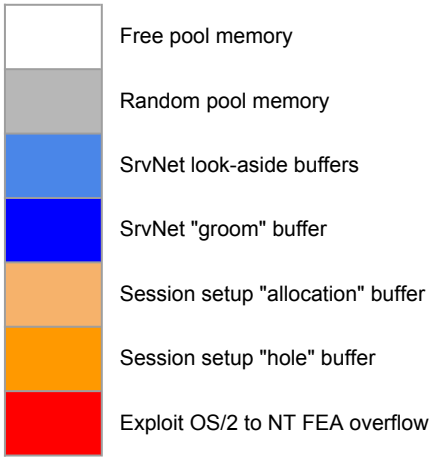




- **Step 6.** Send final groom packets
  - Hopefully a groom is after the Hole buffer

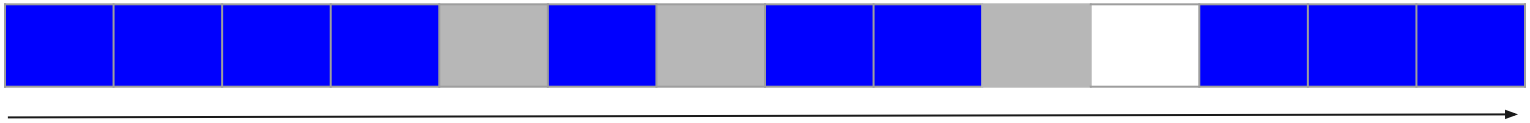
Win7 EternalBlue Grooming



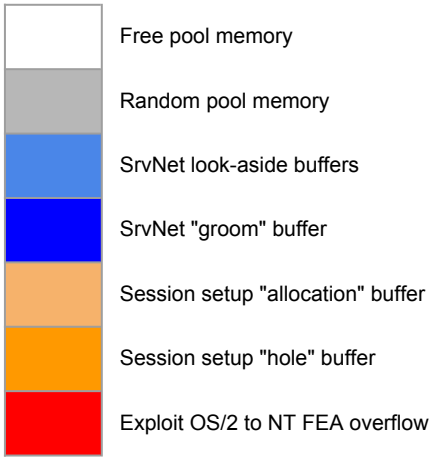


- **Step 6.** Send final groom packets
  - Hopefully a groom is after the Hole buffer

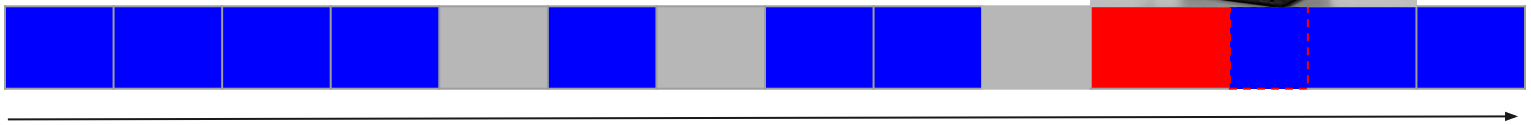
Win7 EternalBlue Grooming



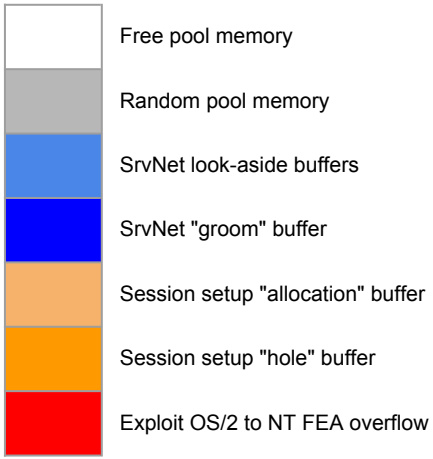
- **Step 7.** Close Hole connection
  - Memory the same size as NT FEA Buffer is now available



Win7 EternalBlue Grooming

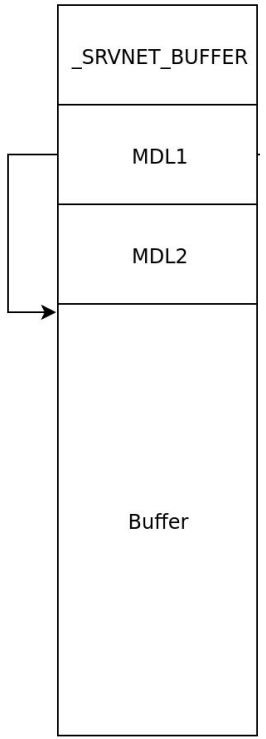


- **Step 8.** Send final FEALIST exploit fragment
  - Erroneously calculated to fit in the free Hole buffer, overflows into groom



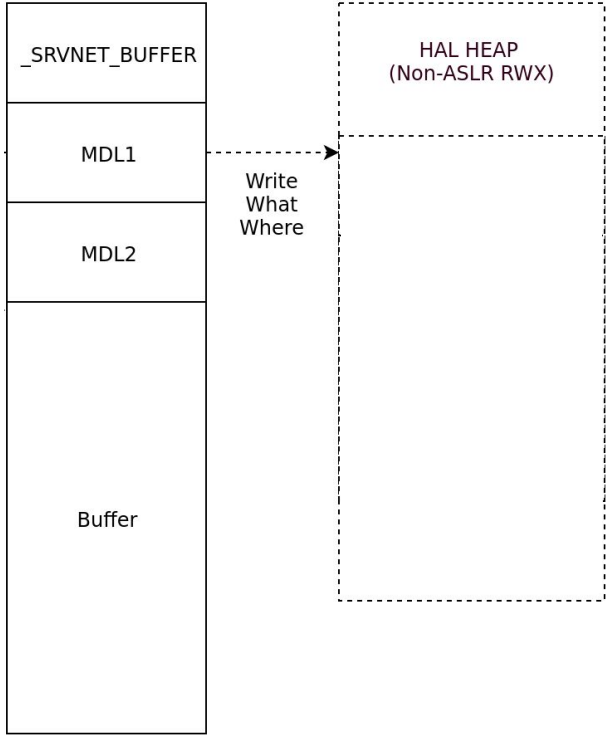
Target of Buffalo Overflow

```
struct _SRVNET_BUFFER_HDR
{
    // …

    SRVNET_WSK_STRUCT  *WskContext;

    // …

    MDL        MDL1;     // MapSysVa = &Buffer
    MDL        MDL2;

    CHAR       Buffer[];
};
```



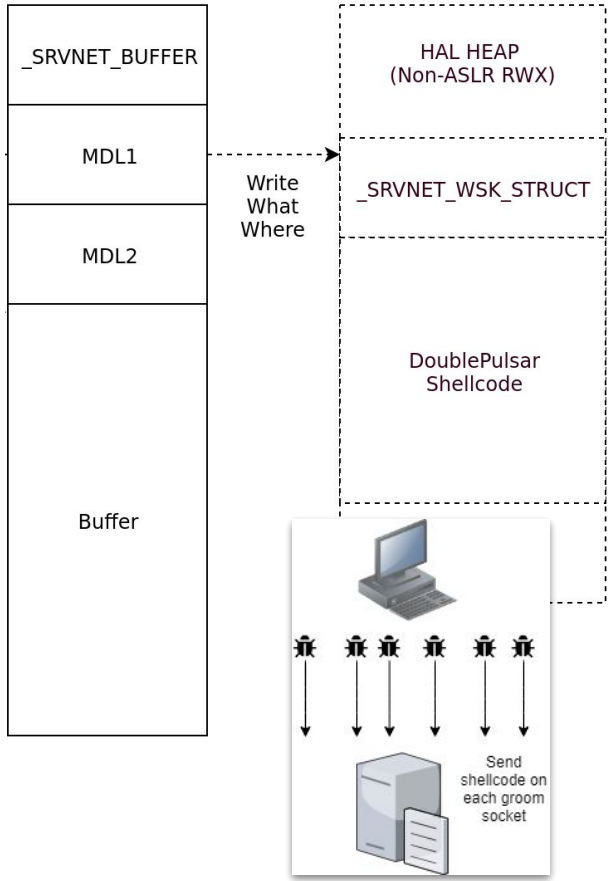
Target of Buffalo Overflow

```
struct _SRVNET_BUFFER_HDR
{
    // …

    SRVNET_WSK_STRUCT  *WskContext;

    // …

    MDL        MDL1;     // MapSysVa = &HAL
    MDL        MDL2;

    CHAR       Buffer[];
};
```



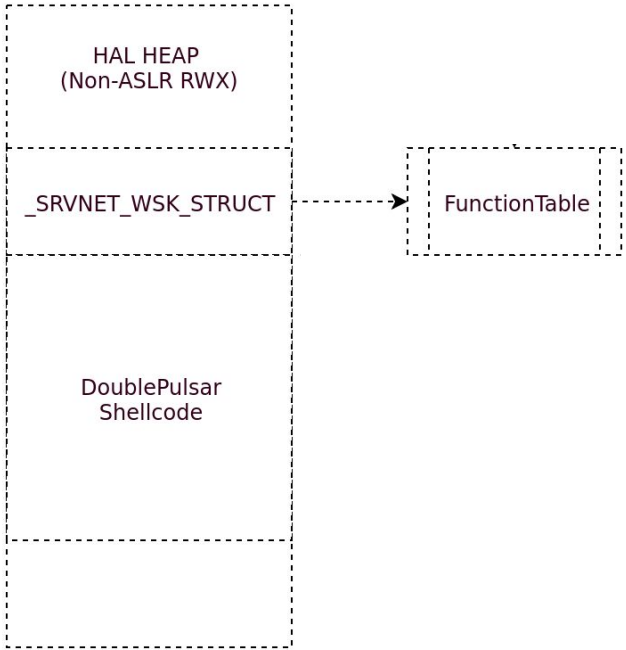
Target of Buffalo Overflow

```
struct _SRVNET_BUFFER_HDR
{
     // …

     SRVNET_WSK_STRUCT  *WskContext;

     // …

     MDL        MDL1;     // MapSysVa = &HAL
     MDL        MDL2;

     CHAR       Buffer[];
};
```



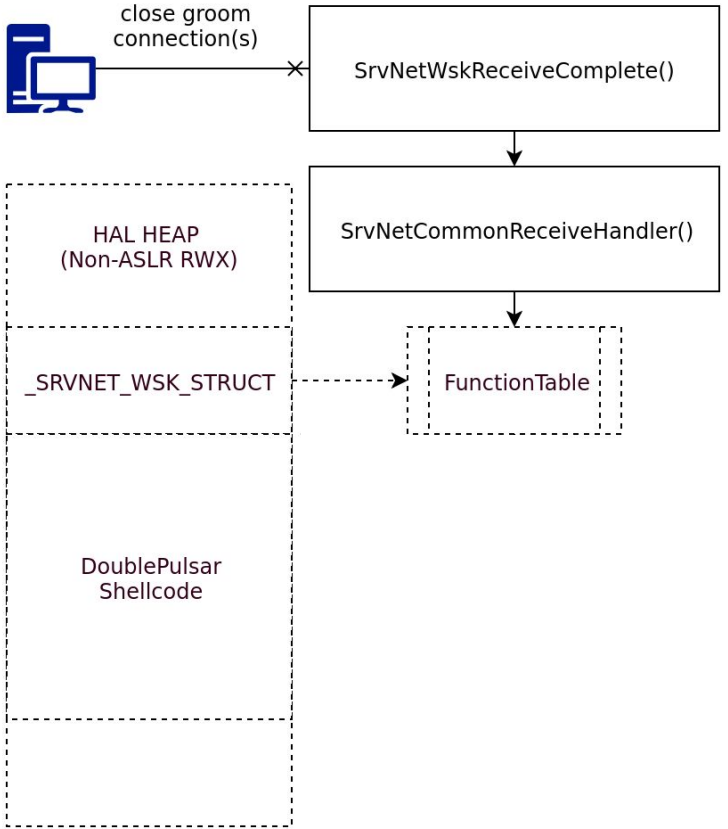
Hijack Execution

```
struct _SRVNET_WSK_STRUCT
{
    // ...

    PVOID     FunctionTable[];

    // ...
};
```



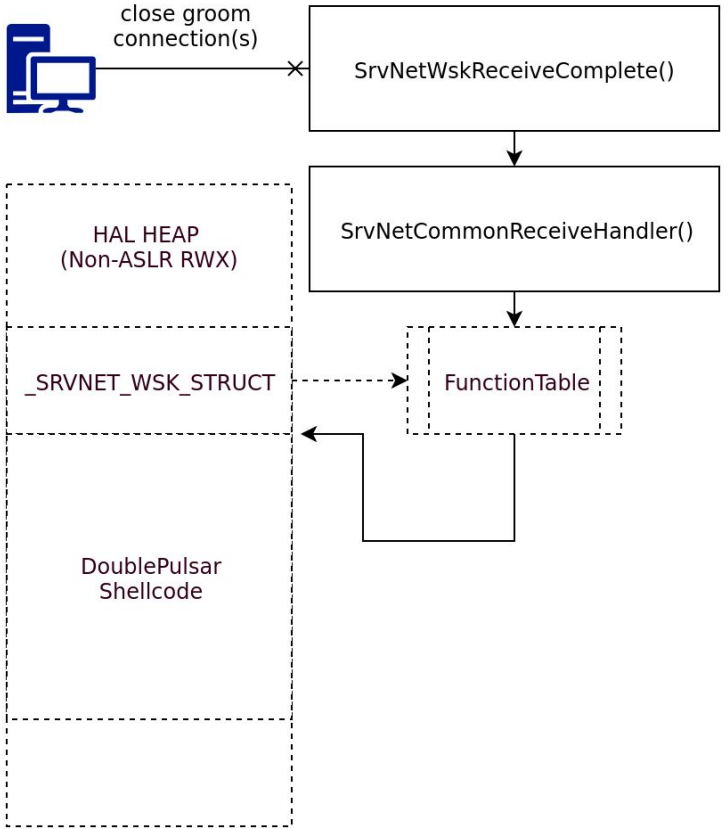
Hijack Execution

```
struct _SRVNET_WSK_STRUCT
{
    // ...

    PVOID     FunctionTable[];

    // ...
};
```

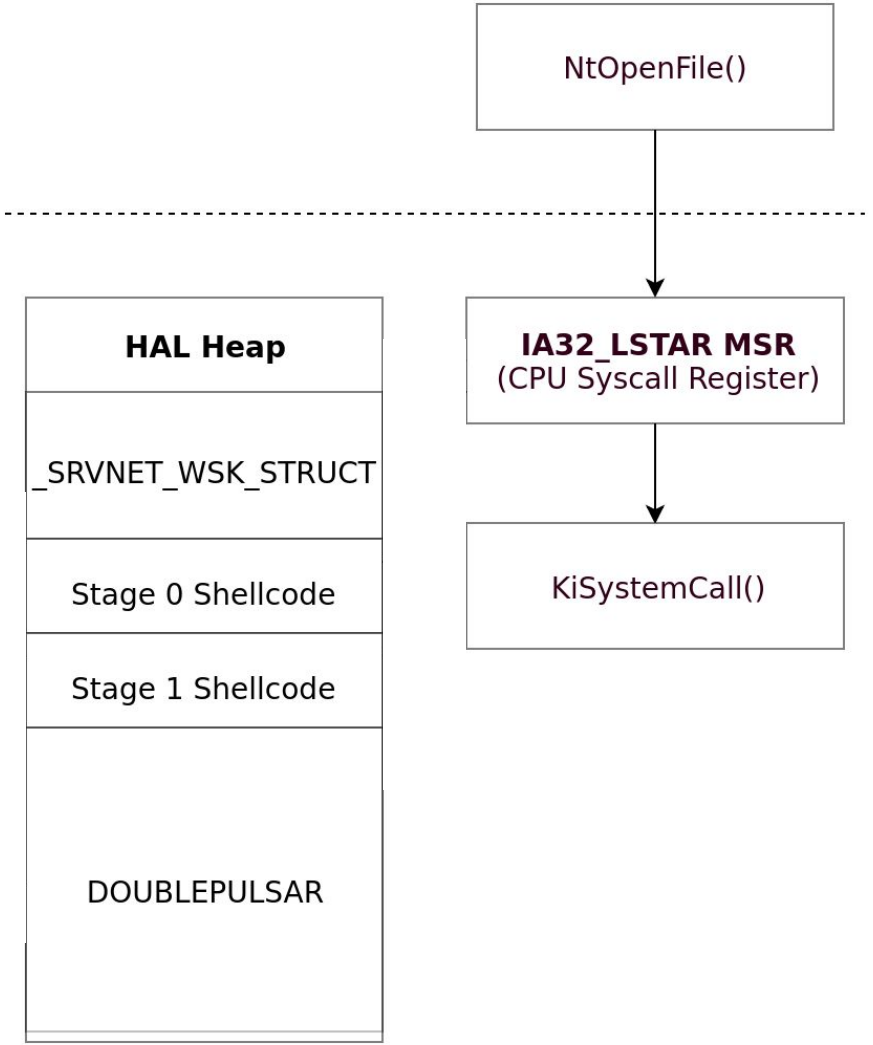


Hijack Execution

```
struct _SRVNET_WSK_STRUCT
{
    // ...

    PVOID     FunctionTable[];

    // ...
};
```



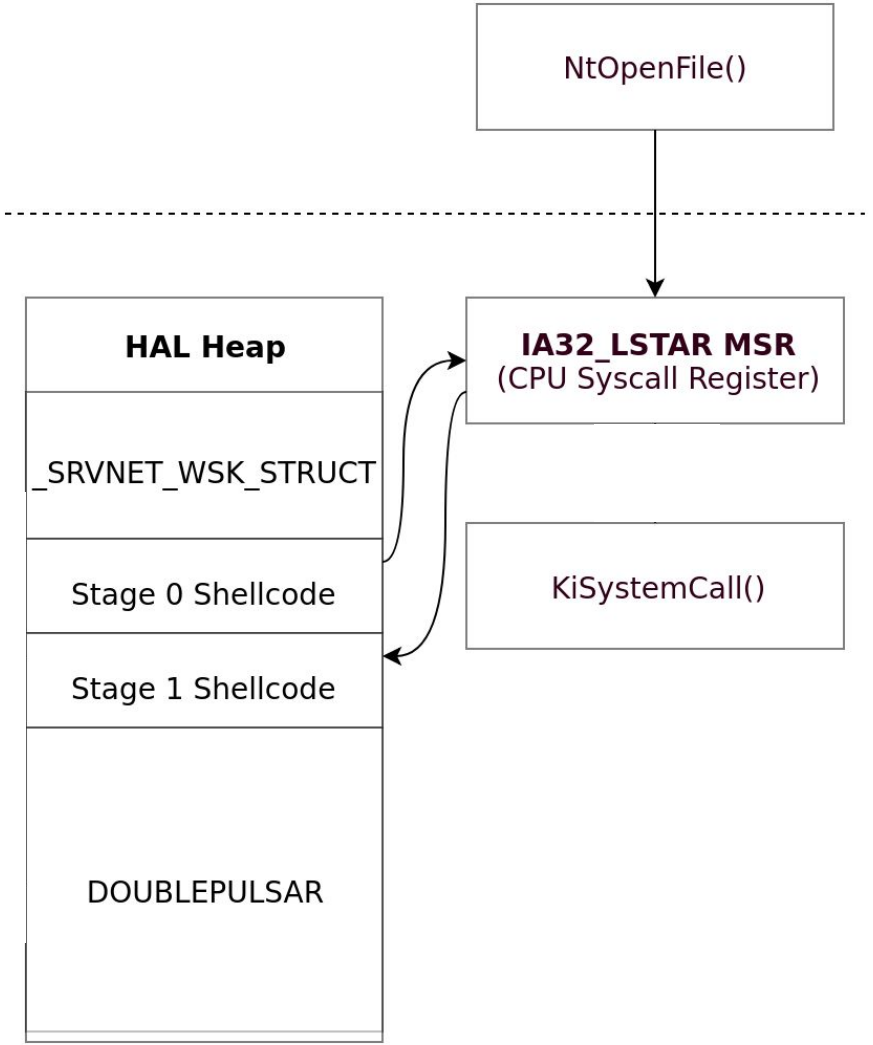
IRQL=DISPATCH_LEVEL



IRQL=DISPATCH_LEVEL

Warning! KPP

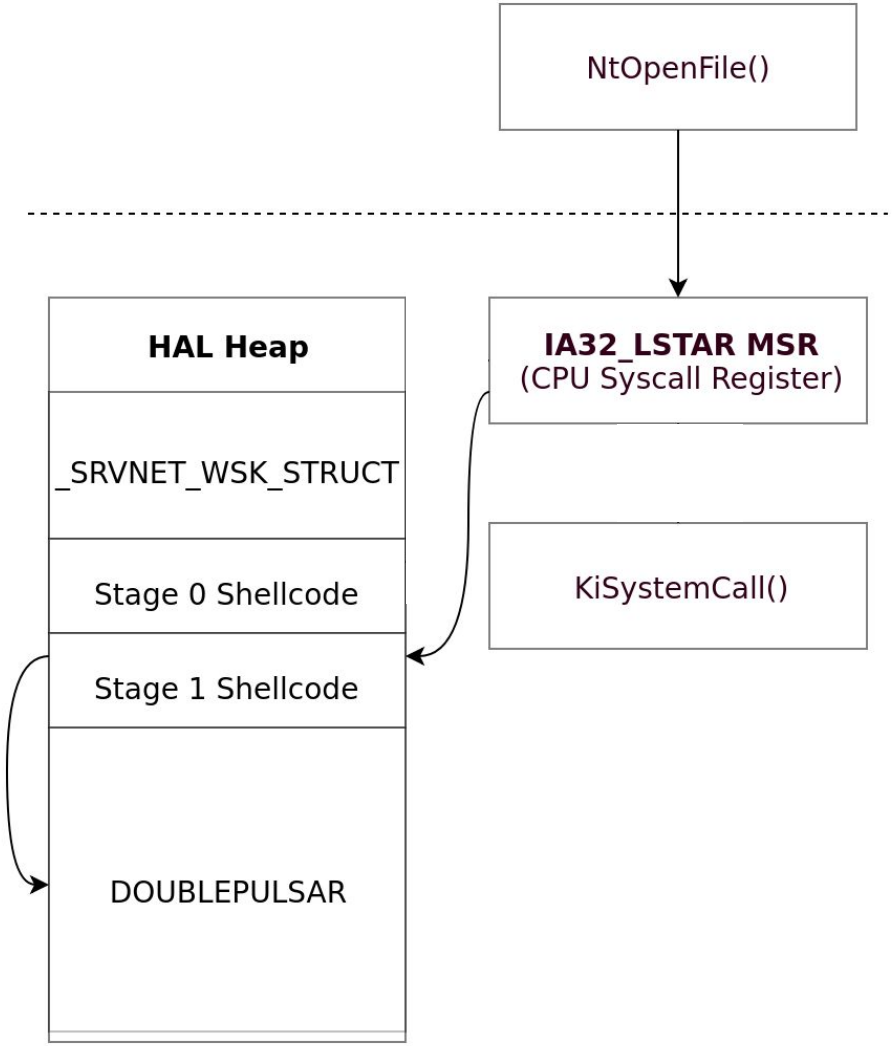
BugCheck CRITICAL_STRUCTURE_CORRUPTION



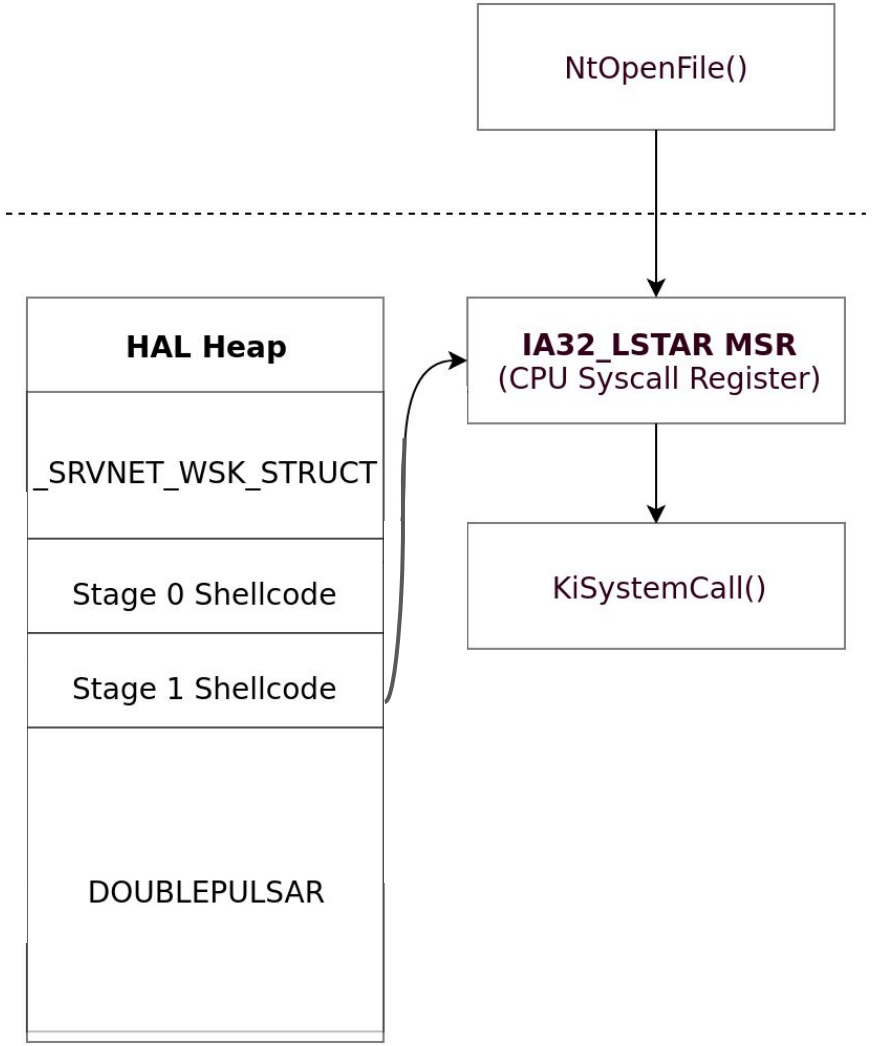
IRQL=PASSIVE_LEVEL

Warning! KPP

BugCheck CRITICAL_STRUCTURE_CORRUPTION



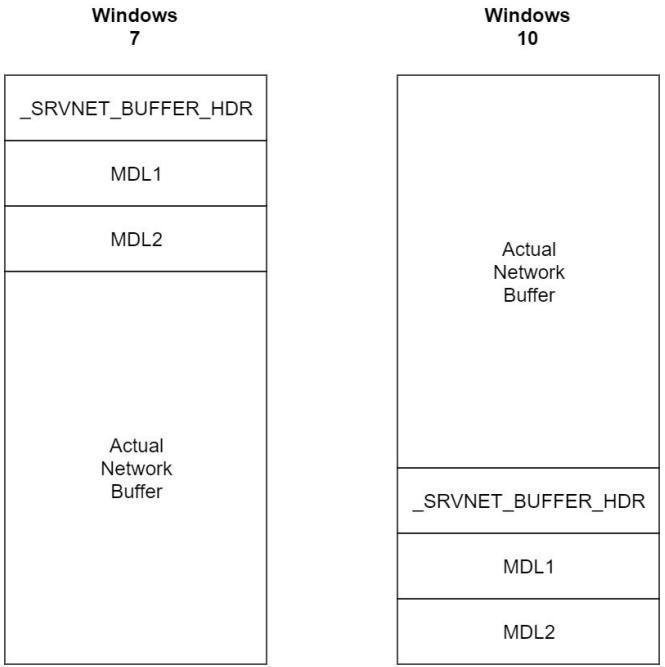
IRQL=PASSIVE_LEVEL



ETERNALBLUE
Win10 (8+)

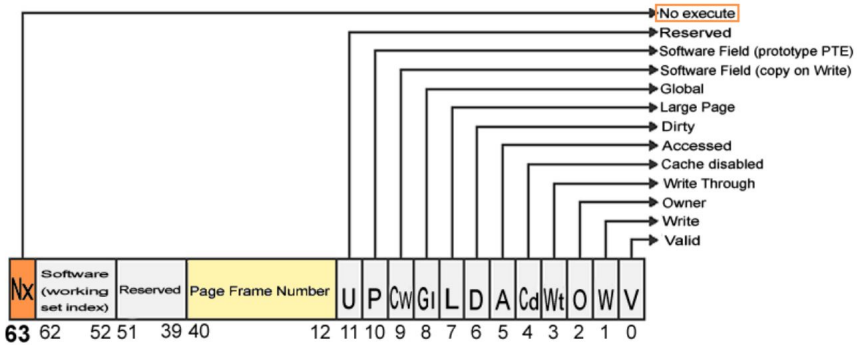
What has changed from Win7?

- Anonymous login to IPC$ blocked
  - Still accessible in many misconfigurations
- Overwrite target now after buffer
  - SRVNET_BUFFER_HDR/MDL1
  - More to buffalo overflow
    - Higher risk to crash?!
- HAL Heap is NX (DEP)
  - Win7 shellcode location



HAL Heap DEP/NX Bypass

- "Bypassing Windows 10 kernel ASLR (remote)"
  - Stefan Le Berre @Heurs
- No ASLR
  - HAL heap
  - PTE for HAL heap
- **Run the buffalo overflow twice!**
  - `MDL1 = &PTE`
  - Overwrite bit63 (NX) with 0!
  - Then, same as Win7




```
3: kd> !pte 0xffffffffffd01000
                                           VA ffffffffffd01000
PXE at FFFFF6FB7DBEDFF8    PPE at FFFFF6FB7DBFFFF8    PDE at FFFFF6FB7FFFFFF0    PTE at FFFFF6FFFFFFE808
contains 000000000111E063  contains 000000000111F063  contains 0000000001120063  contains 8000000000002963
pfn 111e      ---DA--KWEV  pfn 111f      ---DA--KWEV  pfn 1120      ---DA--KWEV  pfn 2         -G-DA--KWEV
```

1000000000000000000000000000000000000000000000000101001011000011

https://github.com/worawit/MS17-010

Outcome

Exploit Chain Differences

	XP	Vista+
Vulnerability	`srv!SrvOs2FeaListSizeToNt`	
Overflow	`srv!SrvOs2FeaToNt`	
Target	`srv!_SRV_POOL_HEADER`	`srvnet!_SRVNET_RECV_BUFFER`
Hijack	`srv!SrvInterlockedFree`	`srvnet!SrvNetCommonReceiveHandler`
Shellcode Addr	`srv!_ClientInfo (`**`Target`**`)`	`hal!_HalHeap (`**`W-W-W`**`)`
DOUBLEPULSAR*	`nt!PopProcessorIdle/` `PopIdle0`	`nt!KiSystemCall`

* DOPU runs at PASSIVE_LEVEL in place of these hijacked functions

Bug #1 Patch - Incorrect Pointer Cast

SrvOs2FeaListSizeToNt():

```
SmbPutUshort(&FeaList->cbList, PTR_DIFF_SHORT(fea, FeaList));
```

Bug #1 Patch - Incorrect Pointer Cast

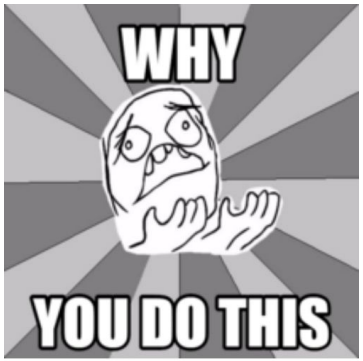
SrvOs2FeaListSizeToNt():

```
SmbPutUlong (&FeaList->cbList, PTR_DIFF_LONG (fea, FeaList));
```

Bug #1 Patch - Incorrect Pointer Cast

SrvOs2FeaListSizeToNt():

```
SmbPutUlong (&FeaList->cbList, PTR_DIFF_LONG (fea, FeaList));
```



Bug #2 Patch - Session Setup Type Confusion

Bug #2 Patch - Session Setup Type Confusion

This slide intentionally left blank

Bug #3 Patch - Oversized Trans2 Dispatch

SrvSmbNtTransaction/SrvSmbTransaction()

```
SrvAllocateTransaction(&Transaction, ...)
Transaction->SecondaryCommand = SMB_COM_NT_TRANS_SECONDARY;      /* &0x38 */
SrvInsertTransaction(&Transaction);
```

SrvFindTransaction()

```
if (FoundTrans->SecondaryCommand != IncomingSmb->Command)
     return NULL;
```

Next-Gen Mitigations (Today)

- Server service permissions revamps
  - Blocked Anonymous Logins by default
  - Death of SMBv1 (30 year NSA pentest?)
- More DEP/NX
  - Srv.sys/NT.exe audited over time
- Full(er) NT KASLR
  - Randomized PTEs
    - Win10 Redstone 1 (1607)
  - HAL heap
    - Win10 Redstone 2 (1703)
- HVCI/VBS
  - **Hyper Guard** (no touchy syscall MSR)
  - **kCFG** (check hijack pointers)

https://www.blackhat.com/docs/us-16/materials/us-16-Weston-Windows-10-Mitigation-Improvements.pdf

Next-Gen Next-Gen Mitigations (Tomorrow)

- Hardware Shadow Stacks (CFI)
  - Stack overflows
  - Shellcode continuation re-design
- Pool Allocation Order Randomization?
  - OS design likes tight allocations
  - **Research:** deal with fragmentation/critical allocations

https://github.com/nsacyber/Control-Flow-Integrity

References

1. http://blogs.360.cn/post/nsa-eternalblue-smb.html
2. https://github.com/worawit/MS17-010
3. https://hitcon.org/2017/CMT/slide-files/d2_s2_r0.pdf
4. https://keybase.pub/jennamagius/EternalBlue_RiskSense-Exploit-Analysis-and-Port-to-Microsoft-Windows-10.pdf
5. https://preview.tinyurl.com/eedefcon
6. https://swithak.github.io/SH20TAATSB18/Home/

Thanks--

Nicolas Joly, pgboy1988, Worawit Wang

Alex Ionescu, Matt Miller, Matt Suiche

MS17-010?

Evolution of Exploitation and Windows Security

zerosum0x0

RISKSENSE

DerbyCon VIII EVOLUTION

LOUISVILLE, KY - 2018