

## Three-Way Handshake TCP tramite Wireshark

L'obiettivo dell'esercitazione odierna era catturare, analizzare e comprendere il processo di connessione TCP noto come **Three-way handshake**. Il tutto è stato eseguito in un ambiente virtuale configurato con **Mininet**, utilizzando strumenti quali **Wireshark** e **tcpdump** per monitorare il traffico generato **tra un client e un server**.

### Scenario Operativo e Preparazione degli Host

Lo svolgimento è stato effettuato sulla macchina virtuale "**CyberOps**" configurata come ambiente di simulazione. **Mininet** è stato utilizzato per creare una topologia di rete emulata, comprendente due nodi principali:

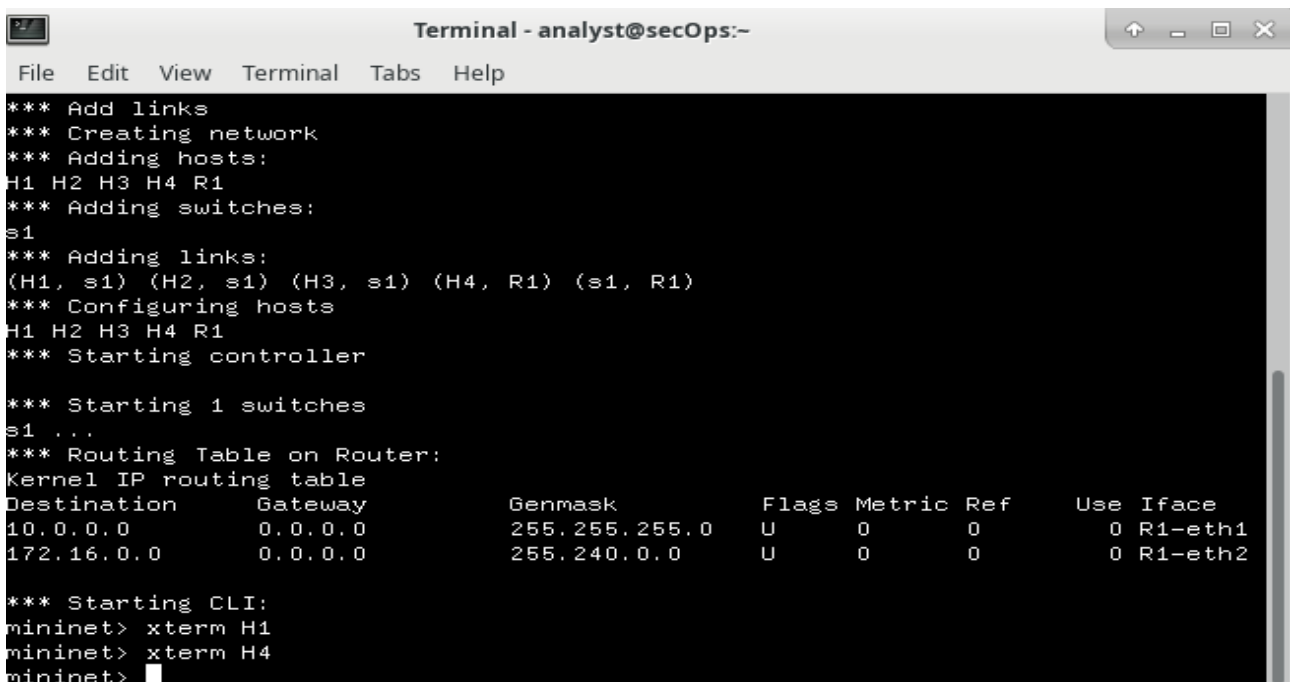
- **-H1: configurato come client.**
- **-H4: configurato come server web.**

### Configurazione

È stata avviata la macchina virtuale **CyberOps** e si è effettuato l'accesso utilizzando le credenziali fornite (*username: **`analyst`**; password: **`cyberops`***).

Da qui è possibile avviare Mininet tramite CLI mediante il comando appropriato (**`sudo mn`**), consentendo la simulazione di una topologia di rete virtuale.

Successivamente è stata effettuata l'inizializzazione degli host H1 e H4, attivati con i comandi di **Mininet** specifici (**`xterm H1`** e **`xterm H4`**), aprendo terminali dedicati per ciascun host.



```
Terminal - analyst@secOps:~
File Edit View Terminal Tabs Help

*** Add links
*** Creating network
*** Adding hosts:
H1 H2 H3 H4 R1
*** Adding switches:
s1
*** Adding links:
(H1, s1) (H2, s1) (H3, s1) (H4, R1) (s1, R1)
*** Configuring hosts
H1 H2 H3 H4 R1
*** Starting controller
*** Starting 1 switches
s1 ...
*** Routing Table on Router:
Kernel IP routing table
Destination      Gateway         Genmask         Flags Metric Ref    Use Iface
10.0.0.0          0.0.0.0         255.255.255.0   U        0      0        0 R1-eth1
172.16.0.0        0.0.0.0         255.240.0.0     U        0      0        0 R1-eth2

*** Starting CLI:
mininet> xterm H1
mininet> xterm H4
mininet> 
```

H4 è stato configurato come server web eseguendo lo script:

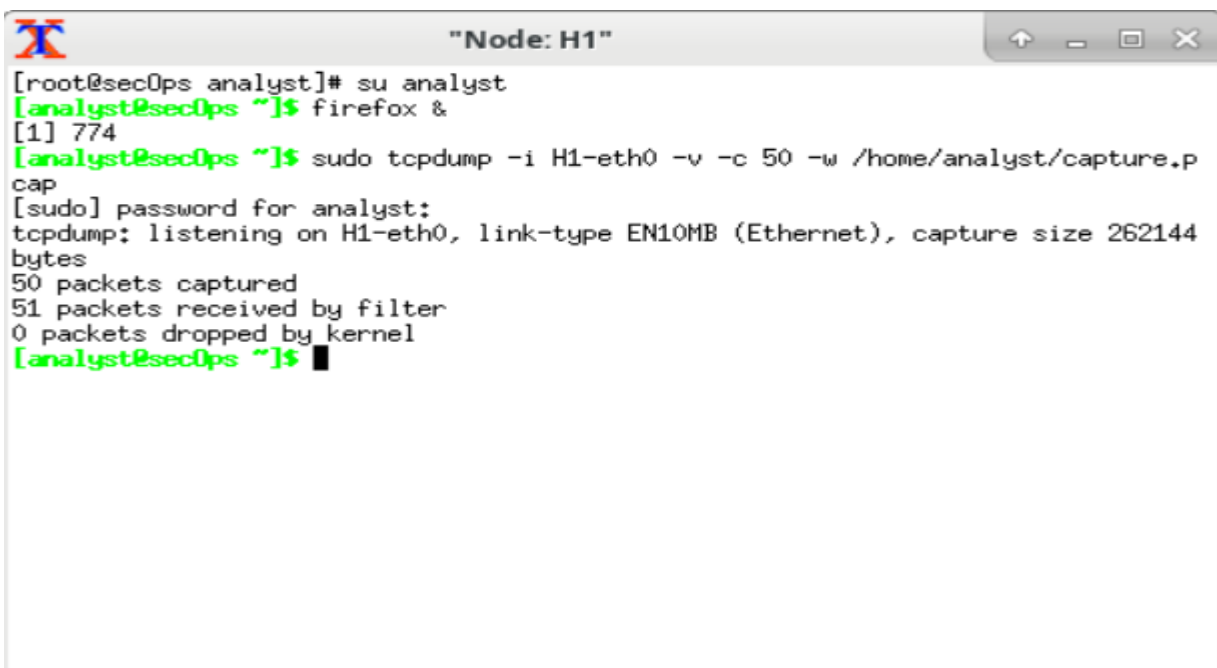
(`/home/analyst/lab.support.files/scripts/reg_server_start.sh`). Esso ha avviato un server HTTP locale all'indirizzo IP **172.16.0.40**.



```
[root@secOps analyst]# /home/analyst/lab.support.files/scripts/reg_server_start.sh
[root@secOps analyst]# 2024/12/11 08:20:54 [error] 770#770: *1 open() "/usr/share/nginx/html/favicon.ico" failed (2: No such file or directory), client: 10.0.0.11, server: localhost, request: "GET /favicon.ico HTTP/1.1", host: "172.16.0.40"
```

Su H1 invece, è stato eseguito un cambio utente dal superutente (**root**) all'utente **analyst** per motivi di sicurezza (`su analyst`).

Successivamente è stato avviato il browser Firefox per simulare il comportamento del client web.



```
[root@secOps analyst]# su analyst
[analyst@secOps ~]$ firefox &
[1] 774
[analyst@secOps ~]$ sudo tcpdump -i H1-eth0 -v -c 50 -w /home/analyst/capture.pcap
[sudo] password for analyst:
tcpdump: listening on H1-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
50 packets captured
51 packets received by filter
0 packets dropped by kernel
[analyst@secOps ~]$
```

## Generazione e cattura del traffico su H1

Prima di inviare richieste al server, è stata avviata una sessione di tcpdump per catturare il traffico di rete.

Il comando eseguito è stato:

```
sudo tcpdump -i H1-eth0 -v -c 50 -w /home/analyst/capture.pcap
```

Con questo comando è stato configurato **tcpdump** al fine di monitorare l'interfaccia di rete di H1 (`H1-eth0`), catturando un massimo di 50 pacchetti e salvandoli in un file PCAP denominato `capture.pcap`.

Una volta avviata la cattura, è stato utilizzato il browser su H1 per accedere all'indirizzo IP del server web (172.16.0.40). Questa azione ha generato traffico HTTP, includendo il **TWH** tra il client e il server.

No.	Time	Source	Destination	Protocol	Length	Info
15	4.094595	10.0.0.11	172.16.0.40	TCP	74	56390 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3425260717 TSecr=0 WS=51
16	4.094704	172.16.0.40	10.0.0.11	TCP	74	80 → 56390 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=1918562858 TSe
17	4.094725	10.0.0.11	172.16.0.40	TCP	66	56390 → 80 [ACK] Seq=1 Ack=1 Win=29696 Len=0 TSval=3425260717 TSecr=1918562858
18	4.095327	10.0.0.11	172.16.0.40	HTTP	358	GET /favicon.ico HTTP/1.1
19	4.095364	172.16.0.40	10.0.0.11	TCP	66	80 → 56390 [ACK] Seq=1 Ack=293 Win=30208 Len=0 TSval=1918562859 TSecr=3425260718
20	4.098034	172.16.0.40	10.0.0.11	HTTP	390	HTTP/1.1 404 Not Found (text/html)
21	4.099982	10.0.0.11	172.16.0.40	TCP	66	56390 → 80 [ACK] Seq=293 Ack=325 Win=30720 Len=0 TSval=3425260723 TSecr=1918562862
46	14.147026	10.0.0.11	172.16.0.40	TCP	66	[TCP Keep-Alive] 56390 → 80 [ACK] Seq=292 Ack=325 Win=30720 Len=0 TSval=3425270770 TSecr=19185
47	14.147049	172.16.0.40	10.0.0.11	TCP	66	[TCP Keep-Alive ACK] 80 → 56390 [ACK] Seq=325 Ack=293 Win=30208 Len=0 TSval=1918572911 TSecr=3

## Analisi dei Pacchetti con Wireshark

Dopo aver completato la cattura del traffico, il file **PCAP** è stato analizzato utilizzando **Wireshark**:

- Il file è stato caricato in Wireshark tramite il menu `File > Open`.
- È stato applicato un filtro `tcp` per isolare i pacchetti rilevanti per l'handshake a tre vie.

Questa analisi ha evidenziato tre pacchetti fondamentali:

- Il client (H1) invia un pacchetto **SYN** al server (H4), indicando l'intenzione di **avviare** una connessione. Il numero di sequenza relativo è impostato a 0.
- Il server risponde con un pacchetto contenente i **flag SYN e ACK**, confermando la ricezione e indicando il proprio numero di sequenza relativo (0). L'acknowledgment è impostato a 1.
- Il client invia un **ACK finale** per completare il **TWH**. A questo punto, la connessione TCP è stabilita.

capture.pcap [Wireshark 2.5.1]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: tcp Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
11	4.979237	10.0.0.11	172.16.0.40	TCP	74	43608 → 80 [SYN] Seq=0 Win=29200 Len=0
12	4.979272	172.16.0.40	10.0.0.11	TCP	74	80 → 43608 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0
13	4.979278	10.0.0.11	172.16.0.40	TCP	66	43608 → 80 [ACK] Seq=1 Ack=1 Win=29200 Len=0
14	4.979338	10.0.0.11	172.16.0.40	HTTP	377	GET / HTTP/1.1
15	4.979343	172.16.0.40	10.0.0.11	TCP	66	80 → 43608 [ACK] Seq=1 Ack=312 Win=3072 Len=0

► Frame 11: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)

► Ethernet II, Src: 12:cd:75:0a:0c:70 (12:cd:75:0a:0c:70), Dst: 76:ab:13:87:93:2a (76:ab:13:87:93:2a)

► Internet Protocol Version 4, Src: 10.0.0.11, Dst: 172.16.0.40

▼ Transmission Control Protocol, Src Port: 43608, Dst Port: 80, Seq: 0, Len: 0

Source Port: 43608

Destination Port: 80

[Stream index: 0]

[TCP Segment Len: 0]

Sequence number: 0 (relative sequence number)

[Next sequence number: 0 (relative sequence number)]

Acknowledgment number: 0

1010 .... = Header Length: 40 bytes (10)

► Flags: 0x002 (SYN)

Window size value: 29200

[Calculated window size: 29200]

Checksum: 0xb671 [unverified]

[Checksum Status: Unverified]

0000 76 ab 13 87 93 2a 12 cd 75 0a 0c 70 08 00 45 00 v...\*.. u..p..E.

0010 00 3c fb 0c 40 00 40 06 89 6c 0a 00 00 0b ac 10 .<..@.@. !.....

0020 00 28 aa 58 00 50 a9 b5 73 85 00 00 00 00 a0 02 .(X.P.. s.....

0030 72 10 b6 71 00 00 02 04 05 b4 04 02 08 0a 85 bd r..q.....

0040 65 46 00 00 00 00 01 03 03 09 eF.....

## TCPdump

Per confermare i risultati ottenuti con Wireshark, tcpdump è stato utilizzato per leggere il file PCAP.

Tramite il comando:

**tcpdump -r /home/analyst/capture.pcap tcp -c 3**

Ci è stato permesso di visualizzare i primi tre pacchetti **TCP**, corrispondenti al **TWH**.

## Considerazioni

Il laboratorio ha dimostrato l'importanza degli strumenti di analisi del traffico come Wireshark e tcpdump nella comprensione e nel monitoraggio delle comunicazioni TCP.

Ciò ha dimostrato che non solo questi strumenti possono essere utilizzati per analisi di sicurezza, debugging di rete e individuazione di anomalie, ma che possano essere anche cruciali per lo svolgimento di queste mansioni.

Wireshark in modo particolare, è utile per diagnosticare problemi di rete e investigare attacchi informatici, mentre tcpdump offre un'alternativa più leggera per scenari che richiedono scripting o automazione.

Prendere confidenza con questi strumenti è quindi molto importante soprattutto per quanto riguarda l'ambito lavorativo, non solo nell'ambito difensivo, parlando quindi di blue team, ma anche per competenze personali, le quali potrebbero tornare utili, senza alcuna ombra di dubbio.