
S11-L2

Wireshark e TCPdump

Emanuele Benedetti | 18 febbraio 2025

Consegna

Utilizzo di wireshark per osservare il TCP three way handshake

Il laboratorio è diviso in tre fasi:

- Preparazione degli host per catturare il traffico
- Analisi dei pacchetti utilizzando Wireshark
- Visualizzazione dei pacchetti utilizzando tcpdump

Il laboratorio è basato sulla traccia seguente:

<https://itexamanswers.net/9-2-6-lab-using-wireshark-to-observe-the-tcp-3-way-handshake-answers.html>

Bonus

Utilizzo di Wireshark per esaminare le catture TCP e UDP

Il laboratorio è incentrato sul raggiungimento di questi obiettivi:

- Identificare i campi dell'intestazione TCP e il funzionamento utilizzando una cattura di sessione FTP in Wireshark.
- Identificare i campi dell'intestazione UDP e il funzionamento utilizzando una cattura di sessione TFTP in Wireshark.

Il laboratorio segue la seguente traccia:

<https://itexamanswers.net/10-4-3-lab-using-wireshark-to-examine-tcp-and-udp-captures-answers.html>

Svolgimento

Preparazione della macchina

Il laboratorio è stato eseguito sulla macchina virtuale *CyberOps VM*.

Ho eseguito il comando `sudo lab.support.files/scripts/cyberops_topo.py` per avviare *Mininet*, una piattaforma open-source che simula reti software-defined (SDN), creando una rete virtuale che include switch e host virtuali.

```
[analyst@secOps ~]$ sudo lab.support.files/scripts/cyberops_topo.py
[sudo] password for analyst:

CyberOPS Topology:

      | R1 |-----| H4 |
      |   |-----|   |
      |   |       |   |
      |   |       |   |
      |---| S1 |---|
      |   |       |   |
      |   |       |   |
      | H1 |   H2 |   H3 |
      |---|   |---|

*** 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
0.0.0.0        0.0.0.0         0.0.0.0         U        0     0        0 R1-eth1
172.16.0.0     0.0.0.0         255.240.0.0     U        0     0        0 R1-eth2

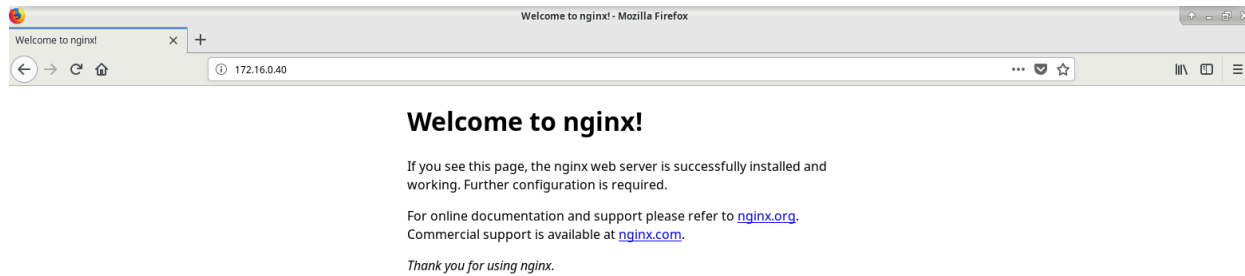
*** Starting CLI:
mininet>
```

Ho prima avviato gli host *H1* e *H4* con `xterm H1` e `xterm H4`, poi il web server su *H4* con `/home/analyst/lab.support.files/scripts/reg_server_start.sh`.

Poiché per ragioni di sicurezza non possiamo eseguire *firefox* come root, sull'host *H1* cambiamo utente come *analyst* con `su analyst` ed avviamo il browser con `firefox &`.

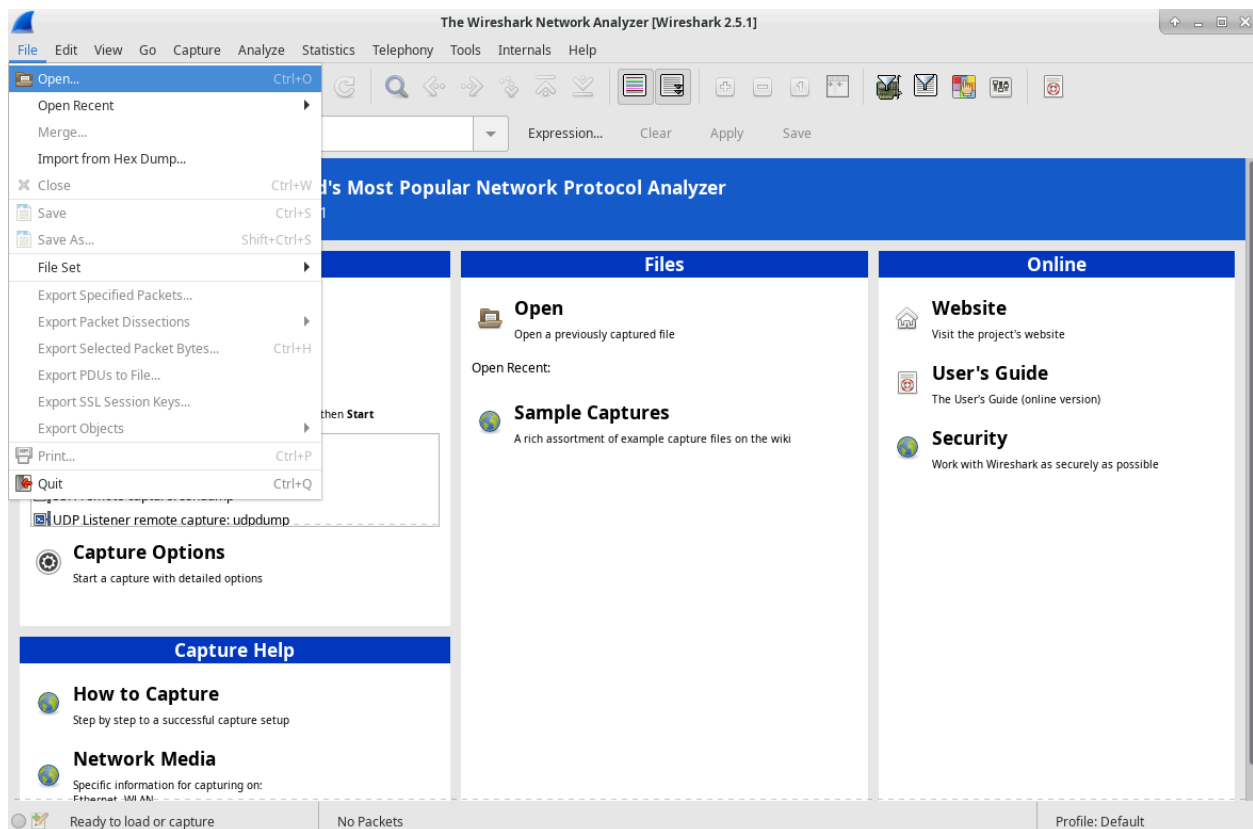
Attendiamo l'apertura della finestra e avviamo una sessione *tcpdump* su *H1*, salvando l'output sul file *capture.pcap* con il comando `sudo tcpdump -i H1-eth0 -v -c 50 -w /home/analyst/capture.pcap`.

Navighiamo con *firefox* all'indirizzo 172.16.0.40



Analisi del traffico con Wireshark

Avviamo *wireshark* per analizzare la cattura effettuata tramite *tcpdump*. Andiamo su *File > Open* e scegliamo il file da analizzare.



Applichiamo il filtro *tcp*. I pacchetti 36, 37 e 38 sono quelli di nostro interesse, in cui viene eseguito il *three way handshake* del protocollo TCP.

No.	Time	Source	Destination	Protocol	Length	Info
36	13.874266	10.0.0.11	172.16.0.40	TCP	74	46838 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2791191893 TSecr=0 WS=512
37	13.874317	172.16.0.40	10.0.0.11	TCP	74	80 → 46838 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=1408201530 TSecr=2791191893 WS=512
38	13.874327	10.0.0.11	172.16.0.40	TCP	66	46838 → 80 [ACK] Seq=1 Ack=1 Win=29696 Len=0 TSval=2791191893 TSecr=1408201530

Il frame 36 avvia la procedura di handshake tra il PC *H1* ed il server *H4*. Possiamo vedere le informazioni del pacchetto tramite il pannello *Packet List* in basso che mostra tutte le informazioni.

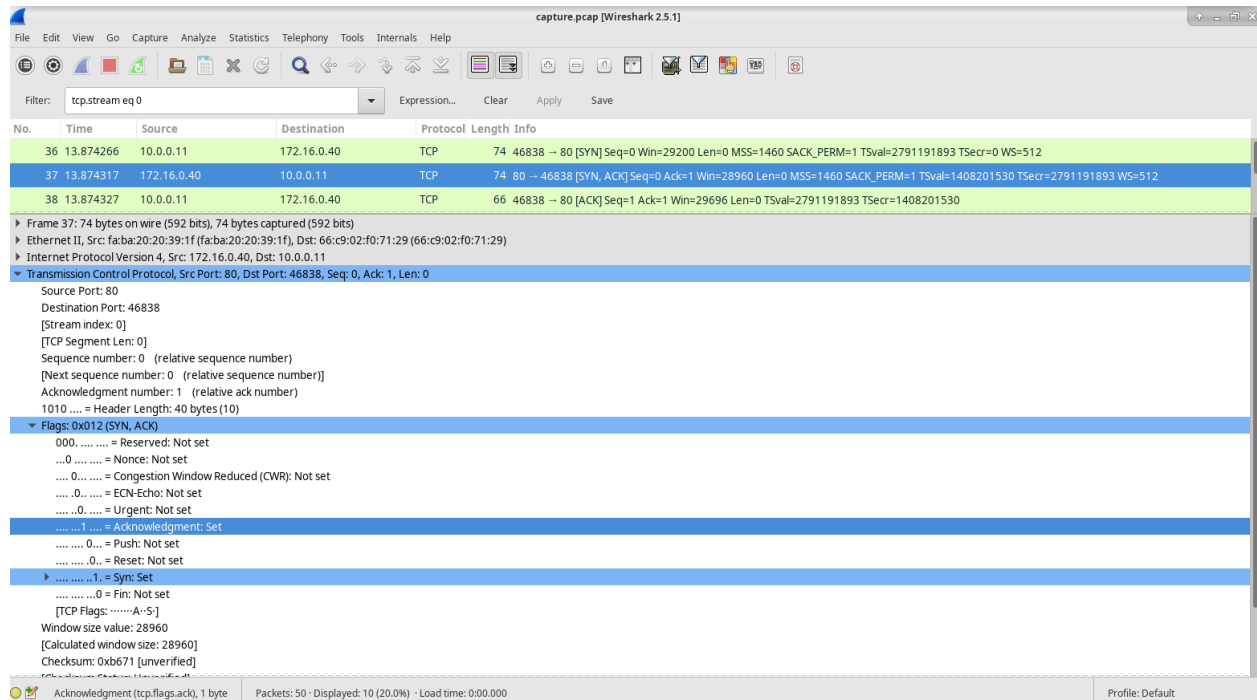
No.	Time	Source	Destination	Protocol	Length	Info
36	13.874266	10.0.0.11	172.16.0.40	TCP	74	46838 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2791191893 TSecr=0 WS=512
37	13.874317	172.16.0.40	10.0.0.11	TCP	74	80 → 46838 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=1408201530 TSecr=2791191893 WS=512
38	13.874327	10.0.0.11	172.16.0.40	TCP	66	46838 → 80 [ACK] Seq=1 Ack=1 Win=29696 Len=0 TSval=2791191893 TSecr=1408201530

Frame 36: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0 Ethernet II, Src: 66:c9:02:f0:71:29 (66:c9:02:f0:71:29), Dst: fa:ba:20:20:39:1f (fa:ba:20:20:39:1f) Internet Protocol Version 4, Src: 10.0.0.11, Dst: 172.16.0.40 Transmission Control Protocol, Src Port: 46838, Dst Port: 80, Seq: 0, Len: 0	
Source Port:	46838
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)	
000. = Reserved:	Not set
...0 = Nonce:	Not set
....0 = Congestion Window Reduced (CWR):	Not set
....0 = ECN-Echo:	Not set
....0 = Urgent:	Not set
....0 = Acknowledgment:	Not set
....0 = Push:	Not set
....0 = Reset:	Not set
.....1. = Syn:	Set
....0 = Fin:	Not set
[TCP Flags:	S]
Window size value:	29200
[Calculated window size:]	29200
Checksum:	0xb671 [unverified]

Nella finestra in basso vengono mostrate tutte le informazioni come indirizzo IP sorgente e destinazione, numeri di porta, indirizzi MAC ecc.

Espandendo le informazioni relative al protocollo TCP ci viene mostrato che il flag *SYN* è settato ad *1*.

Il pacchetto 37 mostra invece la risposta del server verso il client *H1*. In questo caso notiamo che nella sezione TCP, sono impostati ad *1* i flag *SYN* e *ACK*.



Il terzo pacchetto, che chiude l'handshake, ha come da aspettative il flag ACK impostato.

É interessante notare i valori di *sequence number* e *acknowledgment number*: nel primo pacchetto il *sequence number* è pari a 0, il server risponde con un *sequence number* uguale a 0 e *acknowledgment number* uguale a 1 (sequence number ricevuto + 1). Infine il client termina l'handshake con *sequence number* 1 e *acknowledge number* 1 (sequence number ricevuto + 1).

Visualizzazione dei pacchetti con tcpdump

Possiamo aprire ed analizzare il traffico catturato anche tramite tcpdump.

Eseguiamo il comando `tcpdump -r /home/analyst/capture.pcap tcp -c 3` per aprire la precedente cattura.

```
[analyst@secOps ~]$ tcpdump -r /home/analyst/capture.pcap tcp -c 3
reading from file /home/analyst/capture.pcap, link-type EN10MB (Ethernet)
08:31:28.584490 IP secOps.46838 > 172.16.0.40,http: Flags [S], seq 1889512870, win 29200, options [mss 1460,sackOK,TS val 2791191893 ecr 0,nop,wscale 9], length 0
08:31:28.584541 IP 172.16.0.40,http > secOps.46838: Flags [S_], seq 1303723797, ack 1889512871, win 28960, options [mss 1460,sackOK,TS val 1408201530 ecr 2791191893,nop,wscale 9], length 0
08:31:28.584551 IP secOps.46838 > 172.16.0.40,http: Flags [A], ack 1, win 58, options [nop,TS val 2791191893 ecr 1408201530], length 0
```

Bonus

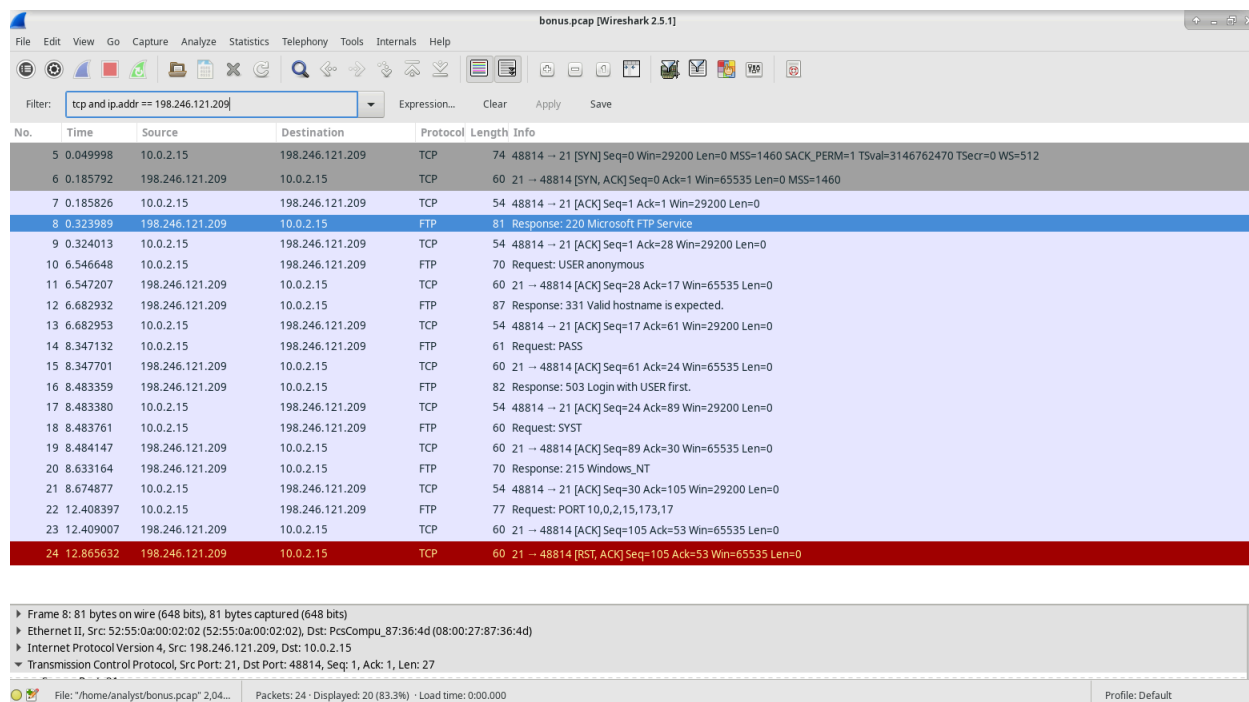
Identificazione header TCP con Wireshark

Avviamo *wireshark* per catturare il traffico di rete sull'interfaccia *enp0s3*. Apriamo una finestra del terminale ed eseguiamo il comando *ftp ftp.cdc.gov* per eseguire una connessione al server *ftp*, autenticandoci con l'utente *anonymous*.

```
[analyst@secOps ~]$ ftp ftp.cdc.gov
Connected to ftp.cdc.gov.
220 Microsoft FTP Service
Name (ftp.cdc.gov:analyst): anonymous
331 Valid hostname is expected.
Password:
503 Login with USER first.
ftp: Login failed.
Remote system type is Windows_NT.
ftp> ls
421 Service not available, remote server has closed connection
ftp: No control connection for command
```

Purtroppo il server è down ma possiamo comunque proseguire il resto dell'esercizio con quanto catturato fin'ora.

Applichiamo il filtro *tcp and ip.addr == 198.246.121.209* alla cattura effettuata per filtrare il traffico tcp da e verso il server ftp.



The screenshot displays the Wireshark 2.5.1 interface. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Tools, Internals, and Help. The toolbar contains icons for file operations, capture control, and analysis. The filter bar at the top shows the active filter: `tcp and ip.addr == 198.246.121.209`. Below the filter bar is a table of captured packets with columns for No., Time, Source, Destination, Protocol, and Length. The packets are filtered to show only those related to the FTP session. The packet details pane at the bottom shows the structure of a selected packet (Frame 8), including Ethernet II, Internet Protocol Version 4, and Transmission Control Protocol fields.

No.	Time	Source	Destination	Protocol	Length	Info
5	0.049998	10.0.2.15	198.246.121.209	TCP	74	48814 → 21 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=3146762470 TSecr=0 WS=512
6	0.185792	198.246.121.209	10.0.2.15	TCP	60	21 → 48814 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
7	0.185826	10.0.2.15	198.246.121.209	TCP	54	48814 → 21 [ACK] Seq=1 Ack=1 Win=29200 Len=0
8	0.323989	198.246.121.209	10.0.2.15	FTP	81	Response: 220 Microsoft FTP Service
9	0.324013	10.0.2.15	198.246.121.209	TCP	54	48814 → 21 [ACK] Seq=1 Ack=28 Win=29200 Len=0
10	6.546648	10.0.2.15	198.246.121.209	FTP	70	Request: USER anonymous
11	6.547207	198.246.121.209	10.0.2.15	TCP	60	21 → 48814 [ACK] Seq=28 Ack=17 Win=65535 Len=0
12	6.682932	198.246.121.209	10.0.2.15	FTP	87	Response: 331 Valid hostname is expected.
13	6.682953	10.0.2.15	198.246.121.209	TCP	54	48814 → 21 [ACK] Seq=17 Ack=61 Win=29200 Len=0
14	8.347132	10.0.2.15	198.246.121.209	FTP	61	Request: PASS
15	8.347701	198.246.121.209	10.0.2.15	TCP	60	21 → 48814 [ACK] Seq=61 Ack=24 Win=65535 Len=0
16	8.483359	198.246.121.209	10.0.2.15	FTP	82	Response: 503 Login with USER first.
17	8.483380	10.0.2.15	198.246.121.209	TCP	54	48814 → 21 [ACK] Seq=24 Ack=89 Win=29200 Len=0
18	8.483761	198.246.121.209	10.0.2.15	FTP	60	Request: SYST
19	8.484147	198.246.121.209	10.0.2.15	TCP	60	21 → 48814 [ACK] Seq=89 Ack=30 Win=65535 Len=0
20	8.633164	198.246.121.209	10.0.2.15	FTP	70	Response: 215 Windows_NT
21	8.674877	10.0.2.15	198.246.121.209	TCP	54	48814 → 21 [ACK] Seq=30 Ack=105 Win=29200 Len=0
22	12.408397	10.0.2.15	198.246.121.209	FTP	77	Request: PORT 10,0,2,15,173,17
23	12.409007	198.246.121.209	10.0.2.15	TCP	60	21 → 48814 [ACK] Seq=105 Ack=53 Win=65535 Len=0
24	12.865632	198.246.121.209	10.0.2.15	TCP	60	21 → 48814 [RST, ACK] Seq=105 Ack=53 Win=65535 Len=0

Frame 8: 81 bytes on wire (648 bits), 81 bytes captured (648 bits)
Ethernet II, Src: 52:55:0a:00:02:02 (52:55:0a:00:02:02), Dst: PcsCompu_87:36:4d (08:00:27:87:36:4d)
Internet Protocol Version 4, Src: 198.246.121.209, Dst: 10.0.2.15
Transmission Control Protocol, Src Port: 21, Dst Port: 48814, Seq: 1, Ack: 1, Len: 27

File: /home/analyst/bonus.pcap* 2,04... Packets: 24 - Displayed: 20 (83.3%) - Load time: 0:00:00.000 Profile: Default

Lo screenshot mostra il risultato della cattura del traffico, con i filtri applicati.

I primi 3 pacchetti catturati, come nel caso precedente, sono relativi all'handshake a tre vie del protocollo TCP.

Tramite il pannello in basso è possibile analizzare tutte le informazioni contenute nell'header dei pacchetti relativi all'handshake.

Le tabelle che seguono mostrano le informazioni dell'header TCP relativo al primo ed al secondo pacchetto catturato

Description	Wireshark results	Description	Wireshark results
Indirizzo IP sorgente	10.0.2.15	Indirizzo IP sorgente	198.246.121.209
Indirizzo IP destinazione	198.246.121.209	Indirizzo IP destinazione	10.0.2.15
Numero porta sorgente	48814	Numero porta sorgente	21
Numero porta destinazione	21	Numero porta destinazione	48814
Sequence number	0	Sequence number	0
Acknowledgment number	-	Acknowledgment number	1
Header length	40	Header length	24
Windows size	29200	Windows size	65535

Lo screenshot che segue mostra il traffico ftp generato durante la cattura:

Filter:	ftp and ip.addr == 198.246.121.209	▼	Expression...	Clear	Apply	Save
No.	Time	Source	Destination	Protocol	Length	Info
8	0.323989	198.246.121.209	10.0.2.15	FTP	81	Response: 220 Microsoft FTP Service
10	6.546648	10.0.2.15	198.246.121.209	FTP	70	Request: USER anonymous
12	6.682932	198.246.121.209	10.0.2.15	FTP	87	Response: 331 Valid hostname is expected.
14	8.347132	10.0.2.15	198.246.121.209	FTP	61	Request: PASS
16	8.483359	198.246.121.209	10.0.2.15	FTP	82	Response: 503 Login with USER first.
18	8.483761	10.0.2.15	198.246.121.209	FTP	60	Request: SYST
20	8.633164	198.246.121.209	10.0.2.15	FTP	70	Response: 215 Windows_NT
22	12.408397	10.0.2.15	198.246.121.209	FTP	77	Request: PORT 10.0.2.15,173,17

Al termine del traffico *ftp* in questo caso la sessione viene terminata con un pacchetto *RST, ACK* anziché con un pacchetto *FIN, ACK*.

Identificazione header UDP con Wireshark

Eseguiamo il comando `sudo lab.support.files/scripts/cyberops_topo.py` in una nuova finestra di terminale ed avviamo *H1* e *H2* con `xterm H1 H2`.

In *H1* avviamo il server *tftpd* con `/home/analyst/lab.support.files/scripts/start_tftpd.sh` e creiamo un file di testo con `echo "Il file contiene dati tftp" > /srv/tftp/my_tftp_data`.

Apriamo *wireshark* ed abilitiamo *Validate the UDP checksum if possible* in *Edit > Preferences > Protocols > UDP*. Avviamo la cattura dei pacchetti di *wireshark* sull'interfaccia *eth0* di *H1* ed eseguiamo `tftp 10.0.0.11 -c get my_tftp_data` su *H2* per scaricare il file di testo creato.

Interrompiamo la cattura del traffico e filtriamo i risultati con `tftp`.

Filter: tftp		Expression...		Clear	Apply	Save
No.	Time	Source	Destination	Protocol	Length	Info
3	0.000065	10.0.0.12	10.0.0.11	TFTP	66	Read Request, File: my_tftp_data, Transfer type: netascii
4	0.000474	10.0.0.11	10.0.0.12	TFTP	74	Data Packet, Block: 1 (last)
5	0.000540	10.0.0.12	10.0.0.11	TFTP	46	Acknowledgement, Block: 1

Anche in questo caso possiamo analizzare tutti i campi dell'header tramite il pannello in basso in *wireshark*.

Description	Wireshark results
Indirizzo IP sorgente	10.0.0.12
Indirizzo IP destinazione	10.0.0.11
Numero porta sorgente	47208
Numero porta destinazione	69
Lunghezza messaggio UDP	32
UDP checksum	0x22a5

Description	Wireshark results
Indirizzo IP sorgente	10.0.0.11
Indirizzo IP destinazione	10.0.0.12
Numero porta sorgente	34233
Numero porta destinazione	47208
Lunghezza messaggio UDP	40
UDP checksum	0xea79