

Utilizzo di Wireshark per Esaminare le Catture TCP e UDP

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

Per esaminare TCP, ho avviato una sessione FTP verso un server pubblico. A causa di restrizioni riscontrate nell'accesso anonimo al server ftp.cdc.gov indicato nella guida, ho utilizzato il server alternativo ftp.gnu.org.

A screenshot of a terminal window titled "Terminal - analyst@secOps:-". The terminal shows an FTP session. The user enters 'ftp ftp.gnu.org' and is connected. They enter 'anonymous' as the user name. The terminal displays a directory listing for the current directory, showing files like CRYPTO.README, MISSING-FILES, and README. Finally, the user enters 'get README' and the terminal shows the file being downloaded successfully.

```
analyst@secOps ~]$ ftp ftp.gnu.org
Connected to ftp.gnu.org.
220 GNU FTP server ready.
Name (ftp.gnu.org:analyst): anonymous
230-NOTICE (Updated October 15 2021):
230-
230-If you maintain scripts used to access ftp.gnu.org over FTP,
230-we strongly encourage you to change them to use HTTPS instead.
230-
230-Eventually we hope to shut down FTP protocol access, but plan
230-to give notice here and other places for several months ahead
230-of time.
230-
230-
230-
230-Due to U.S. Export Regulations, all cryptographic software on this
230-site is subject to the following legal notice:
230-
230-   This site includes publicly available encryption source code
230-   which, together with object code resulting from the compiling of
230-   publicly available source code, may be exported from the United
230-   States under License Exception "TSU" pursuant to 15 C.F.R. Section
230-   740.13(e).
230-
230-This legal notice applies to cryptographic software only. Please see
230-the Bureau of Industry and Security (www.bxa.doc.gov) for more
230-information about current U.S. regulations.
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxrwxrwx   1 0      0           8 Aug 20   2004 CRYPTO.README -> .message
-rw-r--r--   1 0      0       17864 Oct 23   2003 MISSING-FILES
-rw-r--r--   1 0      0       4178 Aug 13   2003 MISSING-FILES.README
-rw-r--r--   1 0      0       2748 May 23   2023 README
-rw-r--r--   1 0      0       405121 Oct 23   2003 before-2003-08-01.md5sums.asc
-rw-rw-r--   1 0      3003    254391 Apr 09 11:32 find.txt.gz
drwxrwxr-x  325 0      3003    12288 Mar 13 15:34 gnu
drwxrwxr-x   3 0      3003     4096 Mar 10 2011 gnu+linux-distros
-rw-rw-r--   1 0      3003    490073 Apr 09 11:32 ls-lrRt.txt.gz
drwxr-xr-x   3 0      0         4096 Apr 20 2005 mirrors
lrwxrwxrwx   1 0      0         11 Apr 15 2004 non-gnu -> gnu/non-gnu
drwxr-xr-x  99 0      0         4096 May 08 2023 old-gnu
lrwxrwxrwx   1 0      0         1 Aug 05 2003 pub -> .
drwxr-xr-x   2 0      0         4096 Nov 08 2007 savannah
drwxr-xr-x   2 0      0         4096 Aug 02 2003 third-party
drwxr-xr-x   2 0      0         4096 Apr 07 2009 tmp
-rw-rw-r--   1 0      3003    577095 Apr 09 11:32 tree.json.gz
drwxr-xr-x   2 0      0         4096 May 07 2013 video
-rw-r--r--   1 0      0       1092 Oct 15 2021 welcome.msg
226 Directory send OK.
ftp> get README
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for README (2748 bytes).
226 Transfer complete.
2748 bytes received in 0.000487 seconds (5.38 Mbytes/s)
ftp>
```

Ho avviato Wireshark per catturare il traffico sull'interfaccia di rete principale della VM. Successivamente, ho utilizzato il client ftp da terminale per connettermi al server, autenticarmi come utente anonymous e eseguire i comandi ls e get README. Al

termine delle operazioni, ho interrotto la cattura Wireshark e applicato filtri per isolare i pacchetti rilevanti.

Filter: tcp and ip.addr == 209.51.188.20

No.	Time	Source	Destination	Protocol	Length	Info
12	6.736479806	192.168.1.222	209.51.188.20	TCP	74	48968 → 21 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2451362440 TSecr=0 WS=128
13	6.860292065	209.51.188.20	192.168.1.222	TCP	74	21 → 48968 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1452 SACK_PERM=1 TSval=2070155859 TSecr=2451362440 WS=128
14	6.860315375	192.168.1.222	209.51.188.20	TCP	66	48968 → 21 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=2451362564 TSecr=2070155859
16	6.998482557	209.51.188.20	192.168.1.222	FTP	93	Response: 220 GNU FTP server ready.
17	6.998503767	192.168.1.222	209.51.188.20	TCP	66	48968 → 21 [ACK] Seq=1 Ack=28 Win=29312 Len=0 TSval=2451362702 TSecr=2070155994
21	12.59013383	192.168.1.222	209.51.188.20	FTP	82	Request: USER anonymous
22	12.71370484	209.51.188.20	192.168.1.222	TCP	66	21 → 48968 [ACK] Seq=28 Ack=17 Win=65280 Len=0 TSval=2070161715 TSecr=2451368293
23	12.71912365	209.51.188.20	192.168.1.222	FTP	105	Response: 230-NOTICE (Updated October 15 2021):
24	12.71913474	192.168.1.222	209.51.188.20	TCP	66	48968 → 21 [ACK] Seq=17 Ack=67 Win=29312 Len=0 TSval=2451368422 TSecr=2070161720
25	12.71918758	209.51.188.20	192.168.1.222	FTP	138	Response: 230-
26	12.71919245	192.168.1.222	209.51.188.20	TCP	66	48968 → 21 [ACK] Seq=17 Ack=139 Win=29312 Len=0 TSval=2451368422 TSecr=2070161720
27	12.71923837	209.51.188.20	192.168.1.222	FTP	433	Response: 230-we strongly encourage you to change them to use HTTPS instead.
28	12.71924335	192.168.1.222	209.51.188.20	TCP	66	48968 → 21 [ACK] Seq=17 Ack=506 Win=30336 Len=0 TSval=2451368422 TSecr=2070161720
29	12.71928864	209.51.188.20	192.168.1.222	FTP	465	Response: 230-
30	12.71929453	192.168.1.222	209.51.188.20	TCP	66	48968 → 21 [ACK] Seq=17 Ack=905 Win=31360 Len=0 TSval=2451368423 TSecr=2070161720
31	12.71933891	209.51.188.20	192.168.1.222	FTP	206	Response: 230-the Bureau of Industry and Security (www.bxa.doc.gov) for more
32	12.71934380	192.168.1.222	209.51.188.20	TCP	66	48968 → 21 [ACK] Seq=17 Ack=1045 Win=32512 Len=0 TSval=2451368423 TSecr=2070161720
33	12.71969176	192.168.1.222	209.51.188.20	FTP	72	Request: SYST
34	12.84440485	209.51.188.20	192.168.1.222	FTP	85	Response: 215 UNIX Type: L8
35	12.88962324	209.51.188.20	209.51.188.20	TCP	66	48968 → 21 [ACK] Seq=23 Ack=1064 Win=32512 Len=0 TSval=2451368593 TSecr=2070161842
43	19.37406630	192.168.1.222	209.51.188.20	FTP	94	Request: PORT 192,168,1,222,147,145

Frame 12: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0

Ethernet II, Src: PcsCompu_97:04:22 (08:00:27:97:04:22), Dst: d4:35:1d:d2:01:43 (d4:35:1d:d2:01:43)

Internet Protocol Version 4, Src: 192.168.1.222, Dst: 209.51.188.20

Transmission Control Protocol, Src Port: 48968, Dst Port: 21, Seq: 0, Len: 0

Source Port: 48968
Destination Port: 21
[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: 0x4ff0 [unverified]
[Checksum Status: Unverified]

L'analisi dei segmenti TCP catturati ha permesso le seguenti osservazioni:

Instaurazione della Connessione: Sebbene non fosse l'obiettivo primario, la cattura ha mostrato l'handshake a tre vie (SYN, SYN-ACK, ACK) per stabilire la connessione di controllo sulla porta TCP 21, confermando la natura orientata alla connessione di TCP. Connessioni TCP separate sono state stabilite per i trasferimenti dati (es. ls, get).

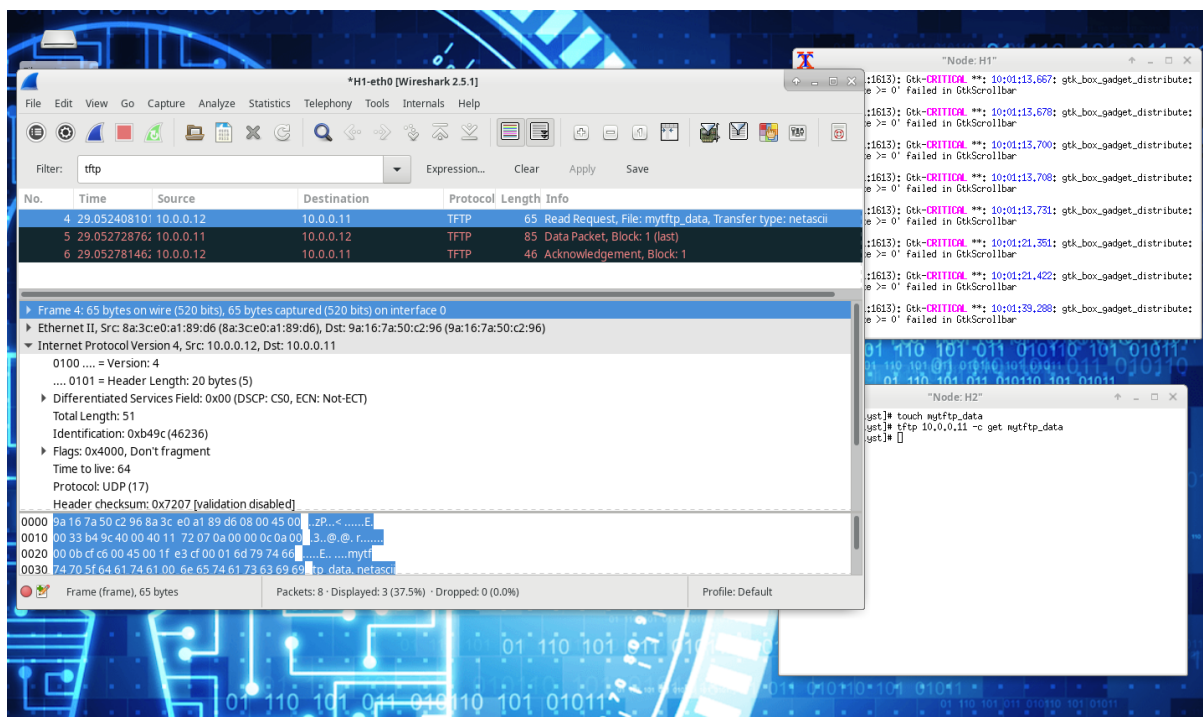
Affidabilità e Ordinamento: La progressione dei valori nei campi Sequence Number e Acknowledgment Number è stata chiaramente osservata. Questi meccanismi sono risultati fondamentali per garantire la consegna ordinata e affidabile dei dati relativi ai comandi FTP, alle risposte del server e al contenuto del file trasferito. Ogni segmento di dati trasmesso richiedeva un corrispondente ACK.

Gestione della Connessione e Flusso: I flag TCP sono stati ispezionati: SYN e ACK per la gestione della connessione; ACK per la conferma dei dati ricevuti; PSH per indicare la necessità di consegnare immediatamente i dati all'applicazione; FIN per la chiusura ordinata delle connessioni. Il campo Window Size mostrava valori dinamici, indicativi del meccanismo di controllo di flusso implementato da TCP.

Struttura dell'Intestazione: È stata verificata la presenza e la funzione dei campi standard dell'header TCP (Porte Sorgente/Destinazione, Numeri di Sequenza/Ack, Flag, Window Size, Checksum, Urgent Pointer - sebbene non utilizzato in questo caso).

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

Per l'analisi di UDP, ho configurato un ambiente TFTP utilizzando Mininet. Ho avviato un server tftpd sull'host H1 e ho utilizzato l'host H2 come client. Ho creato un file di test (my_tftp_data) su H1. Ho avviato Wireshark su H1 per catturare il traffico sull'interfaccia H1-eth0. Da H2, ho eseguito il comando `tftp 10.0.0.11 -c get my_tftp_data` per scaricare il file. Completato il trasferimento, ho fermato la cattura e applicato filtri (tftp) in Wireshark.



L'esame dei datagrammi UDP ha rivelato un comportamento distinto rispetto a TCP:

Natura Connectionless: Non è stato osservato alcun handshake preliminare. La comunicazione è iniziata direttamente con l'invio del pacchetto TFTP RRQ (Read Request) da H2 alla porta UDP 69 di H1.

Mancanza di Affidabilità Intrinseca: UDP non implementa meccanismi intrinseci per garantire la consegna, l'ordine o la duplicazione dei pacchetti.

Affidabilità a Livello Applicativo: La necessaria affidabilità per il trasferimento del file è stata gestita interamente dal protocollo TFTP stesso. Questo è stato evidente dall'osservazione dello scambio di pacchetti DATA (contenenti blocchi di file numerati progressivamente) e pacchetti ACK specifici di TFTP, utilizzati per confermare la ricezione di ciascun blocco. La gestione di ritrasmissioni in caso di perdita di pacchetti DATA o ACK sarebbe a carico di TFTP, non di UDP.