Relazione Progetto – Reti di Telecomunicazione

Traccia 3 - Wireshark

Giacomo Pierbattista – matricola 0900069485

giacomo.pierbattista@studio.unibo.it

Introduzione

Questo report è lo svolgimento della traccia 3, la quale consiste in catturare e analizzare il traffico durante il trasferimento di file su HTTP per comprendere il funzionamento del protocollo TCP/IP.

Il repository del progetto si trova all'indirizzo

https://github.com/jackprb/http-analysis-wireshark

Contesto

Sullo stesso computer Windows avente indirizzo IP locale 192.168.1.2 con subnet mask 255.255.255.0 sono stati attivati:

- un server HTTP locale (XAMPP) raggiungibile all'indirizzo IP 192.168.1.2 e in esecuzione come processo in ascolto sulla porta 80 di default di HTTP

```
Scheda LAN wireless Wi-Fi:

   Suffisso DNS specifico per connessione: home
   Indirizzo IPv6 locale rispetto al collegamento . : fe80::d3f1:3f32:f4c0:bc56%5
   Indirizzo IPv4. . . . . . . . . . . . . : 192.168.1.2
   Subnet mask . . . . . . . . . . . . . . : 255.255.255.0
   Gateway predefinito . . . . . . . . . . : 192.168.1.1
```

- una macchina virtuale VirtualBox con Lubuntu 20.04 in esecuzione con indirizzo IP locale 10.0.2.15 e subnet mask 255.255.255.0

```
jack@jack-virtualbox:~/Scrivania$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::2ccd:50a4:5f5c:7b86  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:86:86:b7  txqueuelen 1000  (Ethernet)
        RX packets 1170  bytes 975089 (975.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 697  bytes 126381 (126.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

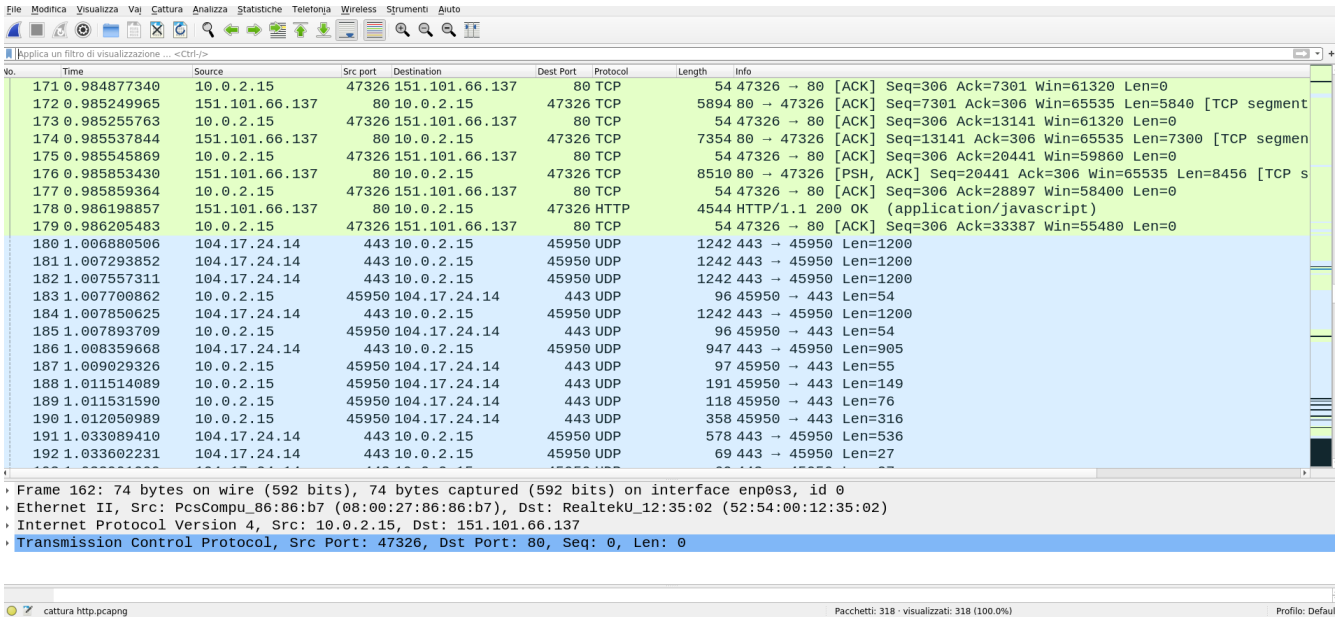
La comunicazione tra macchina virtuale e macchina reale è stata possibile attraverso il NAT di default di VirtualBox, il quale inoltra il traffico della macchina virtuale attraverso la scheda di rete fisica (in questo caso, la scheda di rete Wi-Fi).

Analisi

È stato analizzato il trasferimento via HTTP della pagina “dashboard” del server XAMPP, visibile in Figura 1 e raggiungibile all’indirizzo http://192.168.1.2/dashboard. Essa è una semplice pagina HTML che incorpora alcune immagini, fogli di stile CSS e script JavaScript.

Figura 1

Come mostrato in Figura 2, sono stati catturati anche pacchetti di altri protocolli (tra cui pacchetti UDP) e provenienti o diretti verso altri indirizzi IP; attraverso il filtro di Wireshark `tcp.stream == n`, con $n \geq 0$, si riescono a distinguere i vari flussi TCP (in questa analisi, i flussi sono numerati da 0 a 7 e solo i flussi 1, 3 e 4 riguardano il trasferimento HTTP della pagina web e degli altri file in essa incorporati o linkati).



No.	Time	Source	Src port	Destination	Dest Port	Protocol	Length	Info
171	0.984877340	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=7301 Win=61320 Len=0
172	0.985249965	151.101.66.137	80	10.0.2.15	47326	TCP	5894	80 → 47326 [ACK] Seq=7301 Ack=306 Win=65535 Len=5840 [TCP segment
173	0.985255763	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=13141 Win=61320 Len=0
174	0.985537844	151.101.66.137	80	10.0.2.15	47326	TCP	7354	80 → 47326 [ACK] Seq=13141 Ack=306 Win=65535 Len=7300 [TCP segmen
175	0.985545869	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=20441 Win=59860 Len=0
176	0.985853430	151.101.66.137	80	10.0.2.15	47326	TCP	8510	80 → 47326 [PSH, ACK] Seq=20441 Ack=306 Win=65535 Len=8456 [TCP s
177	0.985859364	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=28897 Win=58400 Len=0
178	0.986198857	151.101.66.137	80	10.0.2.15	47326	HTTP	4544	HTTP/1.1 200 OK (application/javascript)
179	0.986205483	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=33387 Win=55480 Len=0
180	1.006880506	104.17.24.14	443	10.0.2.15	45950	UDP	1242	443 → 45950 Len=1200
181	1.007293852	104.17.24.14	443	10.0.2.15	45950	UDP	1242	443 → 45950 Len=1200
182	1.007557311	104.17.24.14	443	10.0.2.15	45950	UDP	1242	443 → 45950 Len=1200
183	1.007700862	10.0.2.15	45950	104.17.24.14	443	UDP	96	45950 → 443 Len=54
184	1.007850625	104.17.24.14	443	10.0.2.15	45950	UDP	1242	443 → 45950 Len=1200
185	1.007893709	10.0.2.15	45950	104.17.24.14	443	UDP	96	45950 → 443 Len=54
186	1.008359668	104.17.24.14	443	10.0.2.15	45950	UDP	947	443 → 45950 Len=905
187	1.009029326	10.0.2.15	45950	104.17.24.14	443	UDP	97	45950 → 443 Len=55
188	1.011514089	10.0.2.15	45950	104.17.24.14	443	UDP	191	45950 → 443 Len=149
189	1.011531590	10.0.2.15	45950	104.17.24.14	443	UDP	118	45950 → 443 Len=76
190	1.012050989	10.0.2.15	45950	104.17.24.14	443	UDP	358	45950 → 443 Len=316
191	1.033089410	104.17.24.14	443	10.0.2.15	45950	UDP	578	443 → 45950 Len=536
192	1.033602231	104.17.24.14	443	10.0.2.15	45950	UDP	69	443 → 45950 Len=27

Figura 2

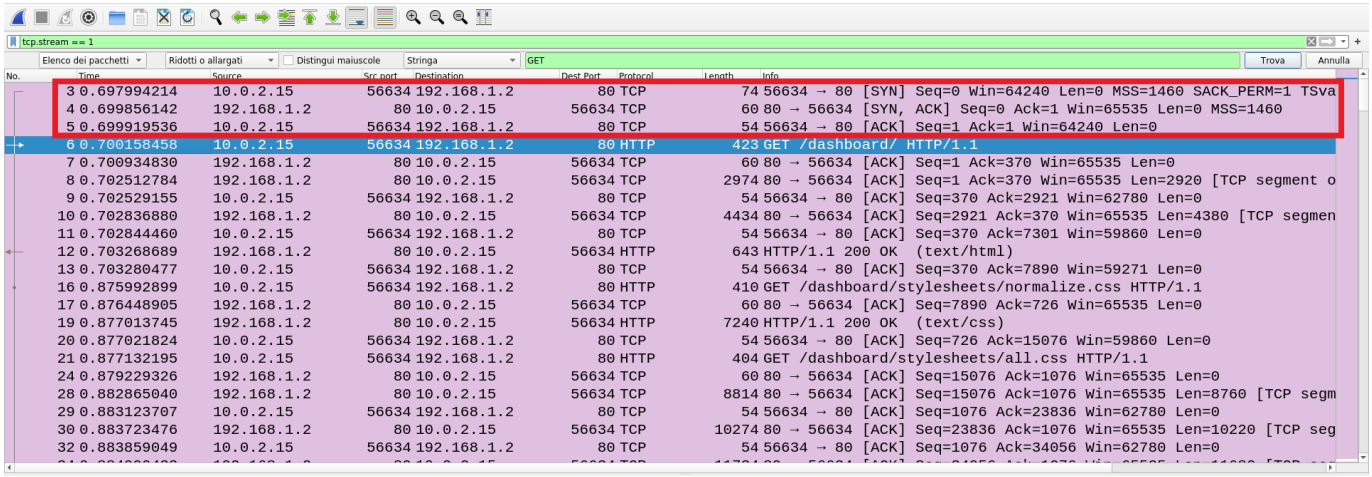
Identificazione della sequenza di handshake

Il protocollo TCP prevede l'invio di una sequenza di messaggi detta "three-way handshake" per stabilire la connessione tra client e server, attraverso l'invio di:

1. pacchetto SYN con flag SYN=1, ACK=0
2. pacchetto SYN-ACK con flag SYN=1, ACK=1
3. pacchetto ACK con flag SYN=0, ACK=1

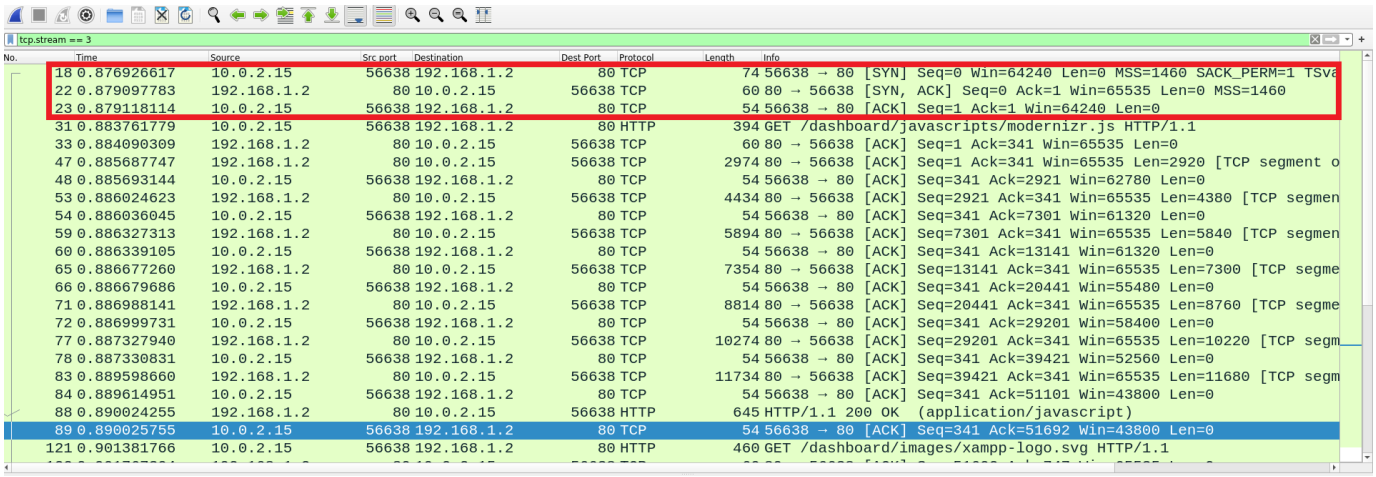
TCP permette la creazione di più connessioni parallele; di conseguenza, ognuna di esse viene aperta con un handshake.

Nelle Figure 3, 4 e 5 si possono notare gli handshake rispettivamente dei flussi `tcp.stream == 1`, `tcp.stream == 3` e `tcp.stream == 4`.



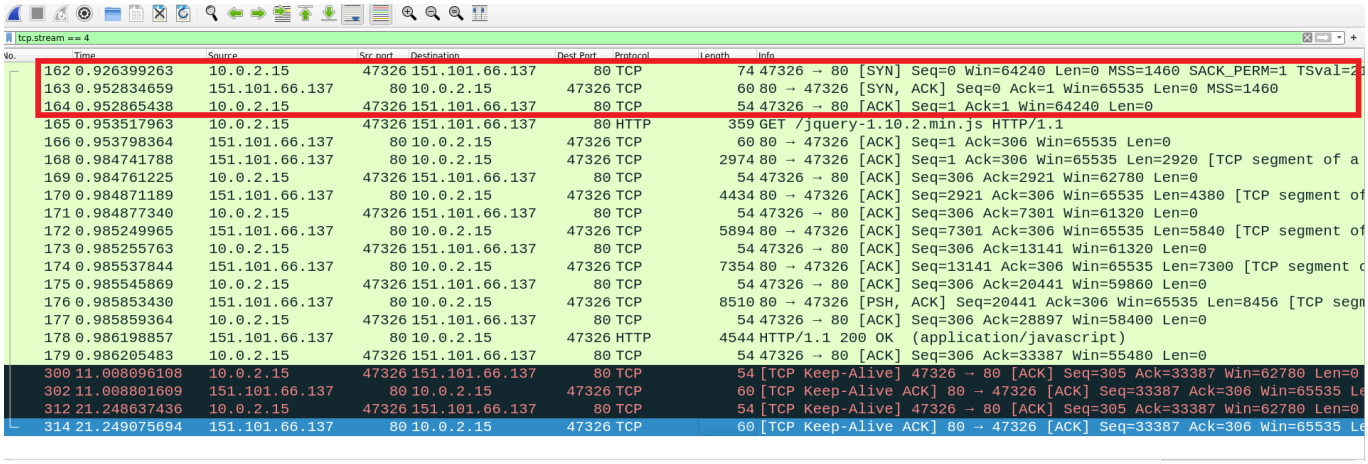
No.	Time	Source	Src port	Destination	Dest Port	Protocol	Length	Info
3	0.697994214	10.0.2.15	56634	192.168.1.2	80	TCP	74	56634 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSva
4	0.699856142	192.168.1.2	80	10.0.2.15	56634	TCP	60	80 → 56634 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
5	0.699919536	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
6	0.700158458	10.0.2.15	56634	192.168.1.2	80	HTTP	423	GET /dashboard/ HTTP/1.1
7	0.700934830	192.168.1.2	80	10.0.2.15	56634	TCP	60	80 → 56634 [ACK] Seq=1 Ack=370 Win=65535 Len=0
8	0.702512784	192.168.1.2	80	10.0.2.15	56634	TCP	2974	80 → 56634 [ACK] Seq=1 Ack=370 Win=65535 Len=2920 [TCP segment o
9	0.702529155	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=370 Ack=2921 Win=62780 Len=0
10	0.702836880	192.168.1.2	80	10.0.2.15	56634	TCP	4434	80 → 56634 [ACK] Seq=2921 Ack=370 Win=65535 Len=4380 [TCP segmen
11	0.702844460	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=370 Ack=7301 Win=59860 Len=0
12	0.703268689	192.168.1.2	80	10.0.2.15	56634	HTTP	643	HTTP/1.1 200 OK (text/html)
13	0.703280477	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=370 Ack=7890 Win=59271 Len=0
16	0.875992899	10.0.2.15	56634	192.168.1.2	80	HTTP	410	GET /dashboard/stylesheets/normalize.css HTTP/1.1
17	0.876448905	192.168.1.2	80	10.0.2.15	56634	TCP	60	80 → 56634 [ACK] Seq=7890 Ack=726 Win=65535 Len=0
19	0.877013745	192.168.1.2	80	10.0.2.15	56634	HTTP	7240	HTTP/1.1 200 OK (text/css)
20	0.877021824	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=726 Ack=15076 Win=59860 Len=0
21	0.877132195	10.0.2.15	56634	192.168.1.2	80	HTTP	404	GET /dashboard/stylesheets/all.css HTTP/1.1
24	0.879229326	192.168.1.2	80	10.0.2.15	56634	TCP	60	80 → 56634 [ACK] Seq=15076 Ack=1076 Win=65535 Len=0
28	0.882865040	192.168.1.2	80	10.0.2.15	56634	TCP	8814	80 → 56634 [ACK] Seq=15076 Ack=1076 Win=65535 Len=8760 [TCP segm
29	0.883123707	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=1076 Ack=23836 Win=62780 Len=0
30	0.883723476	192.168.1.2	80	10.0.2.15	56634	TCP	10274	80 → 56634 [ACK] Seq=23836 Ack=1076 Win=65535 Len=10220 [TCP seg
32	0.883859049	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=1076 Ack=34056 Win=62780 Len=0

Figura 3



No.	Time	Source	Src port	Destination	Dest Port	Protocol	Length	Info
18	0.876926617	10.0.2.15	56638	192.168.1.2	80	TCP	74	56638 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSva
22	0.879097783	192.168.1.2	80	10.0.2.15	56638	TCP	60	80 → 56638 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
23	0.879118114	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
31	0.883761779	10.0.2.15	56638	192.168.1.2	80	HTTP	394	GET /dashboard/javascripts/modernizr.js HTTP/1.1
33	0.884090309	192.168.1.2	80	10.0.2.15	56638	TCP	60	80 → 56638 [ACK] Seq=1 Ack=341 Win=65535 Len=0
47	0.885687747	192.168.1.2	80	10.0.2.15	56638	TCP	2974	80 → 56638 [ACK] Seq=1 Ack=341 Win=65535 Len=2920 [TCP segment o
48	0.885693144	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=2921 Win=62780 Len=0
53	0.886024623	192.168.1.2	80	10.0.2.15	56638	TCP	4434	80 → 56638 [ACK] Seq=2921 Ack=341 Win=65535 Len=4380 [TCP segmen
54	0.886036045	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=7301 Win=61320 Len=0
59	0.886327313	192.168.1.2	80	10.0.2.15	56638	TCP	5894	80 → 56638 [ACK] Seq=7301 Ack=341 Win=65535 Len=5840 [TCP segmen
60	0.886339105	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=13141 Win=61320 Len=0
65	0.886677260	192.168.1.2	80	10.0.2.15	56638	TCP	7354	80 → 56638 [ACK] Seq=13141 Ack=341 Win=65535 Len=7300 [TCP segme
66	0.886679686	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=20441 Win=55480 Len=0
71	0.886988141	192.168.1.2	80	10.0.2.15	56638	TCP	8814	80 → 56638 [ACK] Seq=20441 Ack=341 Win=65535 Len=8760 [TCP segme
72	0.886999731	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=29201 Win=58400 Len=0
77	0.887327940	192.168.1.2	80	10.0.2.15	56638	TCP	10274	80 → 56638 [ACK] Seq=29201 Ack=341 Win=65535 Len=10220 [TCP segm
78	0.887330831	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=39421 Win=52560 Len=0
83	0.889598660	192.168.1.2	80	10.0.2.15	56638	TCP	11734	80 → 56638 [ACK] Seq=39421 Ack=341 Win=65535 Len=11680 [TCP segm
84	0.889614951	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=51101 Win=43800 Len=0
88	0.890024255	192.168.1.2	80	10.0.2.15	56638	HTTP	645	HTTP/1.1 200 OK (application/javascript)
89	0.890025755	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=51692 Win=43800 Len=0
121	0.901381766	10.0.2.15	56638	192.168.1.2	80	HTTP	460	GET /dashboard/images/xampp-logo.svg HTTP/1.1

Figura 4



No.	Time	Source	Src port	Destination	Dest Port	Protocol	Length	Info
162	0.926399263	10.0.2.15	47326	151.101.66.137	80	TCP	74	47326 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2
163	0.952834659	151.101.66.137	80	10.0.2.15	47326	TCP	60	80 → 47326 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
164	0.952865438	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
165	0.953517963	10.0.2.15	47326	151.101.66.137	80	HTTP	359	GET /jquery-1.10.2.min.js HTTP/1.1
166	0.953798364	151.101.66.137	80	10.0.2.15	47326	TCP	60	80 → 47326 [ACK] Seq=1 Ack=306 Win=65535 Len=0
168	0.984741788	151.101.66.137	80	10.0.2.15	47326	TCP	2974	80 → 47326 [ACK] Seq=1 Ack=306 Win=65535 Len=2920 [TCP segment of a
169	0.984761225	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=2921 Win=62780 Len=0
170	0.984871189	151.101.66.137	80	10.0.2.15	47326	TCP	4434	80 → 47326 [ACK] Seq=2921 Ack=306 Win=65535 Len=4380 [TCP segment of
171	0.984877340	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=7301 Win=61320 Len=0
172	0.985249965	151.101.66.137	80	10.0.2.15	47326	TCP	5894	80 → 47326 [ACK] Seq=7301 Ack=306 Win=65535 Len=5840 [TCP segment of
173	0.985255763	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=13141 Win=61320 Len=0
174	0.985537844	151.101.66.137	80	10.0.2.15	47326	TCP	7354	80 → 47326 [ACK] Seq=13141 Ack=306 Win=65535 Len=7300 [TCP segment c
175	0.985545869	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=20441 Win=59860 Len=0
176	0.985853430	151.101.66.137	80	10.0.2.15	47326	TCP	8510	80 → 47326 [PSH, ACK] Seq=20441 Ack=306 Win=65535 Len=8456 [TCP segm
177	0.985859364	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=28897 Win=58400 Len=0
178	0.986198857	151.101.66.137	80	10.0.2.15	47326	HTTP	4544	HTTP/1.1 200 OK (application/javascript)
179	0.986205483	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=33387 Win=55480 Len=0
300	11.008096108	10.0.2.15	47326	151.101.66.137	80	TCP	54	[TCP Keep-Alive] 47326 → 80 [ACK] Seq=305 Ack=33387 Win=62780 Len=0
302	11.008801609	151.101.66.137	80	10.0.2.15	47326	TCP	60	[TCP Keep-Alive ACK] 80 → 47326 [ACK] Seq=33387 Ack=306 Win=65535 Le
312	21.248637436	10.0.2.15	47326	151.101.66.137	80	TCP	54	[TCP Keep-Alive] 47326 → 80 [ACK] Seq=305 Ack=33387 Win=62780 Len=0
314	21.249075694	151.101.66.137	80	10.0.2.15	47326	TCP	60	[TCP Keep-Alive ACK] 80 → 47326 [ACK] Seq=33387 Ack=306 Win=65535 Le

Figura 5

TCP è un protocollo progettato per rendere affidabili le connessioni: ogni pacchetto inviato ha un numero di sequenza e un numero di acknowledgement, i quali, insieme, permettono di rilevare se un pacchetto è stato ricevuto con successo o meno.

In Figura 6 si possono notare i pacchetti scambiati nella fase di handshake avvenuta nel flusso TCP 4, mentre in Figura 7 lo schema corrispondente. Da tali figure si nota che il numero di sequenza iniziale è un numero casuale, il numero di ACK è zero nel pacchetto SYN, mentre dal pacchetto SYN-ACK in poi il numero di ACK è dato dal:

- numero di sequenza del pacchetto a cui l'ACK si riferisce più uno, in caso di pacchetti con segmento TCP di zero byte,
- numero di sequenza del pacchetto a cui l'ACK si riferisce più la dimensione del segmento TCP negli altri casi.

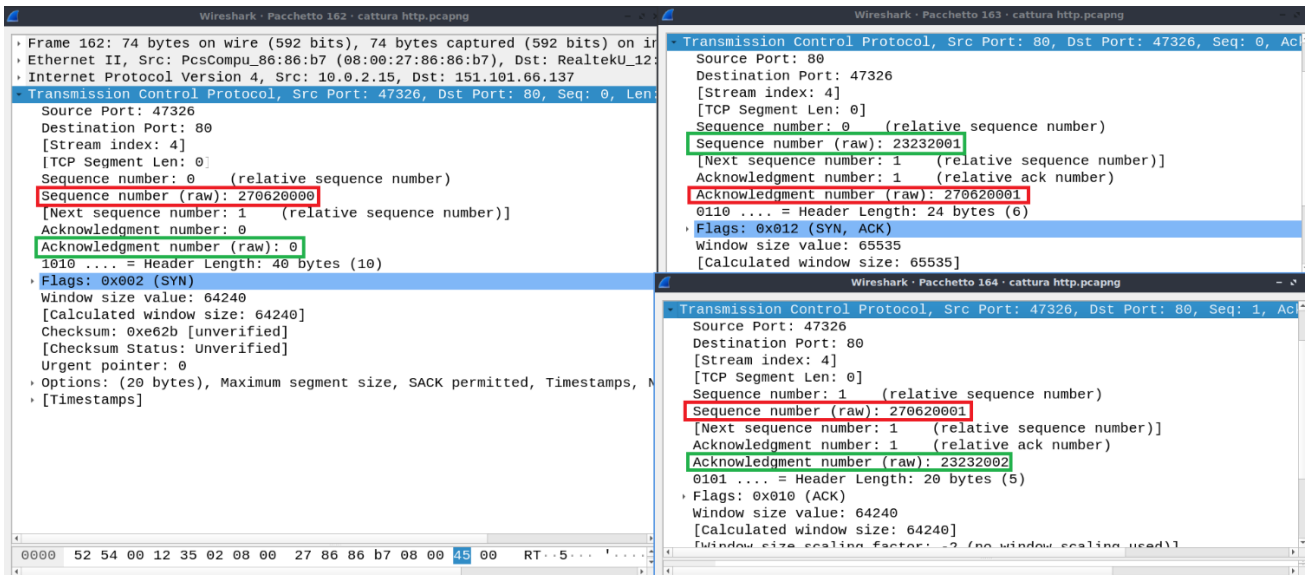


Figura 6

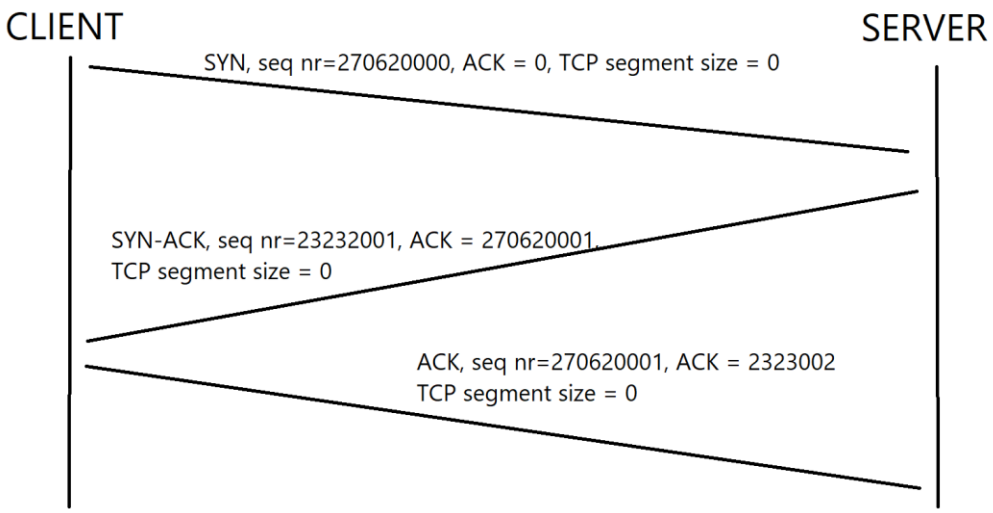
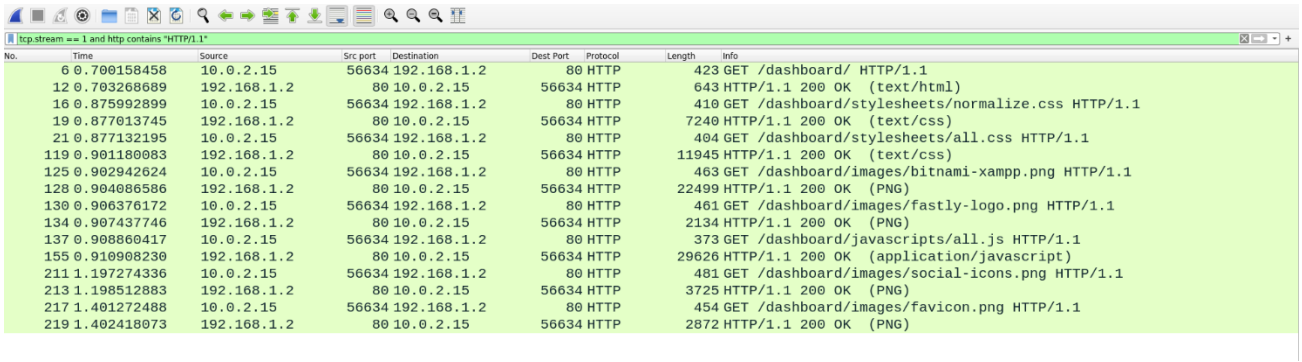


Figura 7

Frammentazione dei pacchetti

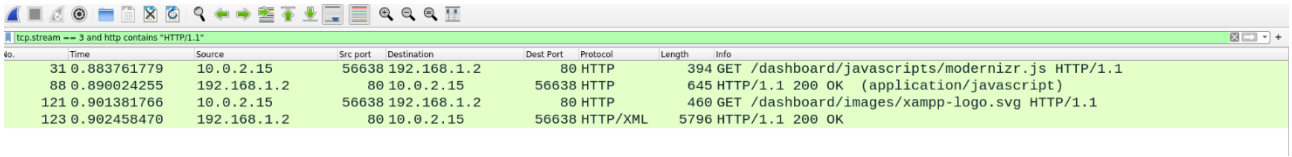
Nelle Figure 8, 9 e 10 si possono notare le richieste HTTP GET con le relative risposte, rispettivamente nei flussi TCP 1, 3 e 4, con le quali sono state scaricate le risorse richieste (la pagina web principale, due fogli di stile CSS, quattro immagini PNG, un file SVG e tre script JavaScript).



tcp.stream == 1 and http contains "HTTP/1.1"

No.	Time	Source	Src port	Destination	Dest Port	Protocol	Length	Info
6	0.700158458	10.0.2.15	56634	192.168.1.2	80	HTTP	423	GET /dashboard/ HTTP/1.1
12	0.703268689	192.168.1.2	80	10.0.2.15	56634	HTTP	643	HTTP/1.1 200 OK (text/html)
16	0.875992899	10.0.2.15	56634	192.168.1.2	80	HTTP	410	GET /dashboard/stylesheets/normalize.css HTTP/1.1
19	0.877013745	192.168.1.2	80	10.0.2.15	56634	HTTP	7240	HTTP/1.1 200 OK (text/css)
21	0.877132195	10.0.2.15	56634	192.168.1.2	80	HTTP	404	GET /dashboard/stylesheets/all.css HTTP/1.1
119	0.901180083	192.168.1.2	80	10.0.2.15	56634	HTTP	11945	HTTP/1.1 200 OK (text/css)
125	0.902942624	10.0.2.15	56634	192.168.1.2	80	HTTP	463	GET /dashboard/images/bitnami-xampp.png HTTP/1.1
128	0.904086586	192.168.1.2	80	10.0.2.15	56634	HTTP	22499	HTTP/1.1 200 OK (PNG)
130	0.906376172	10.0.2.15	56634	192.168.1.2	80	HTTP	461	GET /dashboard/images/fastly-logo.png HTTP/1.1
134	0.907437746	192.168.1.2	80	10.0.2.15	56634	HTTP	2134	HTTP/1.1 200 OK (PNG)
137	0.908860417	10.0.2.15	56634	192.168.1.2	80	HTTP	373	GET /dashboard/javascripts/all.js HTTP/1.1
155	0.910908230	192.168.1.2	80	10.0.2.15	56634	HTTP	29626	HTTP/1.1 200 OK (application/javascript)
211	1.197274336	10.0.2.15	56634	192.168.1.2	80	HTTP	481	GET /dashboard/images/social-icons.png HTTP/1.1
213	1.198512883	192.168.1.2	80	10.0.2.15	56634	HTTP	3725	HTTP/1.1 200 OK (PNG)
217	1.401272488	10.0.2.15	56634	192.168.1.2	80	HTTP	454	GET /dashboard/images/favicon.png HTTP/1.1
219	1.402418073	192.168.1.2	80	10.0.2.15	56634	HTTP	2872	HTTP/1.1 200 OK (PNG)

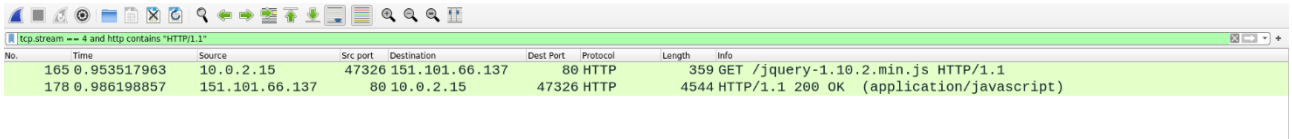
Figura 8



tcp.stream == 3 and http contains "HTTP/1.1"

No.	Time	Source	Src port	Destination	Dest Port	Protocol	Length	Info
31	0.883761779	10.0.2.15	56638	192.168.1.2	80	HTTP	394	GET /dashboard/javascripts/modernizr.js HTTP/1.1
88	0.890024255	192.168.1.2	80	10.0.2.15	56638	HTTP	645	HTTP/1.1 200 OK (application/javascript)
121	0.901381766	10.0.2.15	56638	192.168.1.2	80	HTTP	460	GET /dashboard/images/xampp-logo.svg HTTP/1.1
123	0.902458470	192.168.1.2	80	10.0.2.15	56638	HTTP/XML	5796	HTTP/1.1 200 OK

Figura 9



tcp.stream == 4 and http contains "HTTP/1.1"

No.	Time	Source	Src port	Destination	Dest Port	Protocol	Length	Info
165	0.953517963	10.0.2.15	47326	151.101.66.137	80	HTTP	359	GET /jquery-1.10.2.min.js HTTP/1.1
178	0.986198857	151.101.66.137	80	10.0.2.15	47326	HTTP	4544	HTTP/1.1 200 OK (application/javascript)

Figura 10

In realtà, i file richiesti non sono stati tutti inviati come un unico pacchetto: quelli la cui dimensione supera il massimo consentito sono stati frammentati in più parti, ma ciò non è visibile nelle Figure 8, 9 e 10 a causa dei filtri applicati per mettere in evidenza le richieste HTTP GET e le rispettive risposte.
Ad esempio, in Figura 10, si nota che nel flusso TCP 4 è stata effettuata la richiesta HTTP GET per ottenere la libreria jQuery (pacchetto numero 165) e la risposta corrispondente viene mostrata come un unico pacchetto avente numero 178.

Ispezionando il pacchetto 178, come mostrato in Figura 11, si nota che esso in realtà è l'insieme di 6 segmenti TCP riassemblati (pacchetti 168, 170, 172, 174, 176 e 178) e nei flag TCP del pacchetto 178 sono attivi i bit di PSH e ACK, i quali indicano di inviare il pacchetto riassemblato immediatamente all'applicazione destinataria.

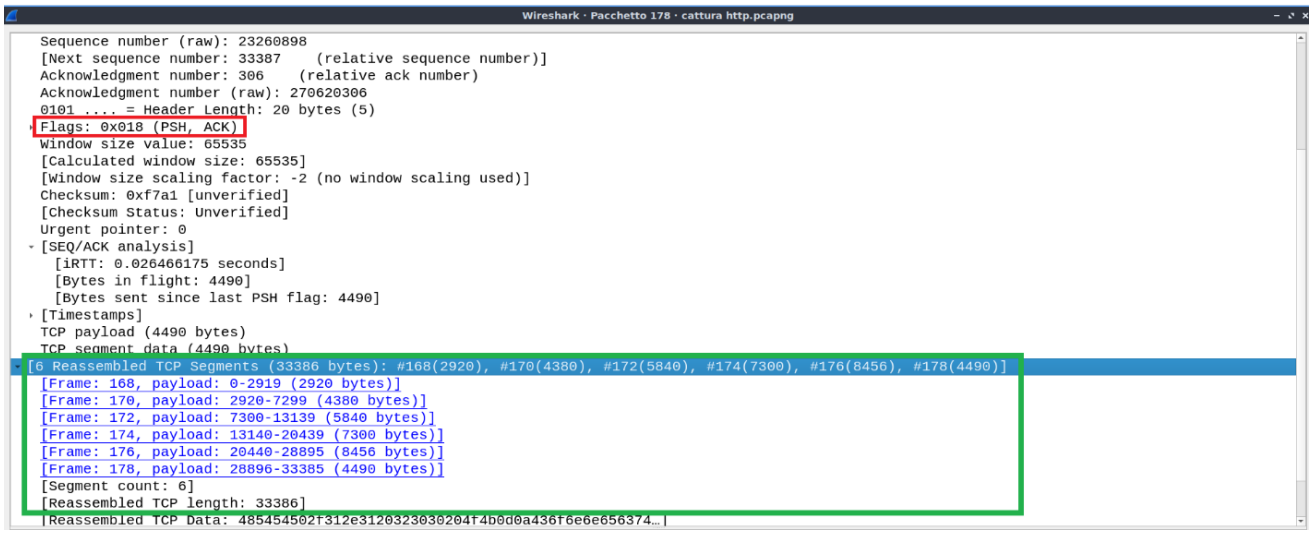


Figura 11

Controllo di flusso

Nelle Figure 12 – 18 sono mostrati i dettagli dei pacchetti scambiati tra server e client per l'invio dei frammenti della libreria jQuery a seguito della richiesta HTTP GET nel pacchetto 165.

In particolare, in Figura 12, si può osservare a sinistra il client che invia la richiesta HTTP GET per ottenere la libreria jQuery, mentre a destra il server che risponde con un ACK.
Invece, nelle Figure 13 – 18, a sinistra si trovano i pacchetti inviati dal server contenenti i frammenti della libreria e a destra gli ACK del client.

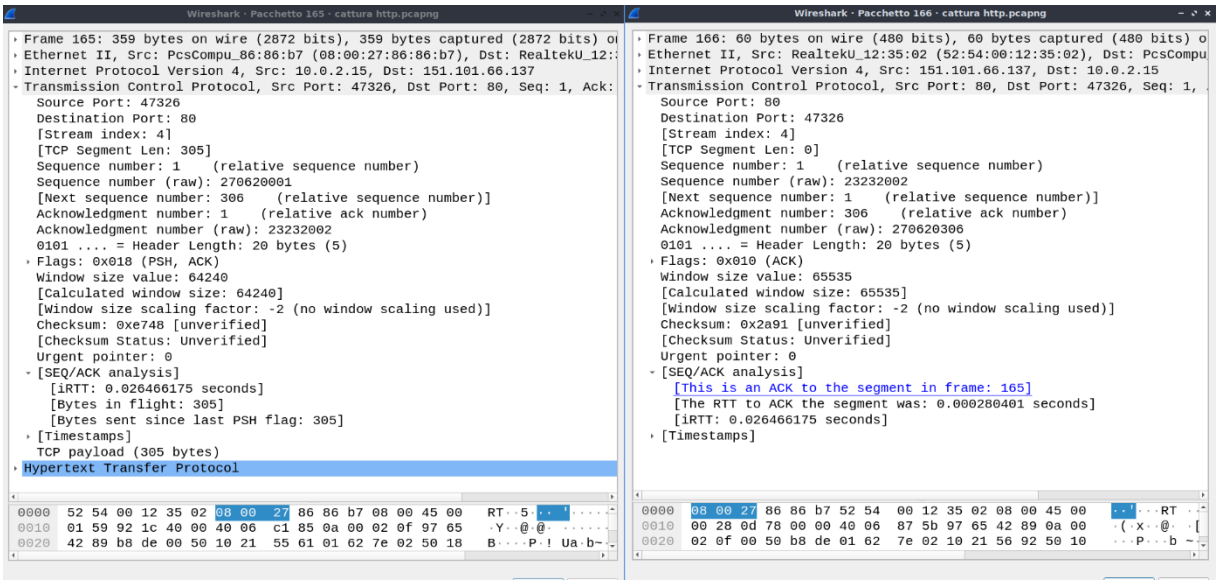


Figura 12

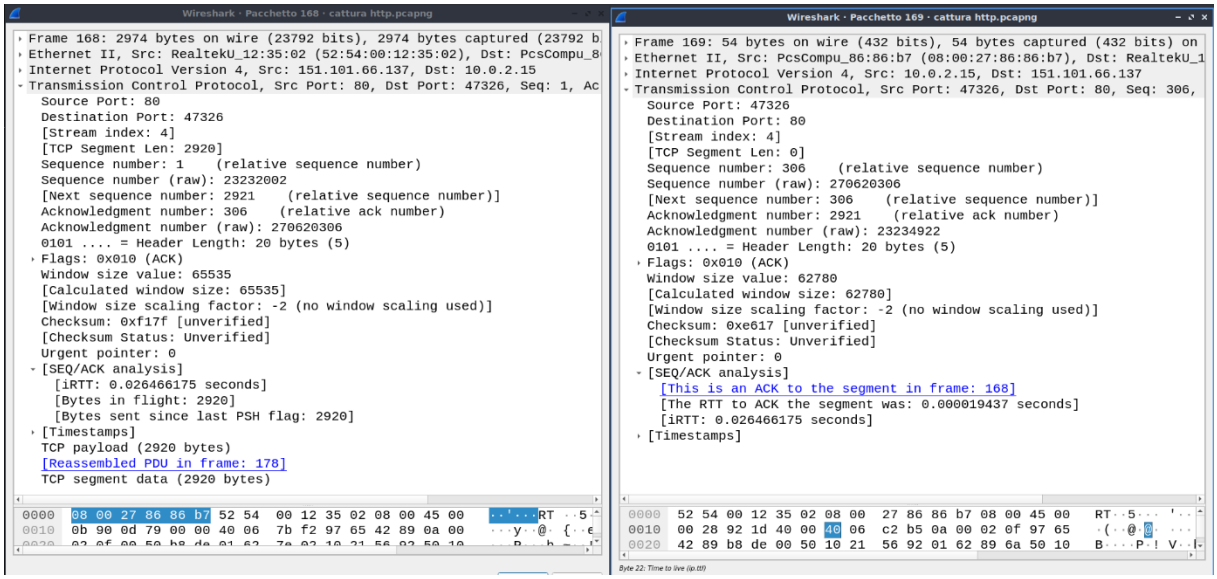


Figura 13

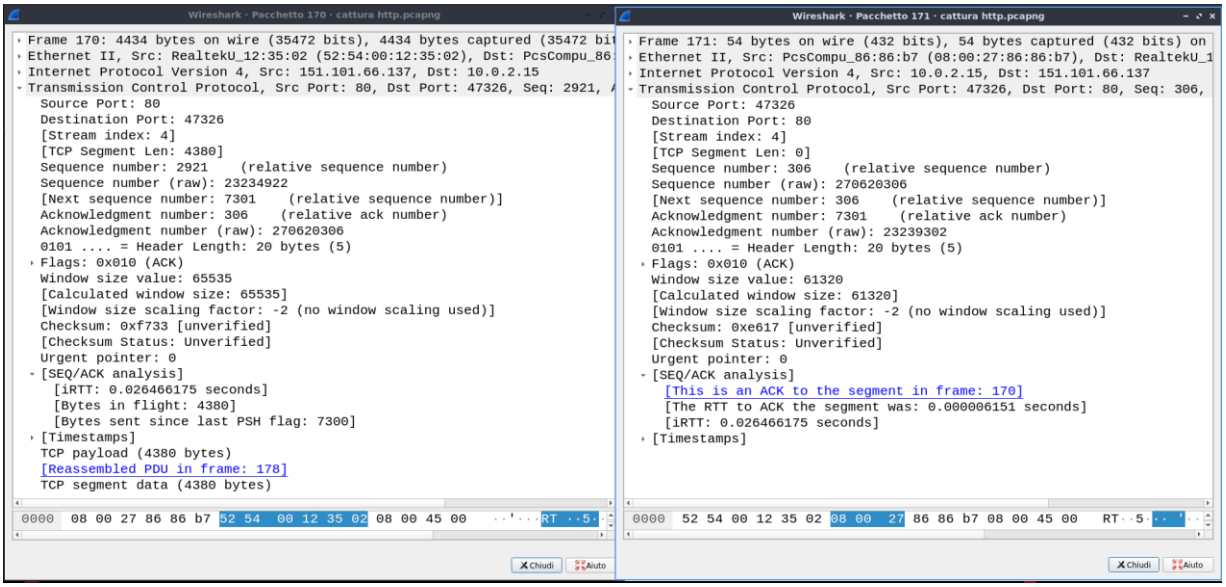


Figura 14

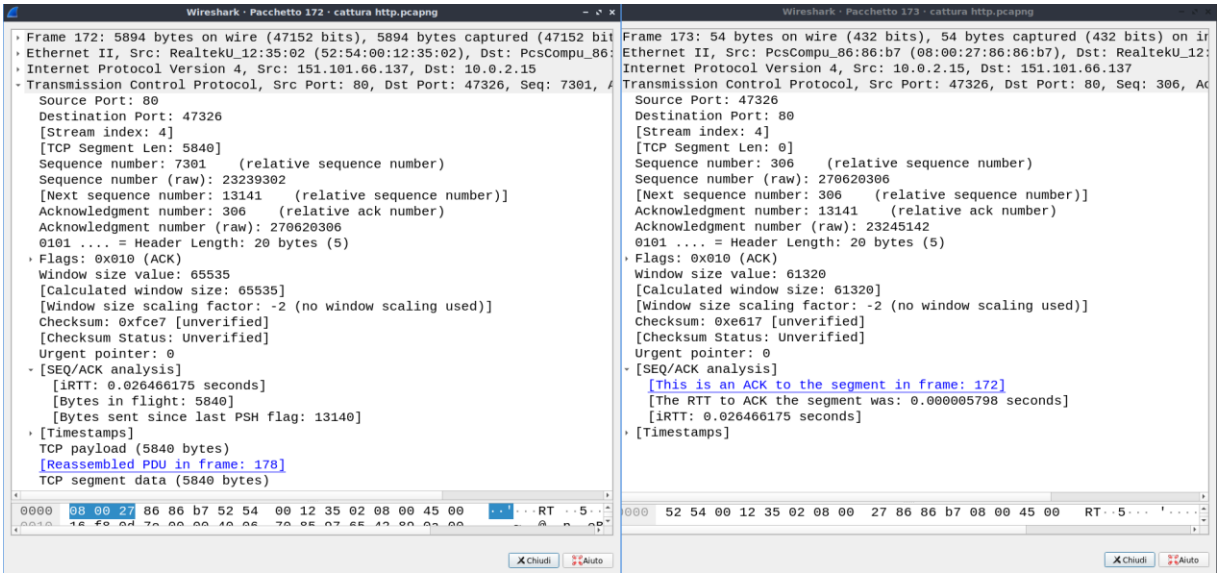


Figura 15

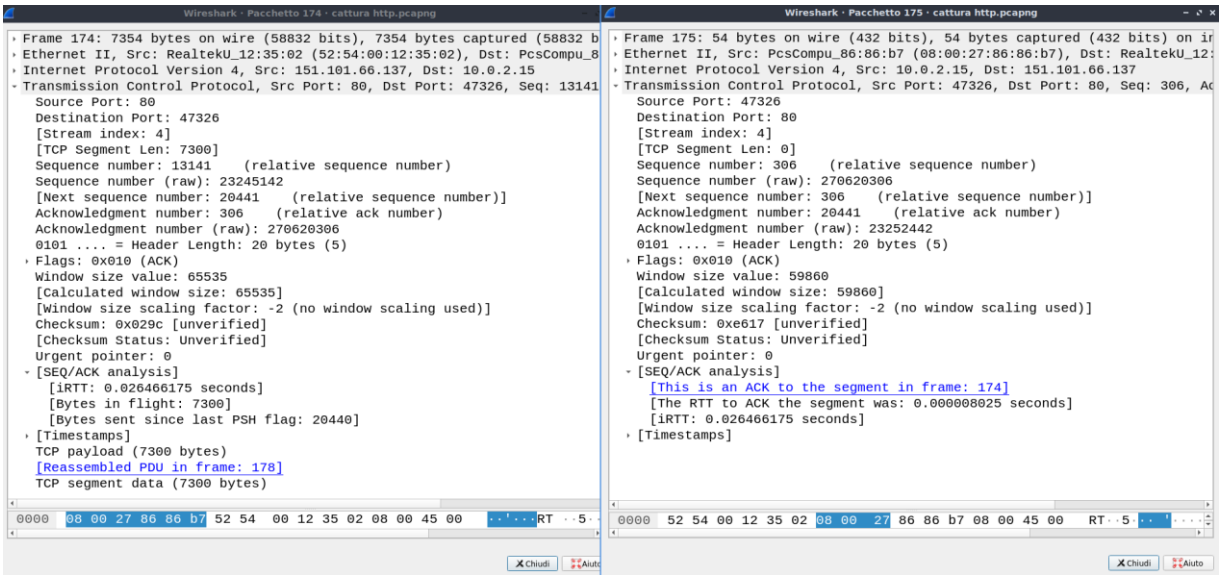


Figura 16

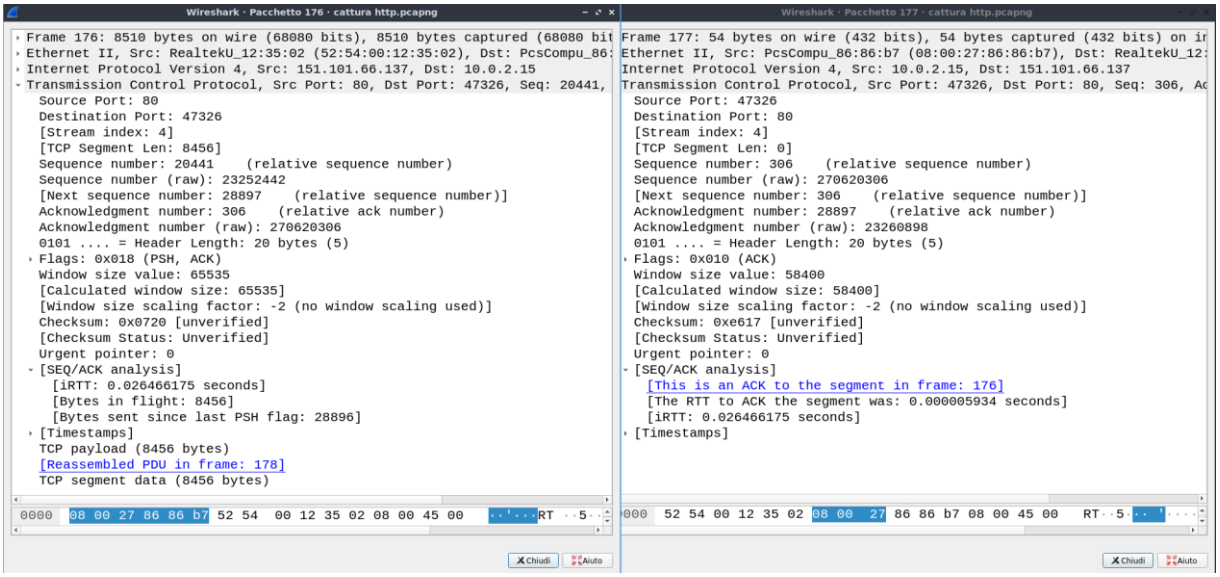


Figura 17

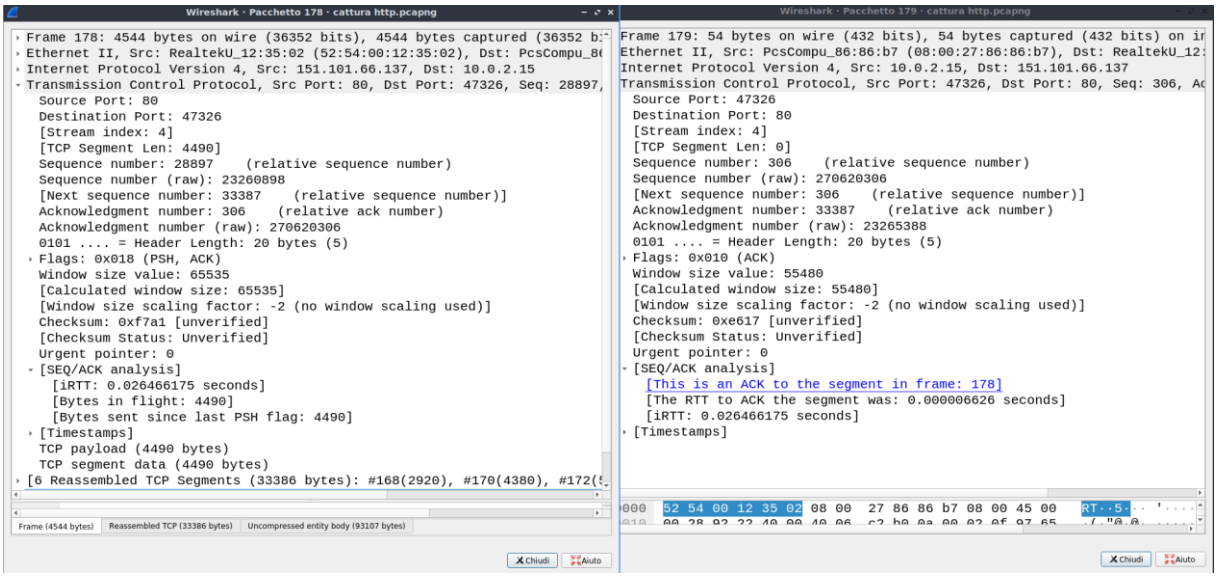


Figura 18

TCP implementa anche la funzionalità di controllo del flusso, per evitare che il mittente invii quantità di dati non gestibili dal ricevente e che di conseguenza potrebbero causare buffer overflow nel ricevente.

Nelle Figure 12 – 18, si può notare che sia il server sia il client hanno impostato il valore della dimensione del proprio buffer di ricezione, indicato come “Window Size Value”, e che tale valore del client diminuisce alla ricezione di nuovi pacchetti.

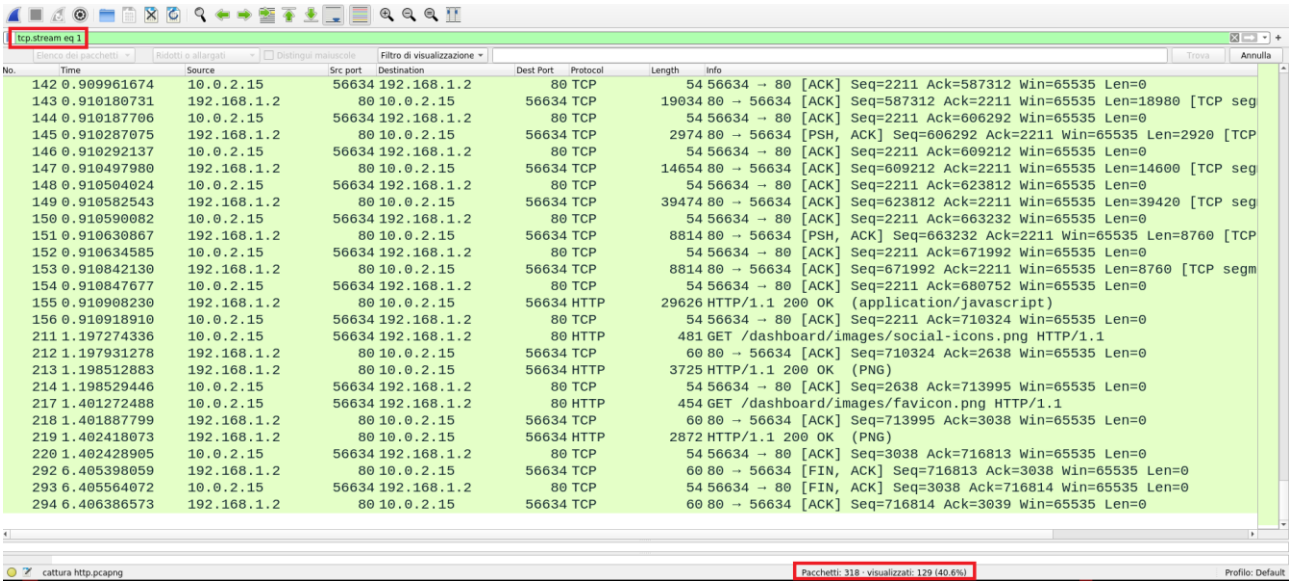
Analisi delle metriche chiave

Numero dei pacchetti

Come descritto in precedenza, per il trasferimento della pagina web e dei file in essa linkati sono state aperte 3 connessioni TCP, identificate da Wireshark come flussi TCP 1, 3 e 4.

Nel flusso TCP 1, come mostrato in Figura 19, sono stati scambiati 129 pacchetti di cui:

- tre pacchetti di handshake per l'apertura della connessione (pacchetti 3, 4 e 5)
- 123 pacchetti per il trasferimento di otto file (con i rispettivi ACK)
- tre pacchetti per la chiusura della connessione (pacchetti 292, 293 e 294).



No.	Time	Source	Src port	Destination	Dest Port	Protocol	Length	Info
142	0.909961674	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=2211 Ack=587312 Win=65535 Len=0
143	0.910180731	192.168.1.2	80	10.0.2.15	56634	TCP	19034	80 → 56634 [ACK] Seq=587312 Ack=2211 Win=65535 Len=18980 [TCP seg
144	0.910187706	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=2211 Ack=606292 Win=65535 Len=0
145	0.910287075	192.168.1.2	80	10.0.2.15	56634	TCP	2974	80 → 56634 [PSH, ACK] Seq=606292 Ack=2211 Win=65535 Len=2920 [TCP
146	0.910292137	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=2211 Ack=609212 Win=65535 Len=0
147	0.910497980	192.168.1.2	80	10.0.2.15	56634	TCP	14654	80 → 56634 [ACK] Seq=609212 Ack=2211 Win=65535 Len=14600 [TCP seg
148	0.910504024	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=2211 Ack=623812 Win=65535 Len=0
149	0.910582543	192.168.1.2	80	10.0.2.15	56634	TCP	39474	80 → 56634 [ACK] Seq=623812 Ack=2211 Win=65535 Len=39420 [TCP seg
150	0.910590082	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=2211 Ack=663232 Win=65535 Len=0
151	0.910630867	192.168.1.2	80	10.0.2.15	56634	TCP	8814	80 → 56634 [PSH, ACK] Seq=663232 Ack=2211 Win=65535 Len=8760 [TCP
152	0.910634585	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=2211 Ack=671992 Win=65535 Len=0
153	0.910842130	192.168.1.2	80	10.0.2.15	56634	TCP	8814	80 → 56634 [ACK] Seq=671992 Ack=2211 Win=65535 Len=8760 [TCP segm
154	0.910847677	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=2211 Ack=680752 Win=65535 Len=0
155	0.910908230	192.168.1.2	80	10.0.2.15	56634	HTTP	29626	HTTP/1.1 200 OK (application/javascript)
156	0.910918910	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=2211 Ack=710324 Win=65535 Len=0
211	1.197274336	10.0.2.15	56634	192.168.1.2	80	HTTP	481	GET /dashboard/images/social-icons.png HTTP/1.1
212	1.197931278	192.168.1.2	80	10.0.2.15	56634	TCP	60	80 → 56634 [ACK] Seq=710324 Ack=2638 Win=65535 Len=0
213	1.198512883	192.168.1.2	80	10.0.2.15	56634	HTTP	3725	HTTP/1.1 200 OK (PNG)
214	1.198529446	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=2638 Ack=713995 Win=65535 Len=0
217	1.401272488	10.0.2.15	56634	192.168.1.2	80	HTTP	454	GET /dashboard/images/favicon.png HTTP/1.1
218	1.401887799	192.168.1.2	80	10.0.2.15	56634	TCP	60	80 → 56634 [ACK] Seq=713995 Ack=3038 Win=65535 Len=0
219	1.402418073	192.168.1.2	80	10.0.2.15	56634	HTTP	2872	HTTP/1.1 200 OK (PNG)
220	1.402428905	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [ACK] Seq=3038 Ack=716813 Win=65535 Len=0
292	6.405398059	192.168.1.2	80	10.0.2.15	56634	TCP	60	80 → 56634 [FIN, ACK] Seq=716813 Ack=3038 Win=65535 Len=0
293	6.405564072	10.0.2.15	56634	192.168.1.2	80	TCP	54	56634 → 80 [FIN, ACK] Seq=3038 Ack=716814 Win=65535 Len=0
294	6.406386573	192.168.1.2	80	10.0.2.15	56634	TCP	60	80 → 56634 [ACK] Seq=716814 Ack=3039 Win=65535 Len=0

Figura 19

Per trasferire un file utilizzando TCP, se viene inviato un ACK per ogni segmento, il numero dei pacchetti totali scambiati per tale trasferimento deve essere necessariamente un numero pari.

Ciò è accaduto sempre in tutti i tre flussi TCP, eccetto in un unico caso nel flusso 1, in cui il server ha inviato i pacchetti 46 e 49 (visibili in Figura 20) mentre il client ha inviato un unico pacchetto di ACK cumulativo visibile in Figura 21 (pacchetto numero 50).

Infatti, si può notare che la somma di

numero di sequenza del pacchetto 46 + dimensione del segmento TCP del pacchetto 46 + dimensione del segmento TCP del pacchetto 49

è esattamente il numero di acknowledge nel pacchetto di ACK cumulativo (cioè 23211057 + 1460 + 16060 = 23228577).

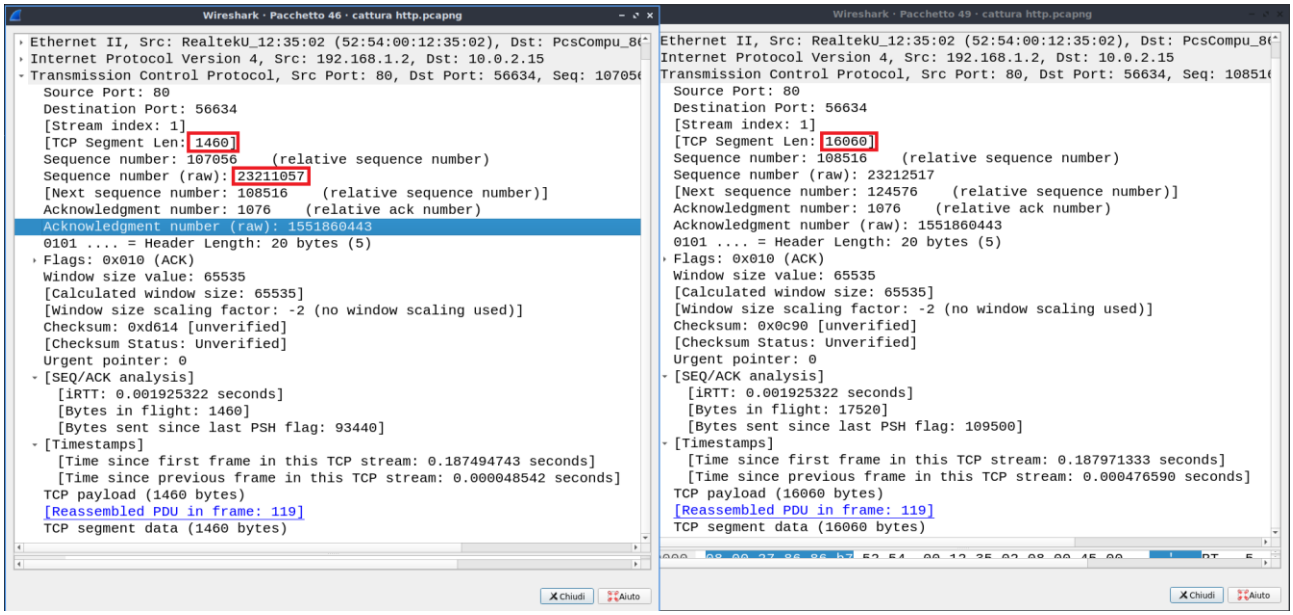


Figura 20

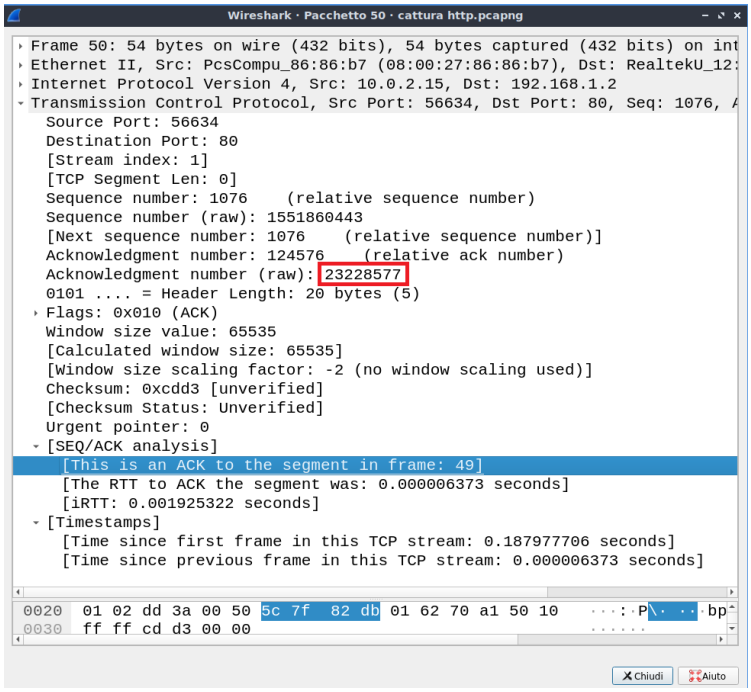
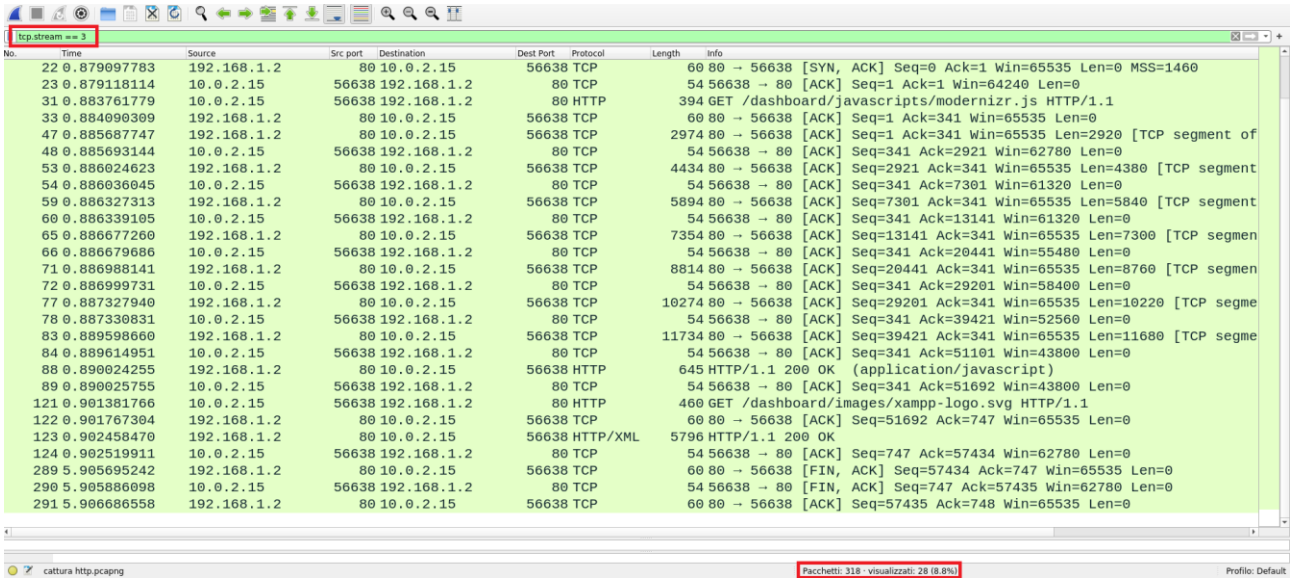


Figura 21

Nel flusso TCP 3, come mostrato in Figura 22, sono stati scambiati 28 pacchetti di cui:

- tre pacchetti di handshake per l'apertura della connessione (pacchetti 18, 22 e 23)
- 22 pacchetti per il trasferimento di due file (con i rispettivi ACK)
- tre pacchetti per la chiusura della connessione (i pacchetti 289, 290 e 291).

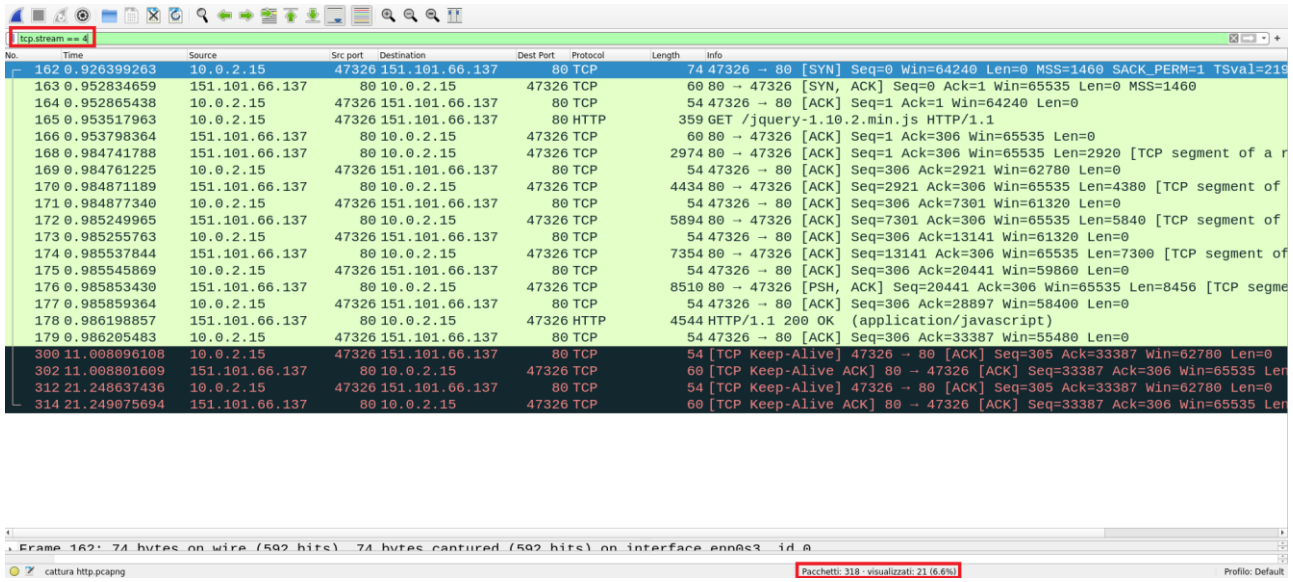


No.	Time	Source	Src port	Destination	Dest Port	Protocol	Length	Info
22	0.879097783	192.168.1.2	80	10.0.2.15	56638	TCP	60	80 → 56638 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
23	0.879118114	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
31	0.883761779	10.0.2.15	56638	192.168.1.2	80	HTTP	394	GET /dashboard/javascripts/modernizr.js HTTP/1.1
33	0.884090309	192.168.1.2	80	10.0.2.15	56638	TCP	60	80 → 56638 [ACK] Seq=1 Ack=341 Win=65535 Len=0
47	0.885687747	192.168.1.2	80	10.0.2.15	56638	TCP	2974	80 → 56638 [ACK] Seq=1 Ack=341 Win=65535 Len=2920 [TCP segment of
48	0.885693144	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=2921 Win=62780 Len=0
53	0.886024623	192.168.1.2	80	10.0.2.15	56638	TCP	4434	80 → 56638 [ACK] Seq=2921 Ack=341 Win=65535 Len=4380 [TCP segment
54	0.886036045	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=7301 Win=61320 Len=0
59	0.886327313	192.168.1.2	80	10.0.2.15	56638	TCP	5894	80 → 56638 [ACK] Seq=7301 Ack=341 Win=65535 Len=5840 [TCP segment
60	0.886339105	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=13141 Win=61320 Len=0
65	0.886677260	192.168.1.2	80	10.0.2.15	56638	TCP	7354	80 → 56638 [ACK] Seq=13141 Ack=341 Win=65535 Len=7300 [TCP segmen
66	0.886679686	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=20441 Win=55480 Len=0
71	0.886988141	192.168.1.2	80	10.0.2.15	56638	TCP	8814	80 → 56638 [ACK] Seq=20441 Ack=341 Win=65535 Len=8760 [TCP segmen
72	0.886999731	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=29201 Win=58400 Len=0
77	0.887327940	192.168.1.2	80	10.0.2.15	56638	TCP	10274	80 → 56638 [ACK] Seq=29201 Ack=341 Win=65535 Len=10220 [TCP segme
78	0.887330831	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=39421 Win=52560 Len=0
83	0.889598660	192.168.1.2	80	10.0.2.15	56638	TCP	11734	80 → 56638 [ACK] Seq=39421 Ack=341 Win=65535 Len=11680 [TCP segme
84	0.889614951	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=51101 Win=43800 Len=0
88	0.890024255	192.168.1.2	80	10.0.2.15	56638	HTTP	645	HTTP/1.1 200 OK (application/javascript)
89	0.890025755	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=341 Ack=51692 Win=43800 Len=0
121	0.901381766	10.0.2.15	56638	192.168.1.2	80	HTTP	460	GET /dashboard/images/xampp-logo.svg HTTP/1.1
122	0.901767304	192.168.1.2	80	10.0.2.15	56638	TCP	60	80 → 56638 [ACK] Seq=51692 Ack=747 Win=65535 Len=0
123	0.902458470	192.168.1.2	80	10.0.2.15	56638	HTTP/XML	5796	HTTP/1.1 200 OK
124	0.902519911	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [ACK] Seq=747 Ack=57434 Win=62780 Len=0
289	5.905695242	192.168.1.2	80	10.0.2.15	56638	TCP	60	80 → 56638 [FIN, ACK] Seq=57434 Ack=747 Win=65535 Len=0
290	5.905886098	10.0.2.15	56638	192.168.1.2	80	TCP	54	56638 → 80 [FIN, ACK] Seq=747 Ack=57435 Win=62780 Len=0
291	5.906686558	192.168.1.2	80	10.0.2.15	56638	TCP	60	80 → 56638 [ACK] Seq=57435 Ack=748 Win=65535 Len=0

Figura 22

Infine, nel flusso TCP 4, visibile in Figura 23, sono stati scambiati 21 pacchetti di cui:

- tre pacchetti di handshake per l'apertura della connessione (pacchetti 162, 163 e 164)
- 14 pacchetti per il trasferimento di un file (con i rispettivi ACK)
- due pacchetti per mantenere attiva la connessione (pacchetti 300 e 312 con etichetta "TCP Keep-Alive") e
- altri due pacchetti per il corrispondente ACK (pacchetti 302 e 314 con etichetta "TCP Keep-Alive ACK").



No.	Time	Source	Src port	Destination	Dest Port	Protocol	Length	Info
162	0.926399263	10.0.2.15	47326	151.101.66.137	80	TCP	74	47326 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=219
163	0.952834659	151.101.66.137	80	10.0.2.15	47326	TCP	60	80 → 47326 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
164	0.952865438	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
165	0.953517963	10.0.2.15	47326	151.101.66.137	80	HTTP	359	GET /jquery-1.10.2.min.js HTTP/1.1
166	0.953798364	151.101.66.137	80	10.0.2.15	47326	TCP	60	80 → 47326 [ACK] Seq=1 Ack=306 Win=65535 Len=0
168	0.984741788	151.101.66.137	80	10.0.2.15	47326	TCP	2974	80 → 47326 [ACK] Seq=1 Ack=306 Win=65535 Len=2920 [TCP segment of a r
169	0.984761225	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=2921 Win=62780 Len=0
170	0.984871189	151.101.66.137	80	10.0.2.15	47326	TCP	4434	80 → 47326 [ACK] Seq=2921 Ack=306 Win=65535 Len=4380 [TCP segment of
171	0.984877340	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=7301 Win=61320 Len=0
172	0.985249965	151.101.66.137	80	10.0.2.15	47326	TCP	5894	80 → 47326 [ACK] Seq=7301 Ack=306 Win=65535 Len=5840 [TCP segment of
173	0.985255763	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=13141 Win=61320 Len=0
174	0.985537844	151.101.66.137	80	10.0.2.15	47326	TCP	7354	80 → 47326 [ACK] Seq=13141 Ack=306 Win=65535 Len=7300 [TCP segment of
175	0.985545869	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=20441 Win=59860 Len=0
176	0.985853430	151.101.66.137	80	10.0.2.15	47326	TCP	8510	80 → 47326 [PSH, ACK] Seq=20441 Ack=306 Win=65535 Len=8456 [TCP segme
177	0.985859364	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=28897 Win=58400 Len=0
178	0.986198857	151.101.66.137	80	10.0.2.15	47326	HTTP	4544	HTTP/1.1 200 OK (application/javascript)
179	0.986205483	10.0.2.15	47326	151.101.66.137	80	TCP	54	47326 → 80 [ACK] Seq=306 Ack=33387 Win=55480 Len=0
300	11.008096108	10.0.2.15	47326	151.101.66.137	80	TCP	54	[TCP Keep-Alive] 47326 → 80 [ACK] Seq=305 Ack=33387 Win=62780 Len=0
302	11.008801609	151.101.66.137	80	10.0.2.15	47326	TCP	60	[TCP Keep-Alive ACK] 80 → 47326 [ACK] Seq=33387 Ack=306 Win=65535 Len
312	21.248637436	10.0.2.15	47326	151.101.66.137	80	TCP	54	[TCP Keep-Alive] 47326 → 80 [ACK] Seq=305 Ack=33387 Win=62780 Len=0
314	21.249075694	151.101.66.137	80	10.0.2.15	47326	TCP	60	[TCP Keep-Alive ACK] 80 → 47326 [ACK] Seq=33387 Ack=306 Win=65535 Len

Figura 23

Come mostrato a sinistra nelle Figure 24 e 25, si nota che i pacchetti “TCP Keep-Alive” inviati dal client sono pacchetti con il flag ACK = 1 e segmento TCP di zero byte.

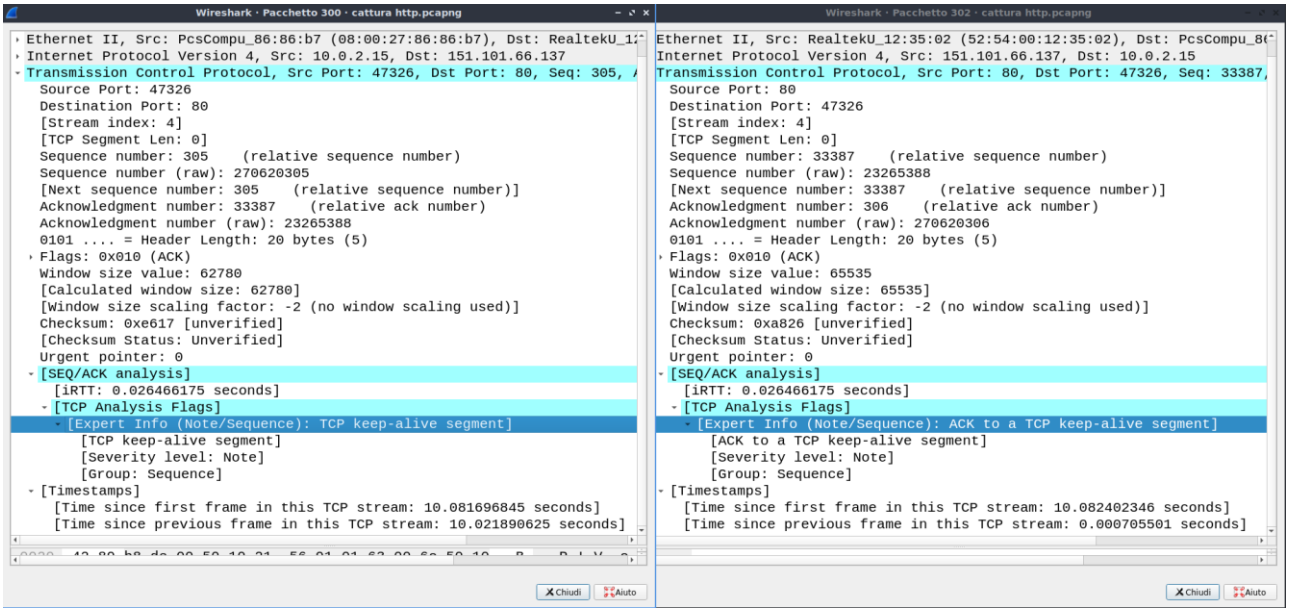


Figura 24

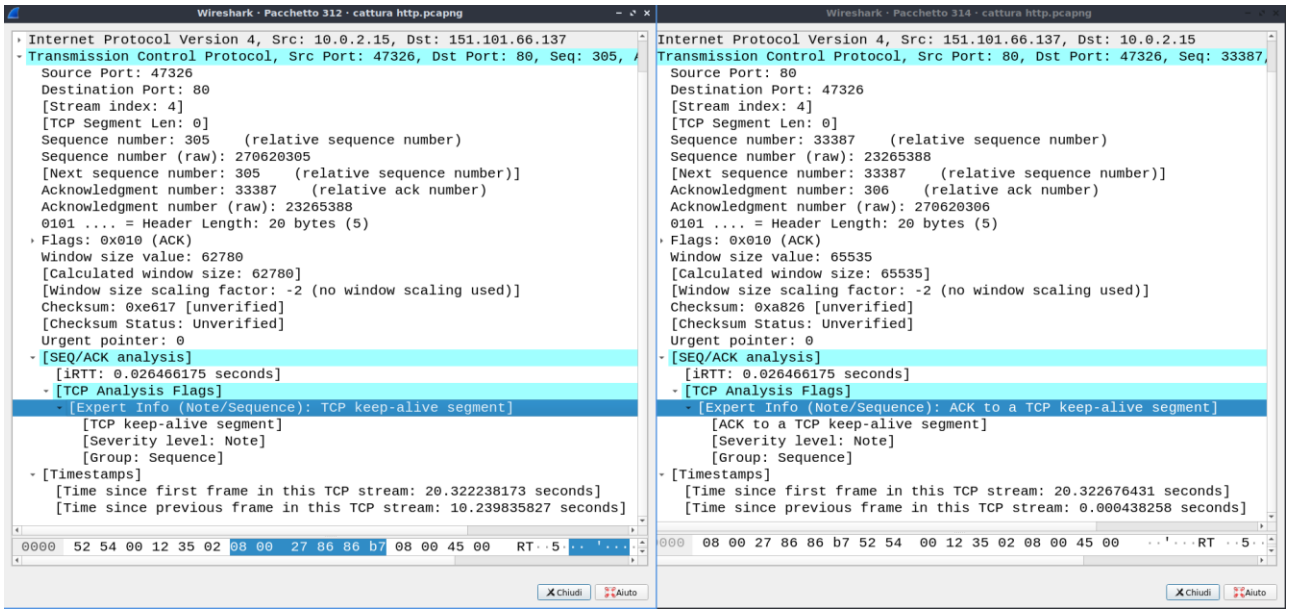


Figura 25

In totale sono stati scambiati 129 + 28 + 21 = 178 pacchetti per il trasferimento della pagina web e dei file in essa linkati.

Tempi di latenza

Con Wireshark è possibile calcolare il tempo impiegato per la fase di handshake, per trasferire un file e per chiudere la connessione, da cui si ricava facilmente anche il tempo totale in cui la connessione è stata attiva.

Apertura della connessione

In Figura 26, si nota che nel flusso TCP 1 il tempo trascorso

- tra la ricezione del pacchetto SYN e l'invio del pacchetto SYN-ACK è di 1,861928 millisecondi
- tra la ricezione del pacchetto SYN-ACK e l'invio dell'ACK corrispondente è di 0,063394 millisecondi.

Quindi il tempo richiesto per l'apertura della connessione è stato di 1,925322 millisecondi.

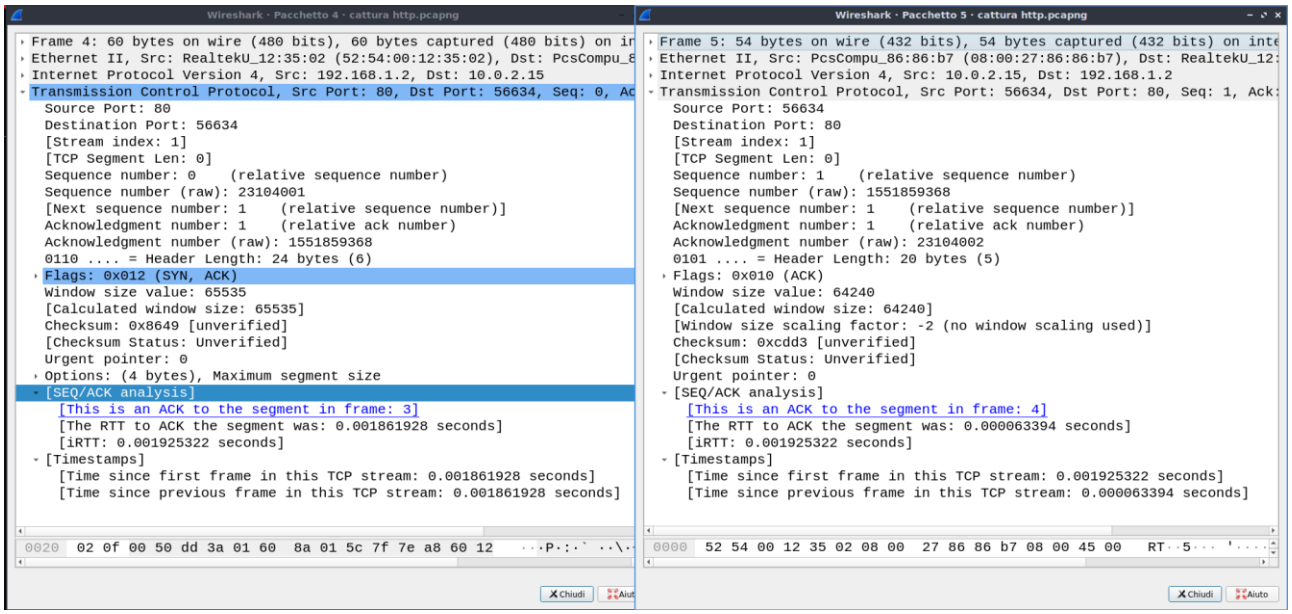


Figura 26

Similmente, in Figura 27, si nota che nel flusso TCP 3 il tempo trascorso

- tra la ricezione del pacchetto SYN e l'invio del pacchetto SYN-ACK è di 2,171166 millisecondi
- tra la ricezione del pacchetto SYN-ACK e l'invio dell'ACK corrispondente è di 0,020331 millisecondi.

Quindi il tempo trascorso per l'apertura della connessione è stato di 2,191497 millisecondi.

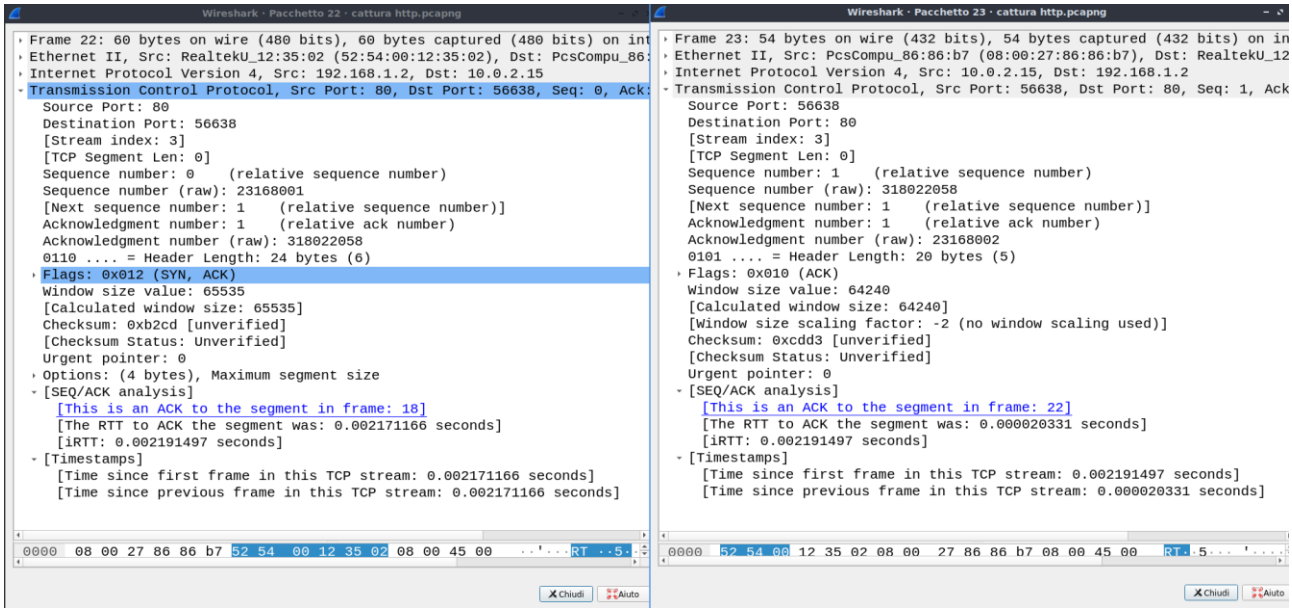


Figura 27

Infine, in Figura 28 si nota che nel flusso TCP 4, il tempo trascorso

- tra la ricezione del pacchetto SYN e l'invio del pacchetto SYN-ACK è di 26,435396 millisecondi
- tra la ricezione del pacchetto SYN-ACK e l'invio dell'ACK corrispondente è di 0,030779 millisecondi.

Quindi il tempo trascorso per l'apertura della connessione è stato di 26,466175 millisecondi.

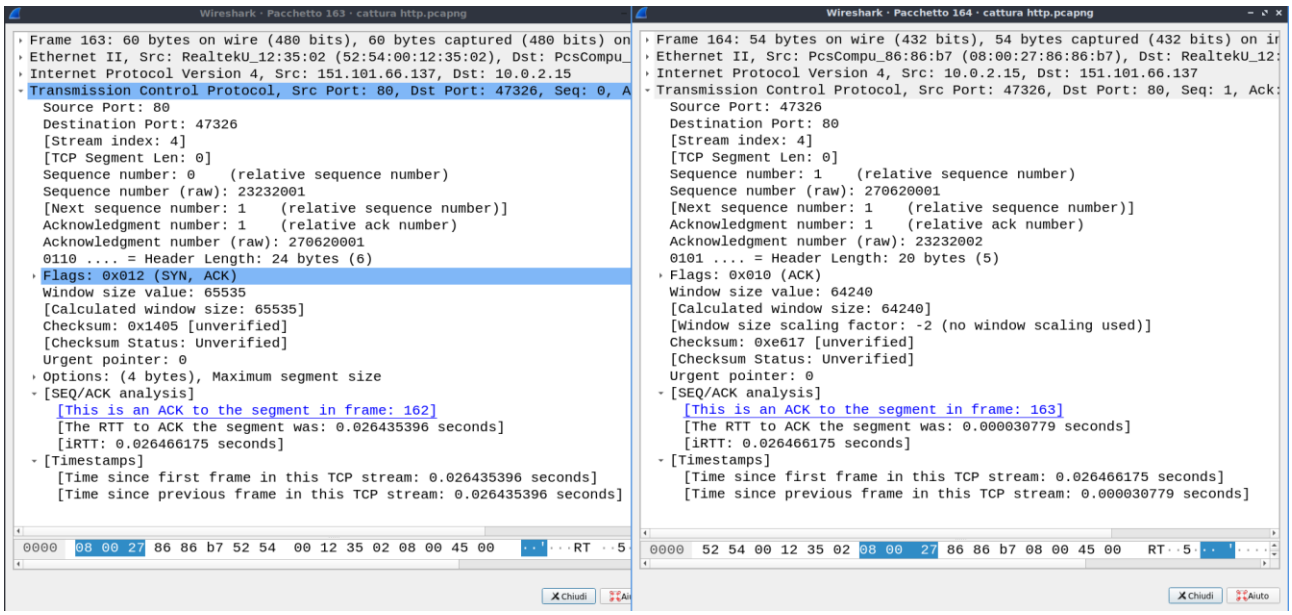


Figura 28

Nei flussi TCP 1 e 3 i pacchetti sono scambiati tra il server XAMPP locale e la macchina virtuale, entrambi in esecuzione sullo stesso computer fisico, mentre nel flusso TCP 4 i pacchetti vengono scambiati tra la macchina virtuale e un server remoto; ciò spiega i tempi di latenza maggiori in quest'ultimo flusso TCP.

Chiusura della connessione

In Figura 29 si nota che nel flusso TCP 1

- dopo che il client ha inviato l'ultimo pacchetto al server, quest'ultimo ha atteso circa 5 secondi, poi ha inviato il pacchetto FIN-ACK per chiudere la connessione
- il client ha risposto con un altro pacchetto FIN-ACK in 0,166013 millisecondi
- il server ha inviato il pacchetto di ACK finale in 0,822501 millisecondi.

Quindi il tempo trascorso per la chiusura della connessione è stato di 0,988514 millisecondi, mentre l'intero flusso TCP è stato attivo per circa 5,71 secondi.

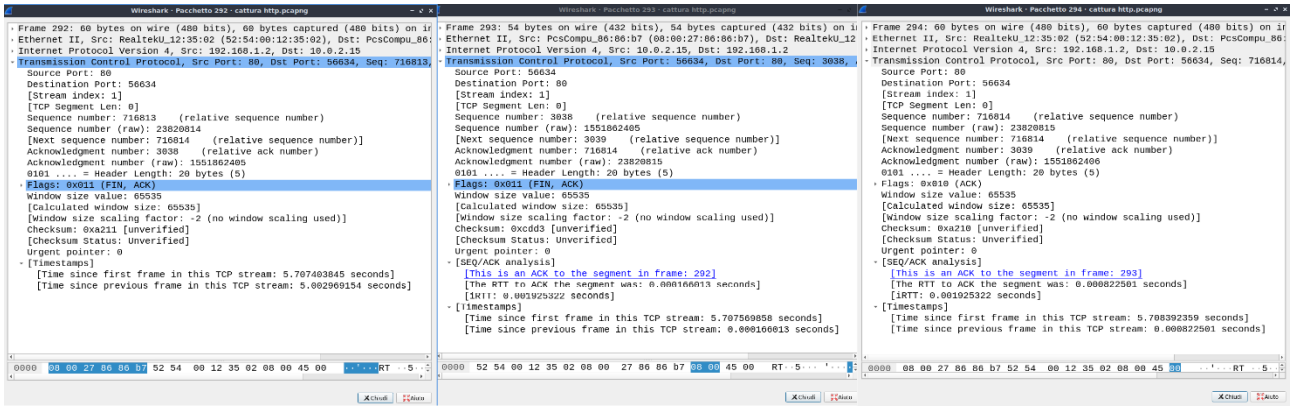


Figura 29

Similmente, in Figura 30 si nota che nel flusso TCP 3

- dopo che il client ha inviato l'ultimo pacchetto al server, quest'ultimo ha atteso circa 5 secondi, poi ha inviato il pacchetto FIN-ACK per chiudere la connessione
- il client ha risposto con un altro pacchetto FIN-ACK in 0,190856 millisecondi
- il server ha inviato il pacchetto di ACK finale in 0,800460 millisecondi.

Quindi il tempo trascorso per la chiusura della connessione è stato di 0,991316 millisecondi, mentre l'intero flusso TCP è stato attivo per circa 5,03 secondi.

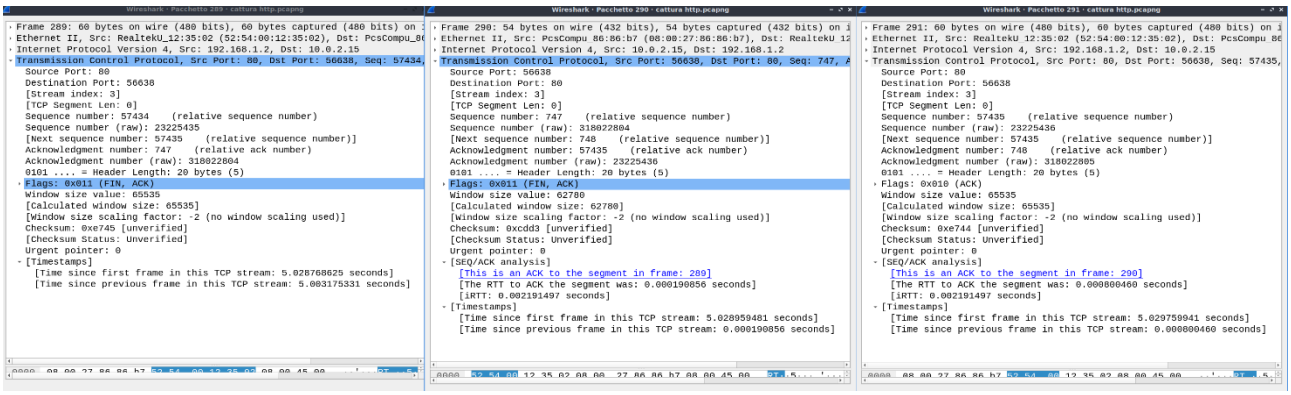


Figura 30

Invece, nel flusso TCP 4, come già mostrato in precedenza, la connessione non viene chiusa:

- in Figura 24 si nota che il client ha inviato il primo pacchetto TCP Keep-Alive circa 10,02 secondi dal pacchetto precedente; il server ha risposto con un ACK in 0,705501 millisecondi
- in Figura 25 si nota che il client ha inviato il secondo pacchetto TCP Keep-Alive circa 10,23 secondi dal pacchetto precedente; il server ha risposto con un ACK in 0,438258 millisecondi.

Quindi l'intero flusso TCP è stato attivo per poco più di 20 secondi fino al termine della cattura.

Segue un riepilogo dei tempi nei tre flussi TCP analizzati.

	Flusso TCP 1	Flusso TCP 3	Flusso TCP 4
Tempo tra SYN e SYN-ACK	0,001861928 s	0,002171166 s	0,026435396 s
Tempo tra SYN-ACK e ACK	0,000063394 s	0,000020331 s	0,000030779 s
Tempo totale per apertura connessione	0,001925322 s	0,002191497 s	0,026466175 s
Numero dei file trasferiti	8	2	1
Tempo totale per trasferimento dei file	0,70333187 s	0,024202256 s	0,033778871 s
Tempo attesa prima di chiudere connessione	5,002969154 s	5,003175331 s	-
Tempo tra FIN e FIN-ACK Tempo tra FIN-ACK e ACK	0,000166013 s 0,000822501 s	0,000190856 s 0,000800460 s	-
Tempo totale per chiusura connessione	0,000988514 s	0,000991316 s	-
Tempo attesa prima di invio primo pacchetto KeepAlive	-	-	10,021890625 s
Tempo tra primo pacchetto KeepAlive e relativo ACK	-	-	0,000705501 s
Tempo attesa prima di invio secondo pacchetto KeepAlive	-	-	10,23983527 s
Tempo tra secondo pacchetto KeepAlive 2 relativo ACK	-	-	0,000438258 s
Tempo attività del flusso TCP	5,70921486 s	5,0305604 s	20,3231147 s