

## **Project 2: Wireshark Packet Capture and Analysis**

**Created by Luis Umana**

## 1.1. Data Capture

### Location/Device

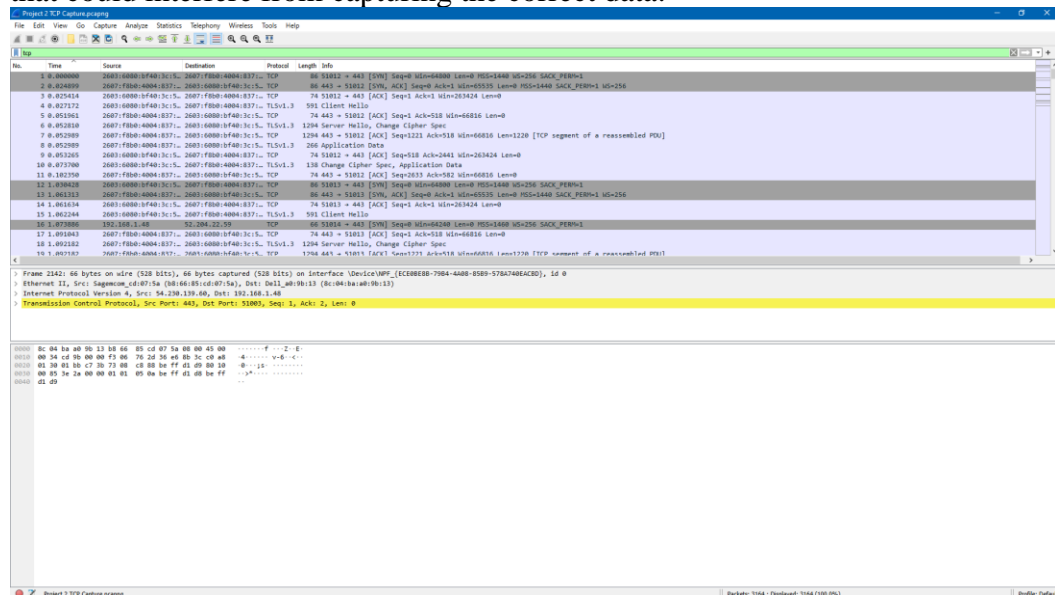
The project was performed at my home with a Dell G7 Laptop. I used an ethernet cable instead of Wi-Fi, in order to isolate the TCP traffic of YouTube on WireShark.

## Wireshark filters for capture and display

### Capture

#### 1. TCP for Capture filter

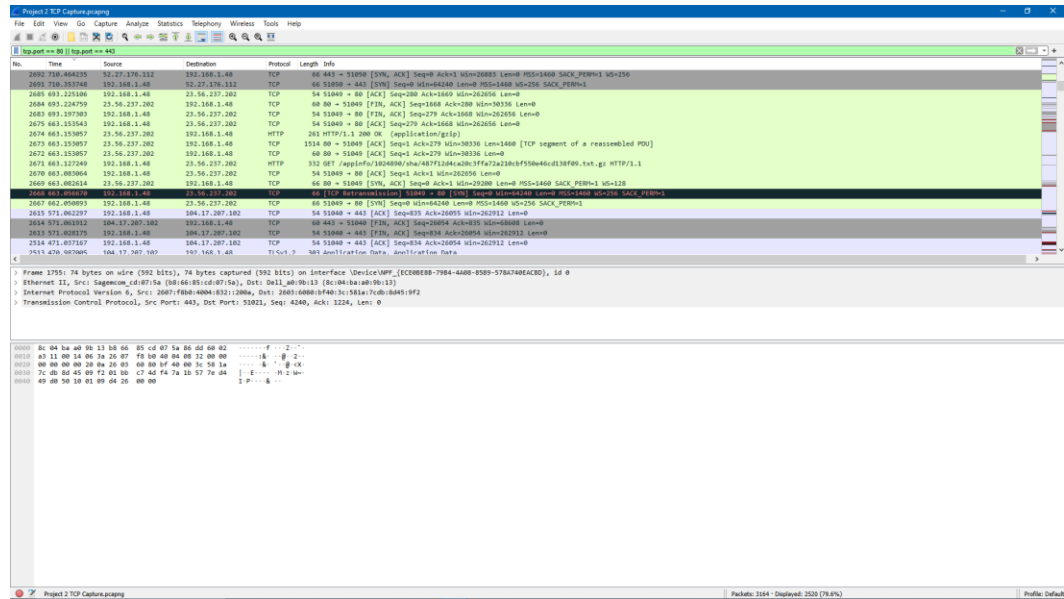
- I decided to use a general TCP capture because, the project required us to capture all TCP and HTTP traffic; HTTP is part of the TCP family, so they can both be captured with the same filter.
- Since I was using an ethernet connection, rather than using the Wi-Fi, I could easily isolate the traffic coming from only my laptop. By closing all applications and websites while only having YouTube open, I can reduce the amount of traffic that could interfere from capturing the correct data.



(Screenshot of all TCP traffic)

#### 2. HTTP Traffic

- The traffic of YouTube can be displayed by using two filters:
- `tcp.port == 80 || tcp.port == 443`
- As stated in the description of the project, port 80 can separate the HTTP packets from the TCP packets and port 443 can capture some HTTP packets that are secure.

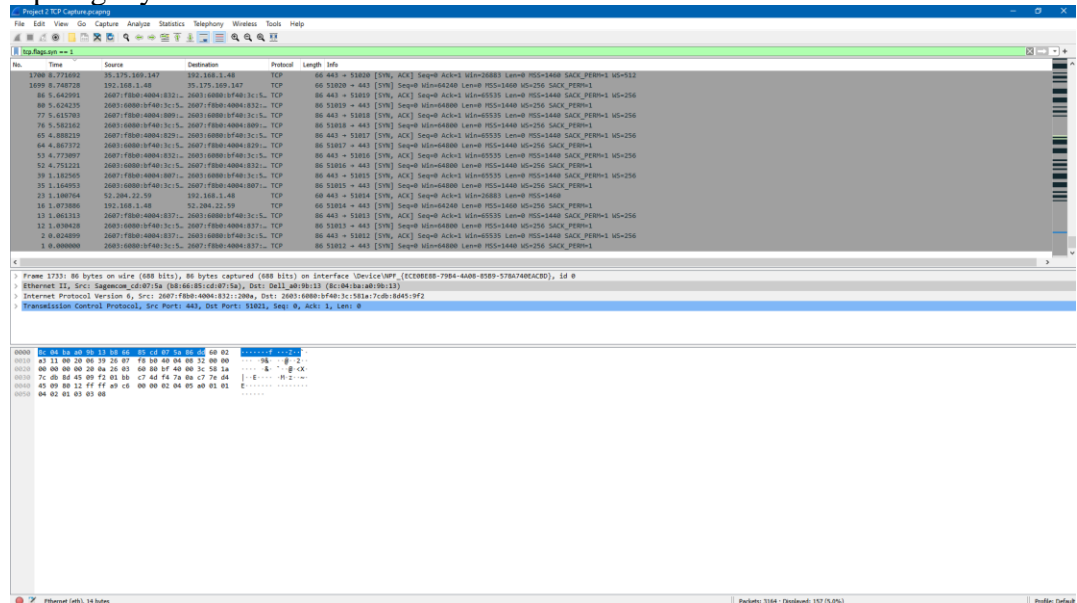


- (Screenshot of all HTTP traffic to/from youtube)
- Note: If tcp.port 80 is only used then it will only show 16 packets while tcp.port 443 shows 2504 packets.

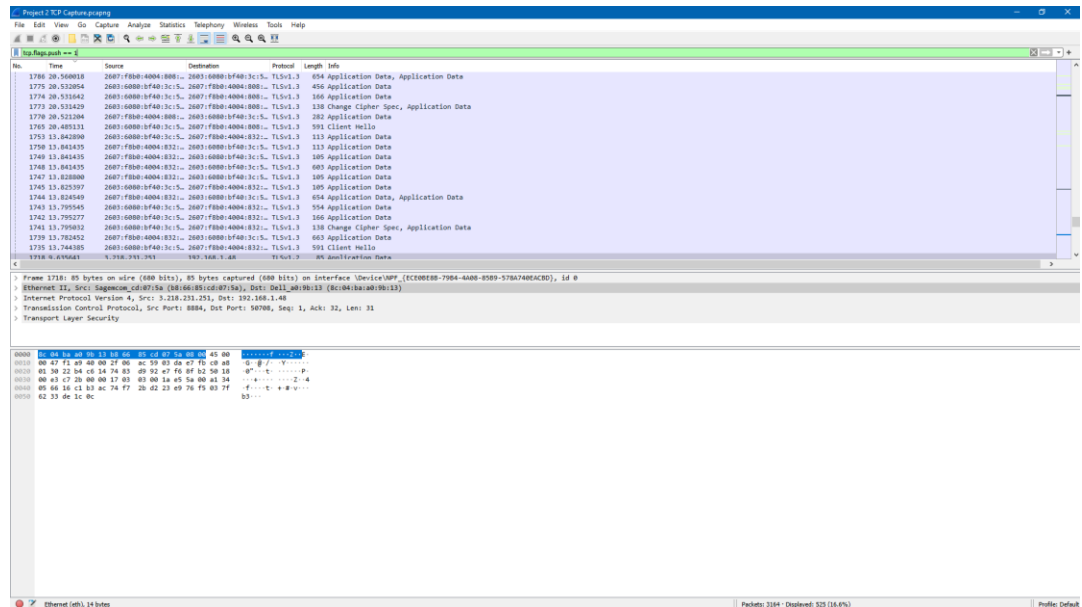
## Display

### 1. SYN, PSH, and RST filters for all TCP packets

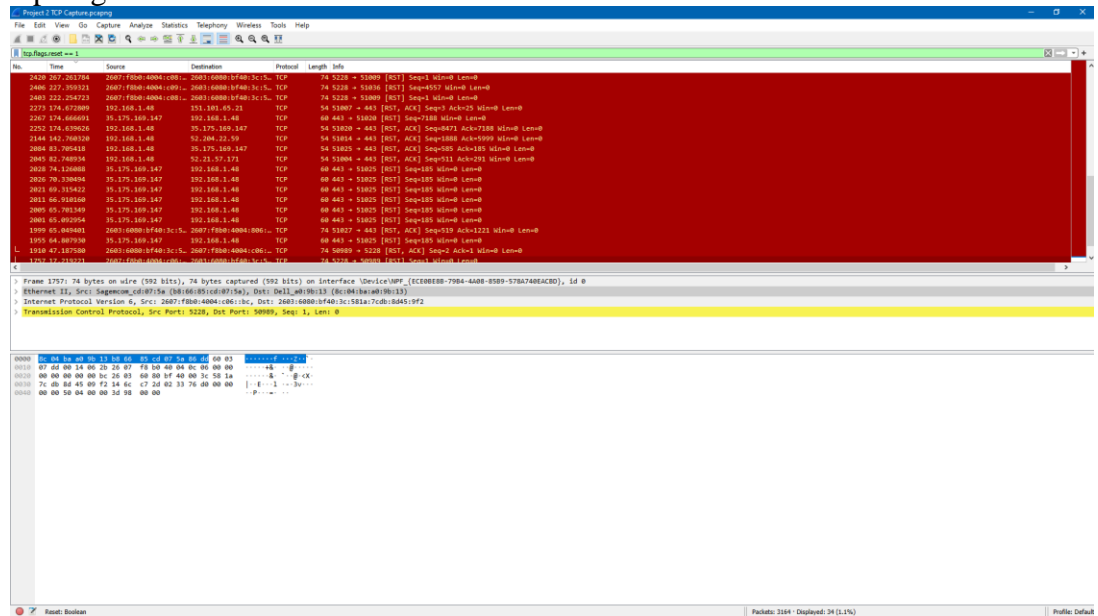
- tcp.flags.syn == 1



- tcp.flags.push == 1



tcp.flags.reset == 1



In this case, the filters are pretty much self-explanatory since the filters displays only the flags, and each filter will give either SYN, PSH, or RST.

2. Packets sent by our computer vs Received from YouTube.

- tcp && ip.src==192.168.1.48
- This is the filter that contains only TCP traffic sent from my computer

Project 2 TCP Capture.pcapng

Filter: tcp && ip.src==130.211.15.53

No.	Time	Source	Destination	Protocol	Length	Info
24	1.101307	192.168.1.48	52.204.22.59	TCP	54	51804 → 443 [ACK] Seq=51804 Win=0 Len=0
25	1.101773	192.168.1.48	52.204.22.59	TLSv1.2	571	Client Hello
31	1.112829	192.168.1.48	52.204.22.59	TCP	54	51804 → 443 [ACK] Seq=51804 Win=0 Len=0
33	1.127732	192.168.1.48	52.204.22.59	TCP	54	51804 → 443 [ACK] Seq=51804 Win=0 Len=0
34	1.150725	192.168.1.48	52.204.22.59	TLSv1.2	180	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
35	1.151738	192.168.1.48	52.204.22.59	TLSv1.2	473	Application Data
50	1.260707	192.168.1.48	52.204.22.59	TCP	54	51804 → 443 [ACK] Seq=1263 Ack=5688 Win=63918 Len=0
117	5.805124	192.168.1.48	130.211.16.53	TLSv1.2	192	Application Data
118	5.805422	192.168.1.48	130.211.16.53	TLSv1.2	59	Application Data
122	5.820807	192.168.1.48	130.211.16.53	TCP	54	51800 → 443 [ACK] Seq=170 Ack=1545 Win=1027 Len=0
125	5.82062	192.168.1.48	130.211.16.53	TCP	54	51800 → 443 [ACK] Seq=170 Ack=2975 Win=1027 Len=0
126	5.820733	192.168.1.48	130.211.16.53	TCP	54	51800 → 443 [ACK] Seq=170 Ack=4405 Win=1027 Len=0
129	5.830701	192.168.1.48	130.211.16.53	TCP	54	51800 → 443 [ACK] Seq=170 Ack=935 Win=1027 Len=0
130	5.830704	192.168.1.48	130.211.16.53	TCP	54	51800 → 443 [ACK] Seq=170 Ack=755 Win=1027 Len=0
133	5.833072	192.168.1.48	130.211.16.53	TCP	54	51800 → 443 [ACK] Seq=170 Ack=18125 Win=1027 Len=0
136	5.83284	192.168.1.48	130.211.16.53	TCP	54	51800 → 443 [ACK] Seq=170 Ack=1555 Win=1027 Len=0
137	5.833006	192.168.1.48	130.211.16.53	TCP	54	51800 → 443 [ACK] Seq=170 Ack=1205 Win=1027 Len=0
140	5.834111	192.168.1.48	130.211.16.53	TCP	54	51800 → 443 [ACK] Seq=170 Ack=14415 Win=1027 Len=0

Frame 181: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface vDevice\NPF\_{E000E0B-70B4-4A00-85B0-57B4740ACB0D}, id 0

Ethernet II, Src: Sagecom\_cd:07:5a (08:00:00:00:00:00), Dst: Sagecom\_cd:07:5a (08:00:00:00:00:00)

Internet Protocol Version 4, Src: 192.168.1.48, Dst: 52.204.22.59

Transmission Control Protocol, Src Port: 51804, Dst Port: 443, Seq: 0, Len: 0

0000 50 66 85 cd 07 5a 0c 04 ba a0 70 13 00 00 45 00 f...Z...E

0010 00 54 4c 54 a0 00 00 00 00 00 c0 a0 00 5a 0c 48 f...g...B

0020 16 30 c7 40 01 50 f0 ba 7a 58 00 00 00 00 02 1f...Z...P

0030 fa f0 00 00 00 02 04 05 34 01 03 03 01 01 00 .....

0040 04 02 .....

- tcp && ip.src==130.211.15.53
- This is the filter used that contains only TCP Traffic received from YouTube.
- If you examine the TCP traffic, you will notice how there are multiple different IP going on as the video plays. When searching for the information on the ips, most of these videos either relate to the video, software such as cloudflare, and other ips related to ads, such as amazon. I decided to choose this ip because, it had the most packets and it was directly related to google.

Project 2 TCP Capture.pcapng

Filter: tcp && ip.src==130.211.16.53

No.	Time	Source	Destination	Protocol	Length	Info
119	5.828246	130.211.16.53	192.168.1.48	TCP	60	443 → 51800 [ACK] Seq=1 Ack=139 Win=277 Len=0
120	5.828777	130.211.16.53	192.168.1.48	TLSv1.2	168	Application Data
121	5.828895	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
122	5.829461	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
124	5.829543	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
127	5.830439	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
129	5.830572	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
131	5.831069	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
132	5.831809	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
134	5.832070	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
135	5.832826	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
136	5.833933	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
139	5.834805	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
142	5.834916	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
143	5.835072	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
146	5.836195	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
147	5.836362	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
150	5.837188	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data
151	5.837336	130.211.16.53	192.168.1.48	TLSv1.2	1404	Application Data

Frame 119: 66 bytes on wire (480 bits), 66 bytes captured (480 bits) on interface vDevice\NPF\_{E000E0B-70B4-4A00-85B0-57B4740ACB0D}, id 0

Ethernet II, Src: Sagecom\_cd:07:5a (08:00:00:00:00:00), Dst: Dell\_d0:90:13 (0c:00:00:00:00:13)

Internet Protocol Version 4, Src: 130.211.16.53, Dst: 192.168.1.48

Transmission Control Protocol, Src Port: 443, Dst Port: 51800, Seq: 1, Ack: 139, Len: 0

0000 8c 04 ba a0 70 13 00 66 85 cd 07 5a 00 00 45 00 .....f...Z...E

0010 00 28 fa a0 00 00 3a 00 32 a6 82 d3 10 35 c0 a0 (-...:..5..

0020 03 30 03 10 c7 3a 03 47 32 a3 a7 57 4c 50 50 10 @...@...P

0030 01 15 92 19 00 00 00 00 00 00 00 00 00 00 00 .....

- The TCP sent from my computer consisted of 24.3% of the total TCP, while the traffic received from YouTube consisted of 39.6% of the total TCP.

## YouTube Video used for this project:

- Video Title: “Teaching a Robot Dog to Pee Beer”

- **YouTuber:** “Michael Reeves”
- **Link:** <https://www.youtube.com/watch?v=tqsy9Wtr1qE>

## 1.2. Data Analytics

### TCP packets received from/Sent to YouTube

Components	# of Packets	Percent of total
Received	1252	39.6%
Sent	770	24.3%
HTTP	2504	79.1%
HTTPS	16	0.5%

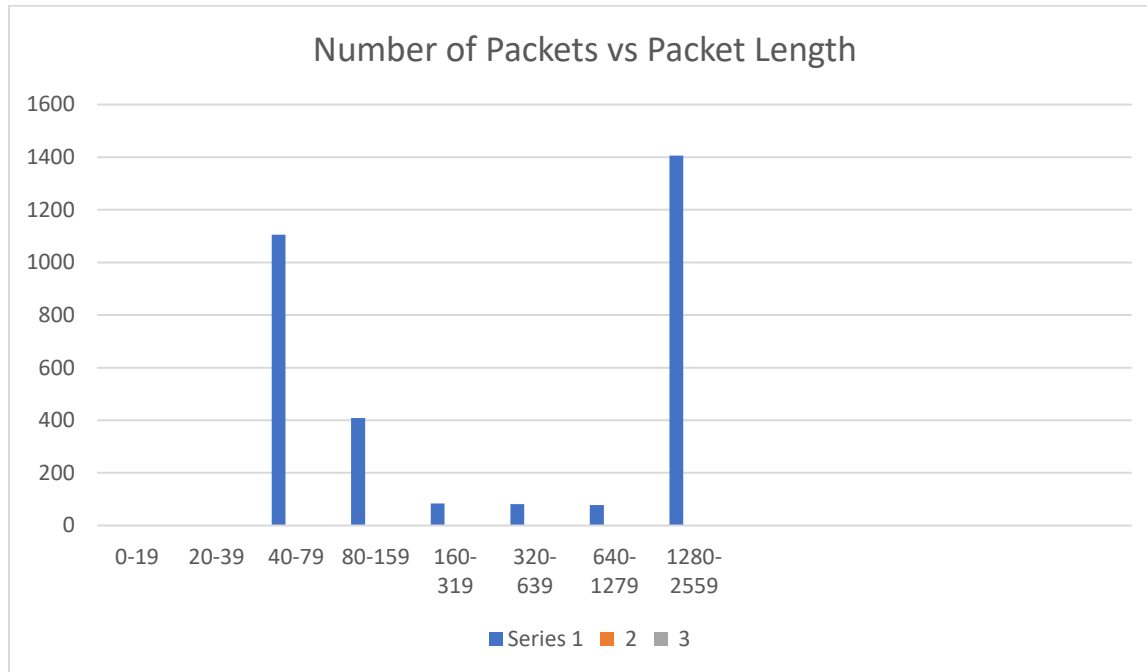
### TCP Flag packets

Components	# of Packets	Percent of total
Syn	157	21.9%
Push	525	73.1%
Reset	35	5%
Total	717	100%

From what I have noticed looking at both the SYN and PSH flags, a majority of both flags were sent from my computer while very little is actually received from YouTube.

## 1.3. Additional Requirements

### YouTube Packet Lengths



## References

<https://ipinfo.io/AS15169/130.211.0.0/16-130.211.14.0/23>

(This was used to show ip address I choose is from google)