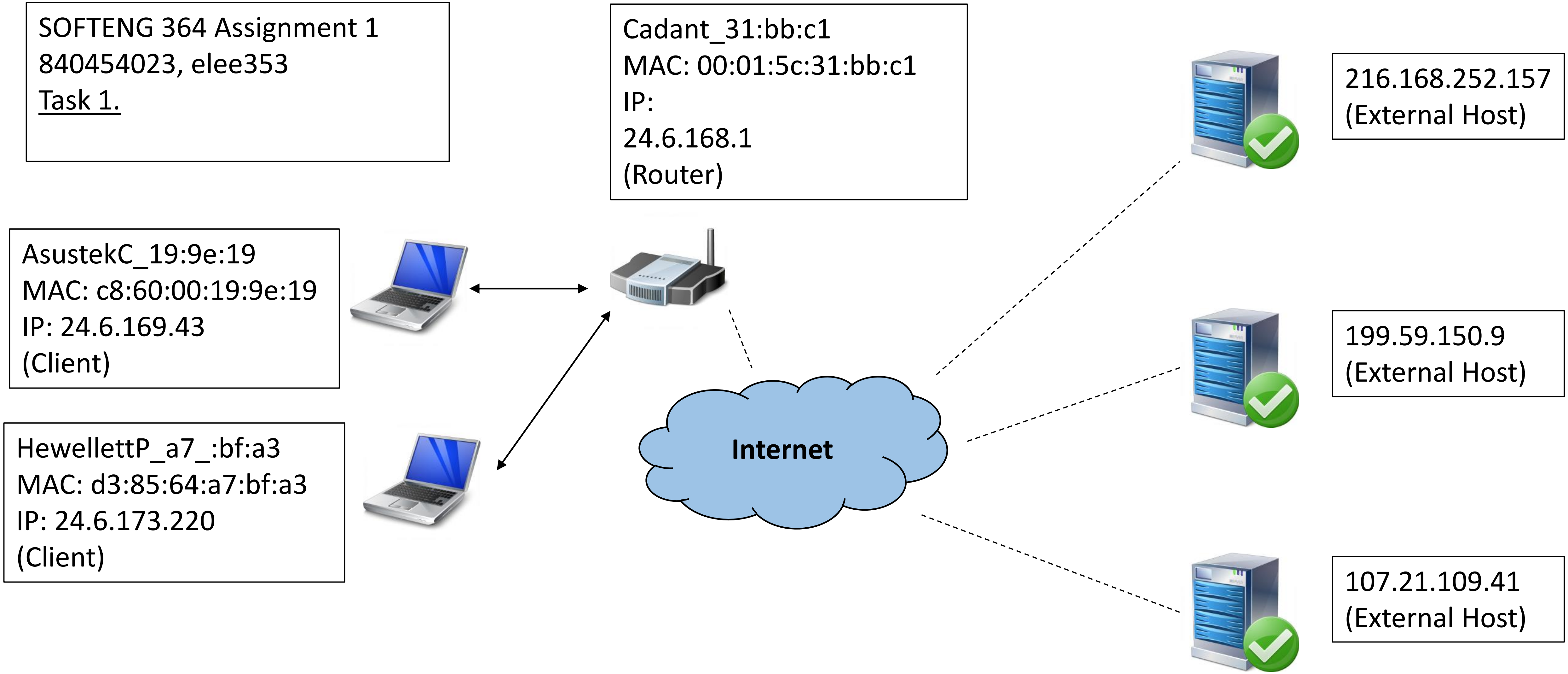
SOFTENG 364 Assignment 1
840454023, elee353
Task 1.



Cadant_31:bb:c1
MAC: 00:01:5c:31:bb:c1
IP:
24.6.168.1
(Router)

AsustekC_19:9e:19
MAC: c8:60:00:19:9e:19
IP: 24.6.169.43
(Client)

HewellettP_a7_:bf:a3
MAC: d3:85:64:a7:bf:a3
IP: 24.6.173.220
(Client)

Internet

216.168.252.157
(External Host)

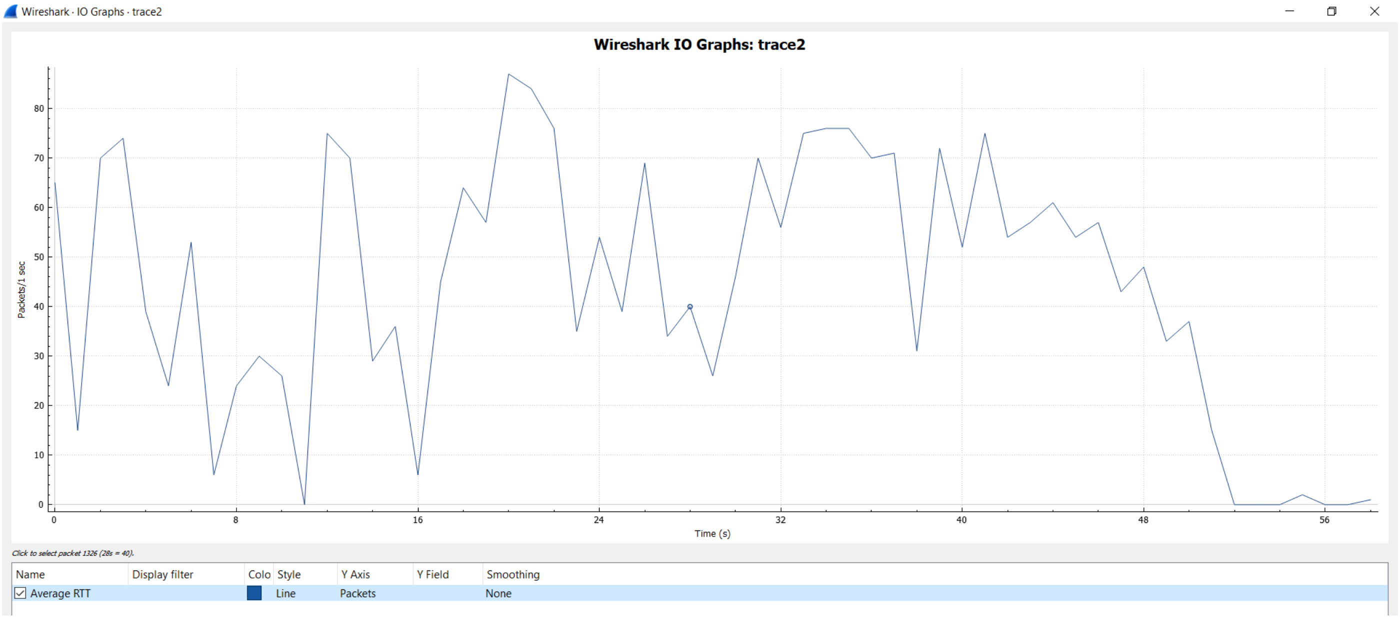
199.59.150.9
(External Host)

107.21.109.41
(External Host)

***All icons are from** [**www.iconexperience.com**](http://www.iconexperience.com)

SOFTENG 364 Assignment 1
840454023, elee353

Task 2. Q1.

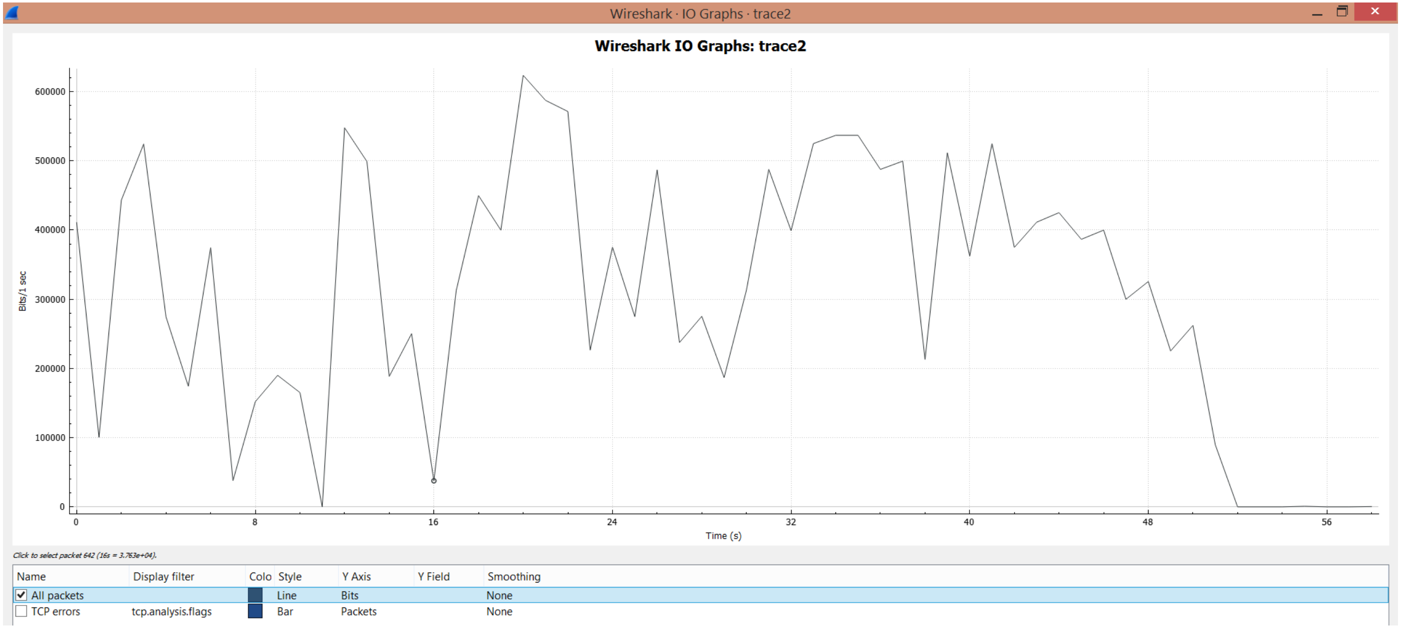


Click to select packet 1326 (28s = 40).

Name	Display filter	Colo	Style	Y Axis	Y Field	Smoothing
☑ Average RTT		■	Line	Packets		None

The highest packets-per-second value seen was 87 packets per second and it occurred in the 20th second.

Task 2. Q2.



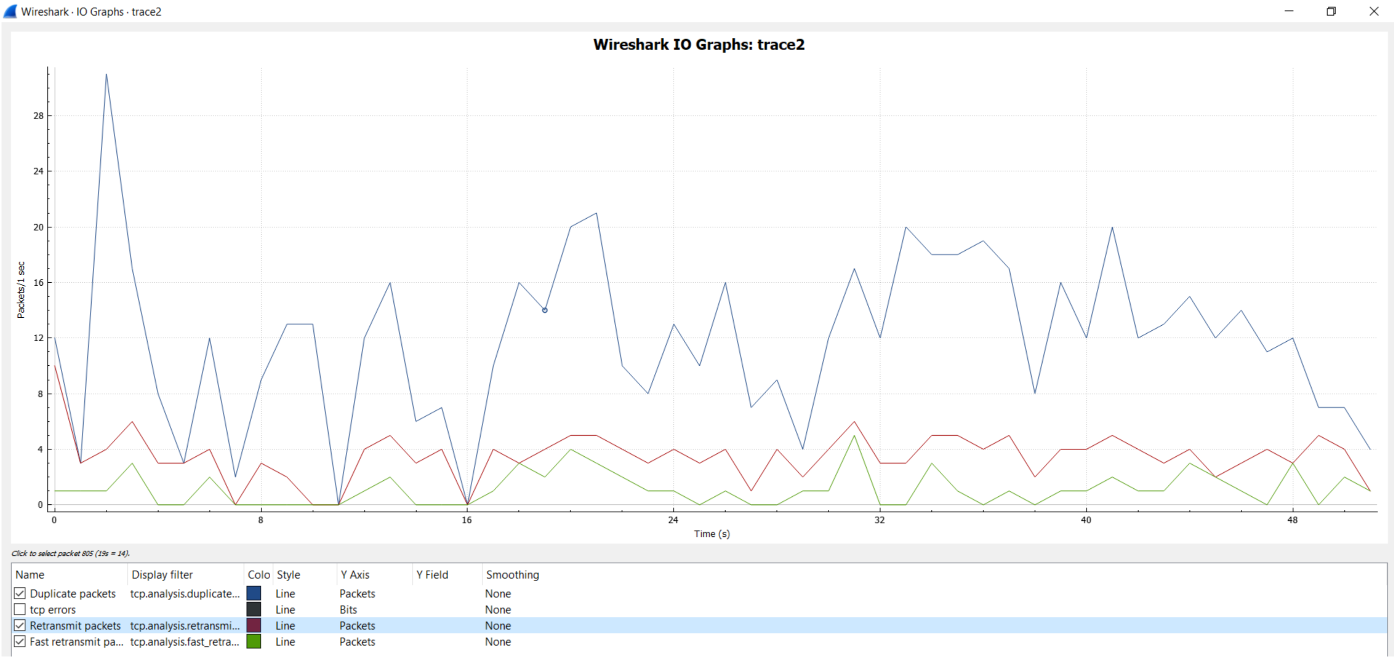
Click to select packet 642 (16s = 3.763e+04).

Name	Display filter	Colo	Style	Y Axis	Y Field	Smoothing
☑ All packets		■	Line	Bits		None
□ TCP errors	tcp.analysis.flags	■	Bar	Packets		None

The highest bits-per-second value is 623500 bits per second and it occurred in the 20th second.

Task 2. Q3.

- Apply filters: tcp.analysis.duplicate_ack, tcp.analysis.fast_retransmission, and tcp.analysis.retransmission.



Click to select packet 805 (19s = 14).

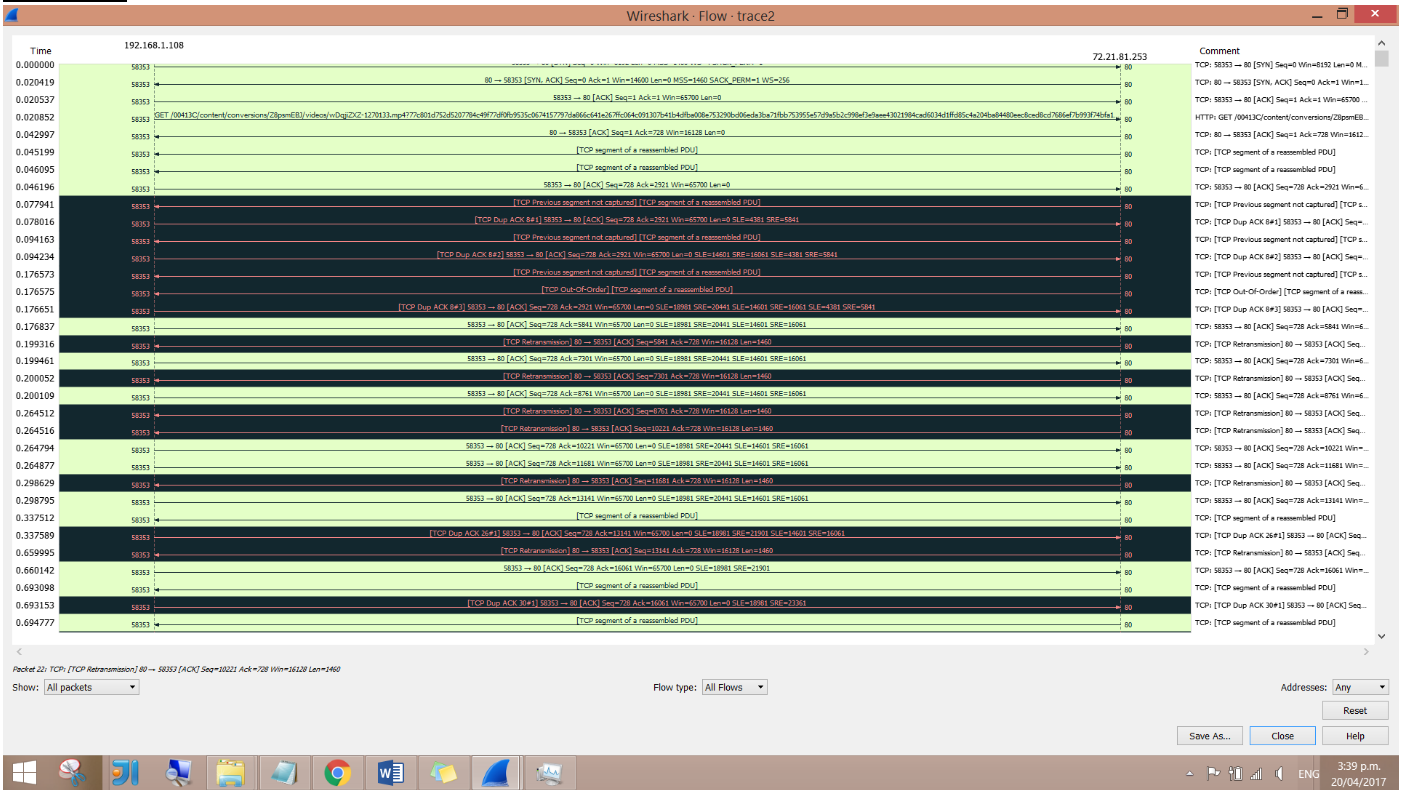
Name	Display filter	Colo	Style	Y Axis	Y Field	Smoothing
☑ Duplicate packets	tcp.analysis.duplicate...	🟦	Line	Packets		None
□ tcp errors		■	Line	Bits		None
☑ Retransmit packets	tcp.analysis.retransmi...	🟥	Line	Packets		None
☑ Fast retransmit pa...	tcp.analysis.fast_retra...	🟩	Line	Packets		None

Task2. Q4.

Duplicate ACKs are two or more acknowledgements containing exactly the same information. They are sent when the receiver sees a gap in the packets it receives. The gap can be caused by a lost segment or just a reordering of segments.

The sender will wait for a small number of duplicate ACKs to be received. If there is only one or two duplicate ACKs received before the reordered segment is processed, there is just a reordering of the segments and a new ACK will be generated. If three or more duplicate ACKs are received in a row, it is a strong indication that a segment has been lost. Consequently, fast retransmission will occur.

Task2. Q5.

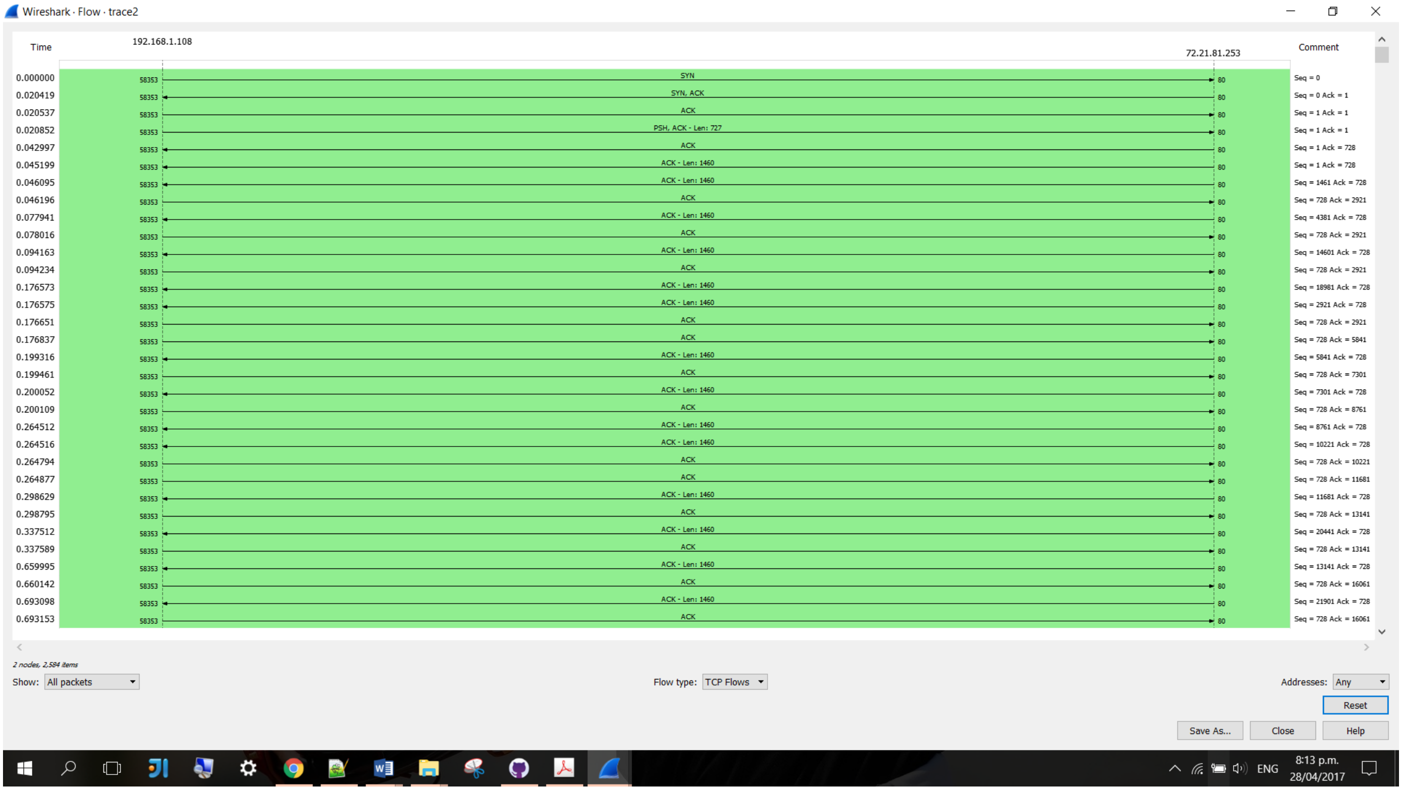


```
Time      192.168.1.108                                                                                          72.21.81.253   Comment
0.000000  58353  58353 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1                                80   TCP: 58353 → 80 [SYN] Seq=0 Win=8192 Len=0 M...
0.020419  58353  80 → 58353 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1460 SACK_PERM=1 WS=256                   80   TCP: 80 → 58353 [SYN, ACK] Seq=0 Ack=1 Win=1...
0.020537  58353  58353 → 80 [ACK] Seq=1 Ack=1 Win=65700 Len=0                                                    80   TCP: 58353 → 80 [ACK] Seq=1 Ack=1 Win=65700 ...
0.020852  58353  GET /00413C/content/conversions/Z8psmEBJ/videos/wDqjiZXZ-1270133.mp4?77c801d752d5207784c49f77df0fb9535c0674157797da866c641e267ffc064c091307b41b4dfba008e753290bd06eda3ba71fbb753955e57d9a5b2c998ef3e9aee43021984cad6034d1ffd85c4a204ba84480eec8ced8cd7686ef7b993f74bfa1...  80   HTTP: GET /00413C/content/conversions/Z8psmEB...
0.042997  58353  80 → 58353 [ACK] Seq=1 Ack=728 Win=16128 Len=0                                                  80   TCP: 80 → 58353 [ACK] Seq=1 Ack=728 Win=1612...
0.045199  58353  [TCP segment of a reassembled PDU]                                                              80   TCP: [TCP segment of a reassembled PDU]
0.046095  58353  [TCP segment of a reassembled PDU]                                                              80   TCP: [TCP segment of a reassembled PDU]
0.046196  58353  58353 → 80 [ACK] Seq=728 Ack=2921 Win=65700 Len=0                                               80   TCP: 58353 → 80 [ACK] Seq=728 Ack=2921 Win=6...
0.077941  58353  [TCP Previous segment not captured] [TCP segment of a reassembled PDU]                          80   TCP: [TCP Previous segment not captured] [TCP s...
0.078016  58353  [TCP Dup ACK 8#1] 58353 → 80 [ACK] Seq=728 Ack=2921 Win=65700 Len=0 SLE=4381 SRE=5841           80   TCP: [TCP Dup ACK 8#1] 58353 → 80 [ACK] Seq=...
0.094163  58353  [TCP Previous segment not captured] [TCP segment of a reassembled PDU]                          80   TCP: [TCP Previous segment not captured] [TCP s...
0.094234  58353  [TCP Dup ACK 8#2] 58353 → 80 [ACK] Seq=728 Ack=2921 Win=65700 Len=0 SLE=14601 SRE=16061 SLE=4381 SRE=5841   80   TCP: [TCP Dup ACK 8#2] 58353 → 80 [ACK] Seq=...
0.176573  58353  [TCP Previous segment not captured] [TCP segment of a reassembled PDU]                          80   TCP: [TCP Previous segment not captured] [TCP s...
0.176575  58353  [TCP Out-Of-Order] [TCP segment of a reassembled PDU]                                           80   TCP: [TCP Out-Of-Order] [TCP segment of a reass...
0.176651  58353  [TCP Dup ACK 8#3] 58353 → 80 [ACK] Seq=728 Ack=2921 Win=65700 Len=0 SLE=18981 SRE=20441 SLE=14601 SRE=16061 SLE=4381 SRE=5841   80   TCP: [TCP Dup ACK 8#3] 58353 → 80 [ACK] Seq=...
0.176837  58353  58353 → 80 [ACK] Seq=728 Ack=5841 Win=65700 Len=0 SLE=18981 SRE=20441 SLE=14601 SRE=16061      80   TCP: 58353 → 80 [ACK] Seq=728 Ack=5841 Win=6...
0.199316  58353  [TCP Retransmission] 80 → 58353 [ACK] Seq=5841 Ack=728 Win=16128 Len=1460                      80   TCP: [TCP Retransmission] 80 → 58353 [ACK] Seq...
0.199461  58353  58353 → 80 [ACK] Seq=728 Ack=7301 Win=65700 Len=0 SLE=18981 SRE=20441 SLE=14601 SRE=16061      80   TCP: 58353 → 80 [ACK] Seq=728 Ack=7301 Win=6...
0.200052  58353  [TCP Retransmission] 80 → 58353 [ACK] Seq=7301 Ack=728 Win=16128 Len=1460                      80   TCP: [TCP Retransmission] 80 → 58353 [ACK] Seq...
0.200109  58353  58353 → 80 [ACK] Seq=728 Ack=8761 Win=65700 Len=0 SLE=18981 SRE=20441 SLE=14601 SRE=16061      80   TCP: 58353 → 80 [ACK] Seq=728 Ack=8761 Win=6...
0.264512  58353  [TCP Retransmission] 80 → 58353 [ACK] Seq=8761 Ack=728 Win=16128 Len=1460                      80   TCP: [TCP Retransmission] 80 → 58353 [ACK] Seq...
0.264516  58353  [TCP Retransmission] 80 → 58353 [ACK] Seq=10221 Ack=728 Win=16128 Len=1460                     80   TCP: [TCP Retransmission] 80 → 58353 [ACK] Seq...
0.264794  58353  58353 → 80 [ACK] Seq=728 Ack=10221 Win=65700 Len=0 SLE=18981 SRE=20441 SLE=14601 SRE=16061     80   TCP: 58353 → 80 [ACK] Seq=728 Ack=10221 Win=...
0.264877  58353  58353 → 80 [ACK] Seq=728 Ack=11681 Win=65700 Len=0 SLE=18981 SRE=20441 SLE=14601 SRE=16061     80   TCP: 58353 → 80 [ACK] Seq=728 Ack=11681 Win=...
0.298629  58353  [TCP Retransmission] 80 → 58353 [ACK] Seq=11681 Ack=728 Win=16128 Len=1460                     80   TCP: [TCP Retransmission] 80 → 58353 [ACK] Seq...
0.298795  58353  58353 → 80 [ACK] Seq=728 Ack=13141 Win=65700 Len=0 SLE=18981 SRE=20441 SLE=14601 SRE=16061     80   TCP: 58353 → 80 [ACK] Seq=728 Ack=13141 Win=...
0.337512  58353  [TCP segment of a reassembled PDU]                                                              80   TCP: [TCP segment of a reassembled PDU]
0.337589  58353  [TCP Dup ACK 26#1] 58353 → 80 [ACK] Seq=728 Ack=13141 Win=65700 Len=0 SLE=18981 SRE=21901 SLE=14601 SRE=16061   80   TCP: [TCP Dup ACK 26#1] 58353 → 80 [ACK] Seq...
0.659995  58353  [TCP Retransmission] 80 → 58353 [ACK] Seq=13141 Ack=728 Win=16128 Len=1460                     80   TCP: [TCP Retransmission] 80 → 58353 [ACK] Seq...
0.660142  58353  58353 → 80 [ACK] Seq=728 Ack=16061 Win=65700 Len=0 SLE=18981 SRE=21901                         80   TCP: 58353 → 80 [ACK] Seq=728 Ack=16061 Win=...
0.693098  58353  [TCP segment of a reassembled PDU]                                                              80   TCP: [TCP segment of a reassembled PDU]
0.693153  58353  [TCP Dup ACK 30#1] 58353 → 80 [ACK] Seq=728 Ack=16061 Win=65700 Len=0 SLE=18981 SRE=23361      80   TCP: [TCP Dup ACK 30#1] 58353 → 80 [ACK] Seq...
0.694777  58353  [TCP segment of a reassembled PDU]                                                              80   TCP: [TCP segment of a reassembled PDU]
```

Packet 22: TCP: [TCP Retransmission] 80 → 58353 [ACK] Seq=10221 Ack=728 Win=16128 Len=1460

With only TCP flows selected:

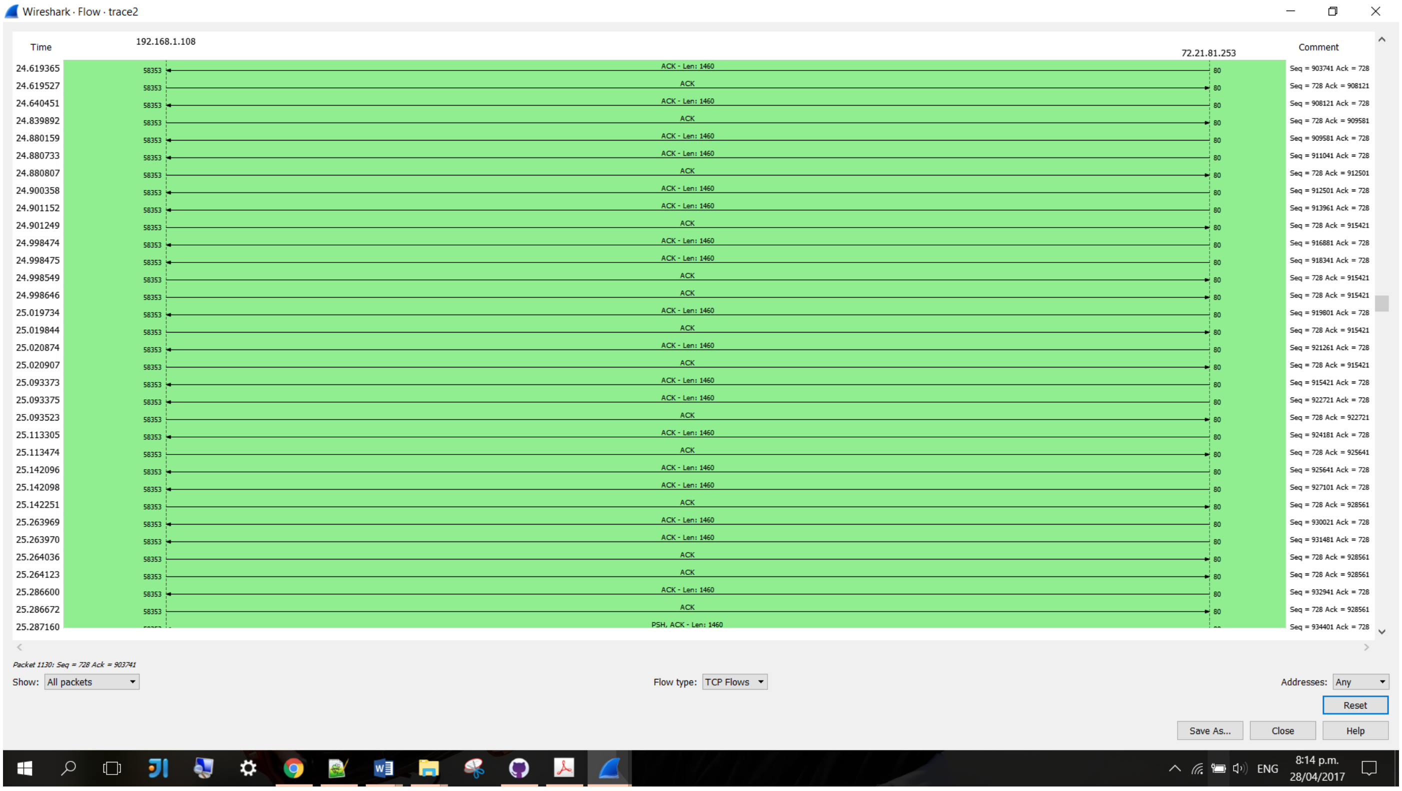
- First 30:



```
Time      192.168.1.108                                72.21.81.253   Comment
0.000000  58353  SYN                                   80   Seq = 0
0.020419  58353  SYN, ACK                              80   Seq = 0 Ack = 1
0.020537  58353  ACK                                   80   Seq = 1 Ack = 1
0.020852  58353  PSH, ACK - Len: 727                   80   Seq = 1 Ack = 1
0.042997  58353  ACK                                   80   Seq = 1 Ack = 728
0.045199  58353  ACK - Len: 1460                       80   Seq = 1 Ack = 728
0.046095  58353  ACK - Len: 1460                       80   Seq = 1461 Ack = 728
0.046196  58353  ACK                                   80   Seq = 728 Ack = 2921
0.077941  58353  ACK - Len: 1460                       80   Seq = 4381 Ack = 728
0.078016  58353  ACK                                   80   Seq = 728 Ack = 2921
0.094163  58353  ACK - Len: 1460                       80   Seq = 14601 Ack = 728
0.094234  58353  ACK                                   80   Seq = 728 Ack = 2921
0.176573  58353  ACK - Len: 1460                       80   Seq = 18981 Ack = 728
0.176575  58353  ACK - Len: 1460                       80   Seq = 2921 Ack = 728
0.176651  58353  ACK                                   80   Seq = 728 Ack = 2921
0.176837  58353  ACK                                   80   Seq = 728 Ack = 5841
0.199316  58353  ACK - Len: 1460                       80   Seq = 5841 Ack = 728
0.199461  58353  ACK                                   80   Seq = 728 Ack = 7301
0.200052  58353  ACK - Len: 1460                       80   Seq = 7301 Ack = 728
0.200109  58353  ACK                                   80   Seq = 728 Ack = 8761
0.264512  58353  ACK - Len: 1460                       80   Seq = 8761 Ack = 728
0.264516  58353  ACK - Len: 1460                       80   Seq = 10221 Ack = 728
0.264794  58353  ACK                                   80   Seq = 728 Ack = 10221
0.264877  58353  ACK                                   80   Seq = 728 Ack = 11681
0.298629  58353  ACK - Len: 1460                       80   Seq = 11681 Ack = 728
0.298795  58353  ACK                                   80   Seq = 728 Ack = 13141
0.337512  58353  ACK - Len: 1460                       80   Seq = 20441 Ack = 728
0.337589  58353  ACK                                   80   Seq = 728 Ack = 13141
0.659995  58353  ACK - Len: 1460                       80   Seq = 13141 Ack = 728
0.660142  58353  ACK                                   80   Seq = 728 Ack = 16061
0.693098  58353  ACK - Len: 1460                       80   Seq = 21901 Ack = 728
0.693153  58353  ACK                                   80   Seq = 728 Ack = 16061
```

2 nodes, 2,584 items

- Middle:



Time	192.168.1.108		72.21.81.253	Comment
24.619365	58353	← ACK - Len: 1460	80	Seq = 903741 Ack = 728
24.619527	58353	ACK →	80	Seq = 728 Ack = 908121
24.640451	58353	← ACK - Len: 1460	80	Seq = 908121 Ack = 728
24.839892	58353	ACK →	80	Seq = 728 Ack = 909581
24.880159	58353	← ACK - Len: 1460	80	Seq = 909581 Ack = 728
24.880733	58353	← ACK - Len: 1460	80	Seq = 911041 Ack = 728
24.880807	58353	ACK →	80	Seq = 728 Ack = 912501
24.900358	58353	← ACK - Len: 1460	80	Seq = 912501 Ack = 728
24.901152	58353	← ACK - Len: 1460	80	Seq = 913961 Ack = 728
24.901249	58353	ACK →	80	Seq = 728 Ack = 915421
24.998474	58353	← ACK - Len: 1460	80	Seq = 916881 Ack = 728
24.998475	58353	← ACK - Len: 1460	80	Seq = 918341 Ack = 728
24.998549	58353	ACK →	80	Seq = 728 Ack = 915421
24.998646	58353	ACK →	80	Seq = 728 Ack = 915421
25.019734	58353	← ACK - Len: 1460	80	Seq = 919801 Ack = 728
25.019844	58353	ACK →	80	Seq = 728 Ack = 915421
25.020874	58353	← ACK - Len: 1460	80	Seq = 921261 Ack = 728
25.020907	58353	ACK →	80	Seq = 728 Ack = 915421
25.093373	58353	← ACK - Len: 1460	80	Seq = 915421 Ack = 728
25.093375	58353	← ACK - Len: 1460	80	Seq = 922721 Ack = 728
25.093523	58353	ACK →	80	Seq = 728 Ack = 922721
25.113305	58353	← ACK - Len: 1460	80	Seq = 924181 Ack = 728
25.113474	58353	ACK →	80	Seq = 728 Ack = 925641
25.142096	58353	← ACK - Len: 1460	80	Seq = 925641 Ack = 728
25.142098	58353	← ACK - Len: 1460	80	Seq = 927101 Ack = 728
25.142251	58353	ACK →	80	Seq = 728 Ack = 928561
25.263969	58353	← ACK - Len: 1460	80	Seq = 930021 Ack = 728
25.263970	58353	← ACK - Len: 1460	80	Seq = 931481 Ack = 728
25.264036	58353	ACK →	80	Seq = 728 Ack = 928561
25.264123	58353	ACK →	80	Seq = 728 Ack = 928561
25.286600	58353	← ACK - Len: 1460	80	Seq = 932941 Ack = 728
25.286672	58353	ACK →	80	Seq = 728 Ack = 928561
25.287160		PSH, ACK - Len: 1460		Seq = 934401 Ack = 728

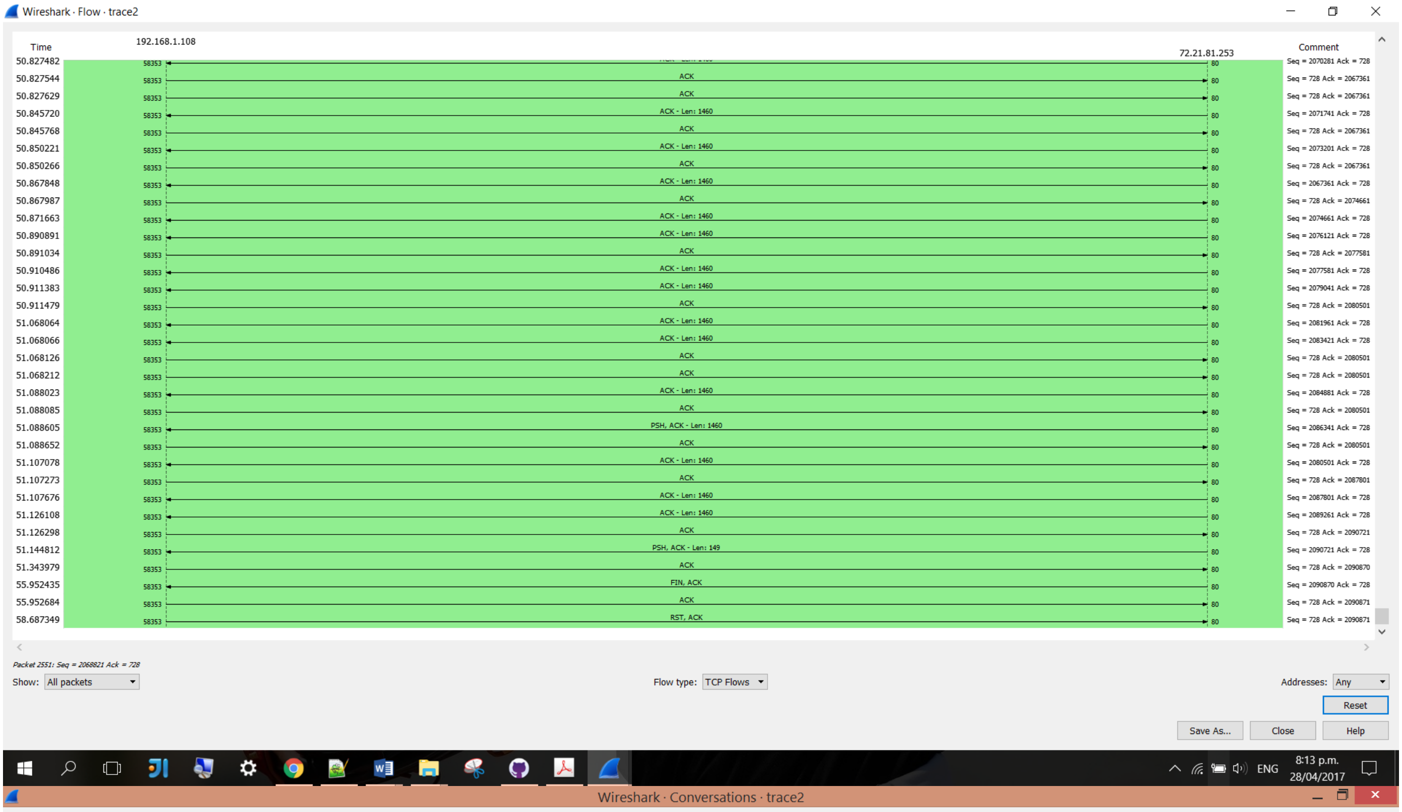
Packet 1130: Seq = 728 Ack = 903741

Show: All packets

Flow type: TCP Flows

Addresses: Any

- Last 30:



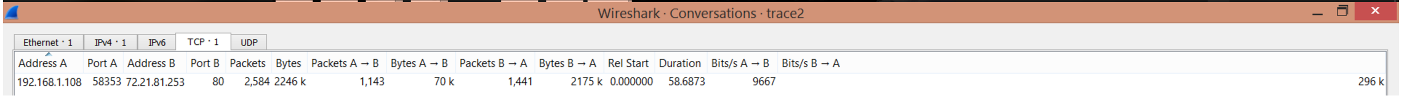
Time	192.168.1.108		72.21.81.253	Comment
50.827482	58353	← ACK - Len: 1460	80	Seq = 2070281 Ack = 728
50.827544	58353	ACK →	80	Seq = 728 Ack = 2067361
50.827629	58353	ACK →	80	Seq = 728 Ack = 2067361
50.845720	58353	← ACK - Len: 1460	80	Seq = 2071741 Ack = 728
50.845768	58353	ACK →	80	Seq = 728 Ack = 2067361
50.850221	58353	← ACK - Len: 1460	80	Seq = 2073201 Ack = 728
50.850266	58353	ACK →	80	Seq = 728 Ack = 2067361
50.867848	58353	← ACK - Len: 1460	80	Seq = 2067361 Ack = 728
50.867987	58353	ACK →	80	Seq = 728 Ack = 2074661
50.871663	58353	← ACK - Len: 1460	80	Seq = 2074661 Ack = 728
50.890891	58353	← ACK - Len: 1460	80	Seq = 2076121 Ack = 728
50.891034	58353	ACK →	80	Seq = 728 Ack = 2077581
50.910486	58353	← ACK - Len: 1460	80	Seq = 2077581 Ack = 728
50.911383	58353	← ACK - Len: 1460	80	Seq = 2079041 Ack = 728
50.911479	58353	ACK →	80	Seq = 728 Ack = 2080501
51.068064	58353	← ACK - Len: 1460	80	Seq = 2081961 Ack = 728
51.068066	58353	← ACK - Len: 1460	80	Seq = 2083421 Ack = 728
51.068126	58353	ACK →	80	Seq = 728 Ack = 2080501
51.068212	58353	ACK →	80	Seq = 728 Ack = 2080501
51.088023	58353	← ACK - Len: 1460	80	Seq = 2084881 Ack = 728
51.088085	58353	ACK →	80	Seq = 728 Ack = 2080501
51.088605	58353	← PSH, ACK - Len: 1460	80	Seq = 2086341 Ack = 728
51.088652	58353	ACK →	80	Seq = 728 Ack = 2080501
51.107078	58353	← ACK - Len: 1460	80	Seq = 2080501 Ack = 728
51.107273	58353	ACK →	80	Seq = 728 Ack = 2087801
51.107676	58353	← ACK - Len: 1460	80	Seq = 2087801 Ack = 728
51.126108	58353	← ACK - Len: 1460	80	Seq = 2089261 Ack = 728
51.126298	58353	ACK →	80	Seq = 728 Ack = 2090721
51.144812	58353	← PSH, ACK - Len: 149	80	Seq = 2090721 Ack = 728
51.343979	58353	ACK →	80	Seq = 728 Ack = 2090870
55.952435	58353	← FIN, ACK	80	Seq = 2090870 Ack = 728
55.952684	58353	ACK →	80	Seq = 728 Ack = 2090871
58.687349	58353	RST, ACK →	80	Seq = 728 Ack = 2090871

Packet 2551: Seq = 2068821 Ack = 728

Show: All packets

Flow type: TCP Flows

Addresses: Any



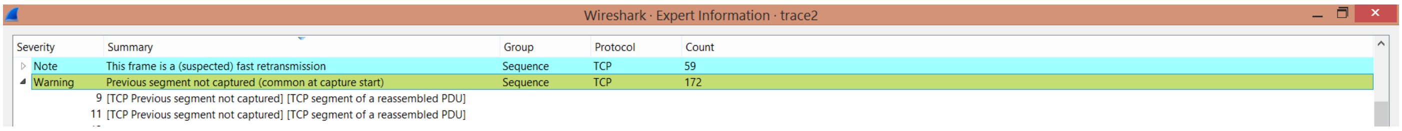
Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
192.168.1.108	58353	72.21.81.253	80	2,584	2246 k	1,143	70 k	1,441	2175 k	0.000000	58.6873	9667	296 k

There is one TCP conversation between the two nodes.

<u>Task2. Q6.</u>

"Tcp previous segment not captured" means Wireshark did not see a packet that should have been in the trace; this means no packet from the same TCP session whose sequence number + byte length matches the sequence number of a present packet.

This is caused by either the packet really wasn't seen on the wire due to a packet loss, or Wireshark did not record the packet fast and timely enough even though it had been on the wire.

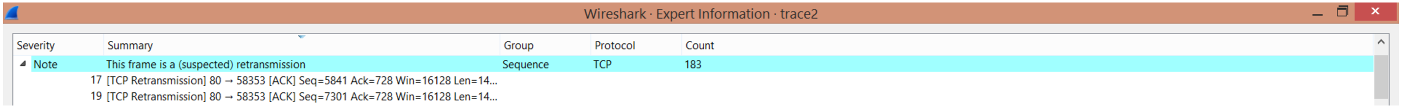


Wireshark · Expert Information · trace2

Severity	Summary	Group	Protocol	Count
Note	This frame is a (suspected) fast retransmission	Sequence	TCP	59
Warning	Previous segment not captured (common at capture start)	Sequence	TCP	172
	9 [TCP Previous segment not captured] [TCP segment of a reassembled PDU]			
	11 [TCP Previous segment not captured] [TCP segment of a reassembled PDU]			

There is 172 times "previous segment not captured" has been detected.

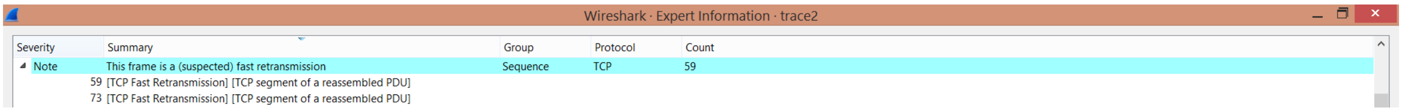
Task 2. Q7.



Wireshark · Expert Information · trace2

Severity	Summary	Group	Protocol	Count
Note	This frame is a (suspected) retransmission	Sequence	TCP	183
	17 [TCP Retransmission] 80 → 58353 [ACK] Seq=5841 Ack=728 Win=16128 Len=14...			
	19 [TCP Retransmission] 80 → 58353 [ACK] Seq=7301 Ack=728 Win=16128 Len=14...			

There are 183 retransmissions in this trace file.

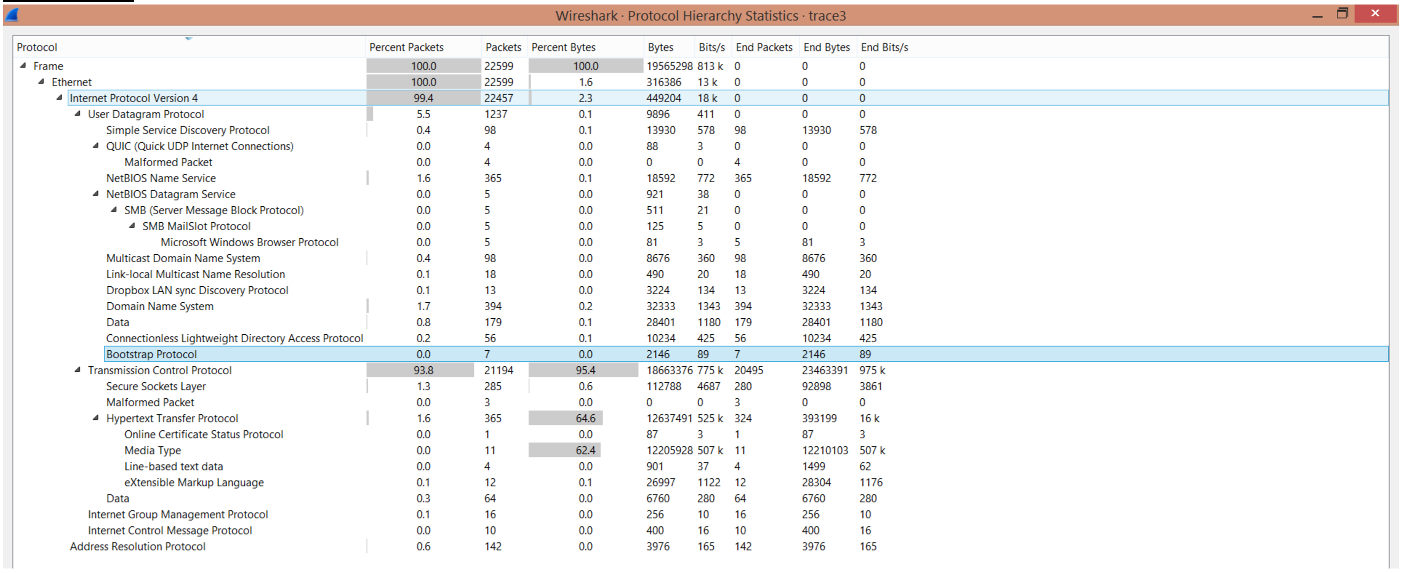


Wireshark · Expert Information · trace2

Severity	Summary	Group	Protocol	Count
Note	This frame is a (suspected) fast retransmission	Sequence	TCP	59
	59 [TCP Fast Retransmission] [TCP segment of a reassembled PDU]			
	73 [TCP Fast Retransmission] [TCP segment of a reassembled PDU]			

There are 59 fast transmissions in this trace file.

Task 3. Q1.



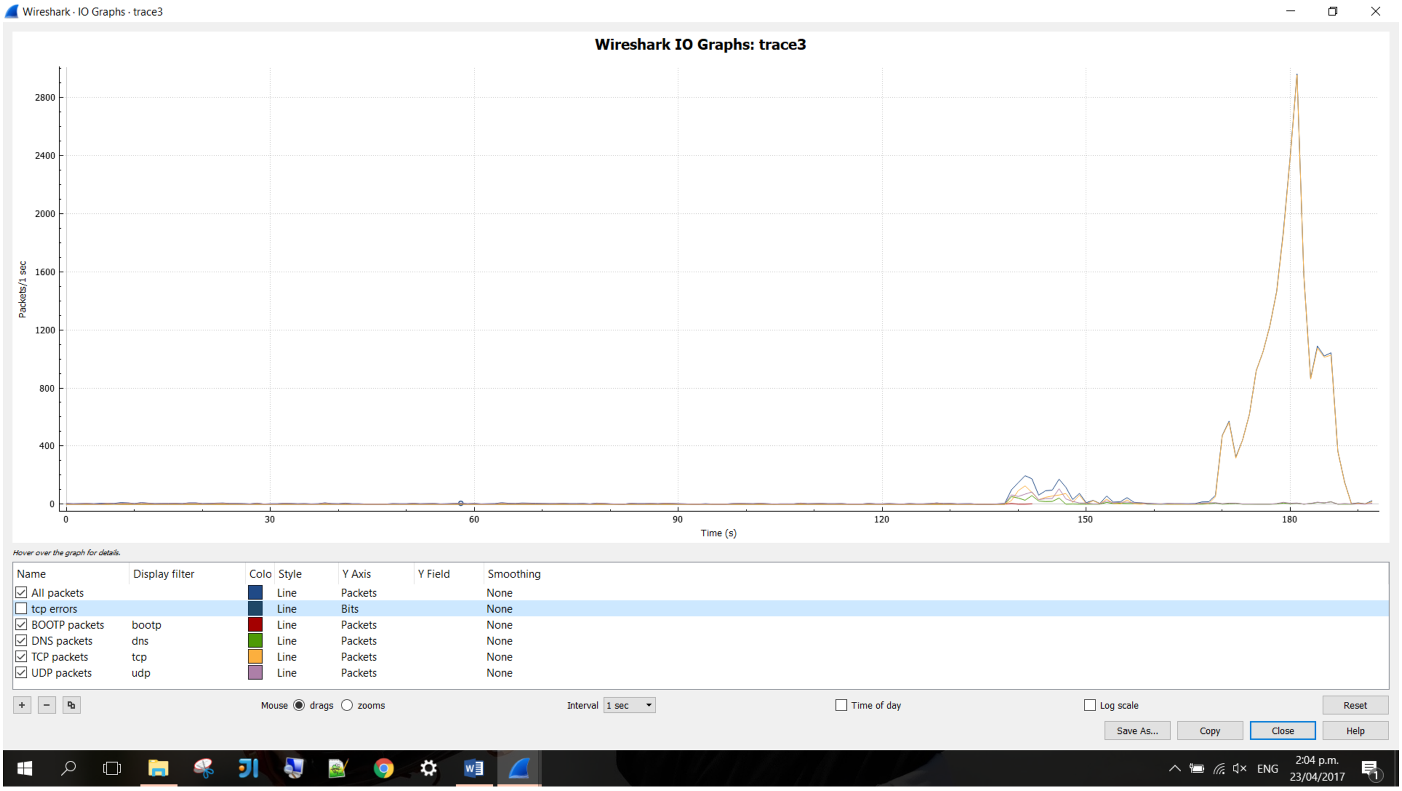
Wireshark · Protocol Hierarchy Statistics · trace3

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
Frame	100.0	22599	100.0	19565298	813 k	0	0	0
Ethernet	100.0	22599	1.6	316386	13 k	0	0	0
Internet Protocol Version 4	99.4	22457	2.3	449204	18 k	0	0	0
User Datagram Protocol	5.5	1237	0.1	9896	411	0	0	0
Simple Service Discovery Protocol	0.4	98	0.1	13930	578	98	13930	578
QUIC (Quick UDP Internet Connections)	0.0	4	0.0	88	3	0	0	0
Malformed Packet	0.0	4	0.0	0	0	4	0	0
NetBIOS Name Service	1.6	365	0.1	18592	772	365	18592	772
NetBIOS Datagram Service	0.0	5	0.0	921	38	0	0	0
SMB (Server Message Block Protocol)	0.0	5	0.0	511	21	0	0	0
SMB MailSlot Protocol	0.0	5	0.0	125	5	0	0	0
Microsoft Windows Browser Protocol	0.0	5	0.0	81	3	5	81	3
Multicast Domain Name System	0.4	98	0.0	8676	360	98	8676	360
Link-local Multicast Name Resolution	0.1	18	0.0	490	20	18	490	20
Dropbox LAN sync Discovery Protocol	0.1	13	0.0	3224	134	13	3224	134
Domain Name System	1.7	394	0.2	32333	1343	394	32333	1343
Data	0.8	179	0.1	28401	1180	179	28401	1180
Connectionless Lightweight Directory Access Protocol	0.2	56	0.1	10234	425	56	10234	425
Bootstrap Protocol	0.0	7	0.0	2146	89	7	2146	89
Transmission Control Protocol	93.8	21194	95.4	18663376	775 k	20495	23463391	975 k
Secure Sockets Layer	1.3	285	0.6	112788	4687	280	92898	3861
Malformed Packet	0.0	3	0.0	0	0	3	0	0
Hypertext Transfer Protocol	1.6	365	64.6	12637491	525 k	324	393199	16 k
Online Certificate Status Protocol	0.0	1	0.0	87	3	1	87	3
Media Type	0.0	11	62.4	12205928	507 k	11	12210103	507 k
Line-based text data	0.0	4	0.0	901	37	4	1499	62
eXtensible Markup Language	0.1	12	0.1	26997	1122	12	28304	1176
Data	0.3	64	0.0	6760	280	64	6760	280
Internet Group Management Protocol	0.1	16	0.0	256	10	16	256	10
Internet Control Message Protocol	0.0	10	0.0	400	16	10	400	16
Address Resolution Protocol	0.6	142	0.0	3976	165	142	3976	165

- The protocols include: ARP, BROWSER, CLDAP, DB-LSP-DISC, DHCP, DNS, HTTP, HTTP/XML, ICMP, IGMPv2, IGMPv3, LLMNR, MDNS, NBNS, OCSP, QUIC, SSDP, TCP, TLSv1, TLSv1.2, UDP
- 0 percent of the total was involved with DCHP messages.
- TCP is used the most here.

Task 3. Q2.

- Apply bootp, dns, tcp, and udp filters.
- Altogether:

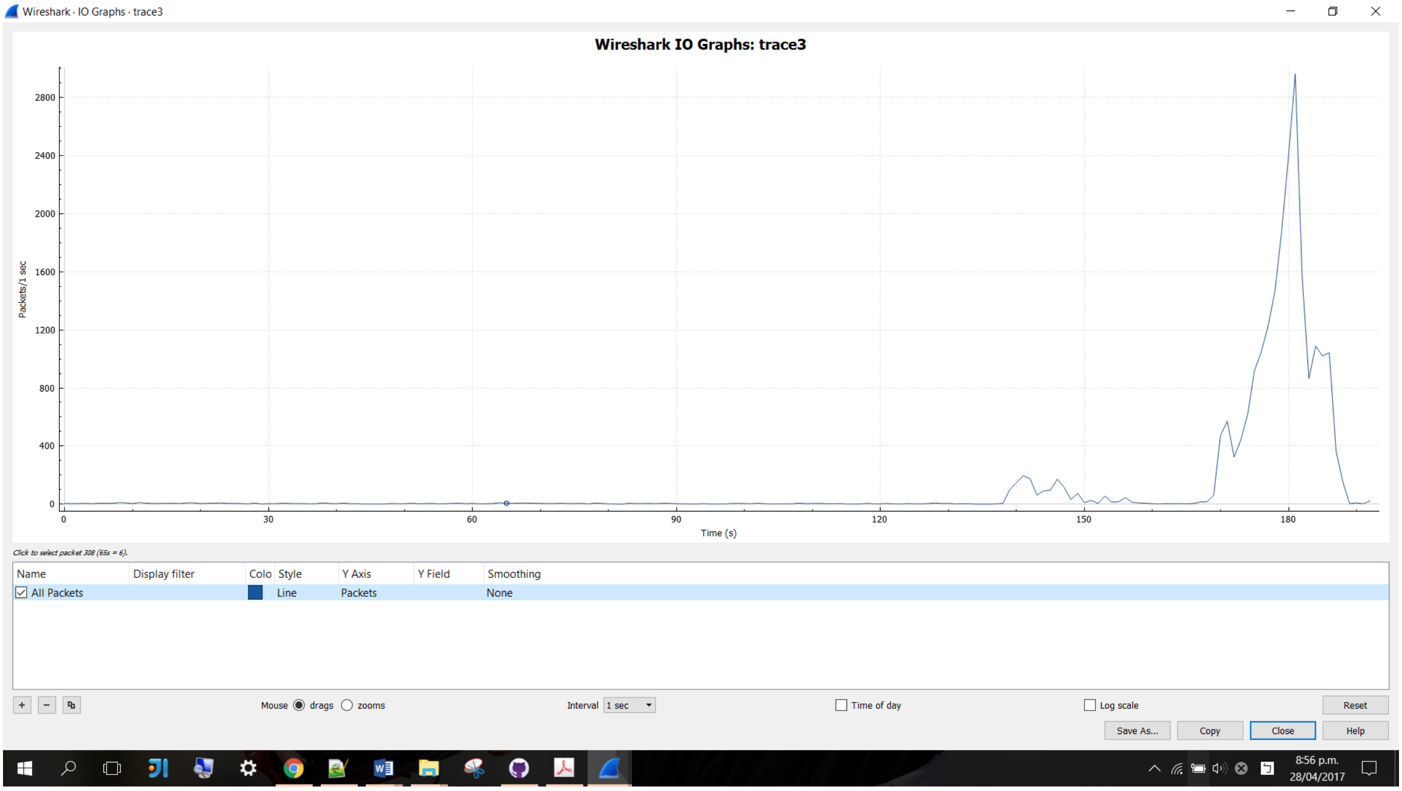


Hover over the graph for details.

Name	Display filter	Colo	Style	Y Axis	Y Field	Smoothing
☑ All packets		🟦	Line	Packets		None
□ tcp errors		🟦	Line	Bits		None
☑ BOOTP packets	bootp	🟥	Line	Packets		None
☑ DNS packets	dns	🟩	Line	Packets		None
☑ TCP packets	tcp	🟧	Line	Packets		None
☑ UDP packets	udp	🟪	Line	Packets		None

Mouse ● drags ○ zooms   Interval 1 sec   □ Time of day   □ Log scale

- All traffic:

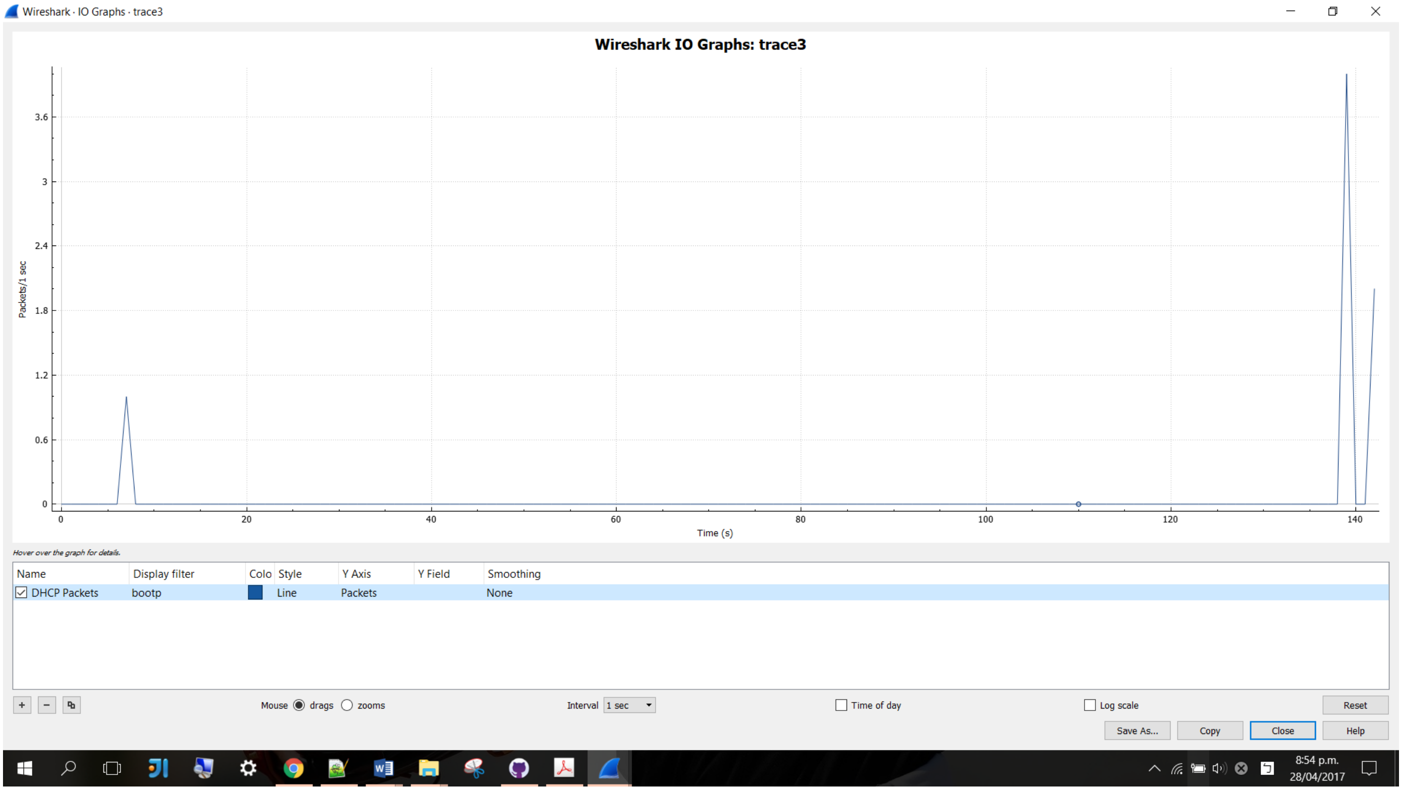


Click to select packet 308 (65s = 6).

Name	Display filter	Colo	Style	Y Axis	Y Field	Smoothing
☑ All Packets		🟦	Line	Packets		None

Mouse ● drags ○ zooms   Interval 1 sec   □ Time of day   □ Log scale

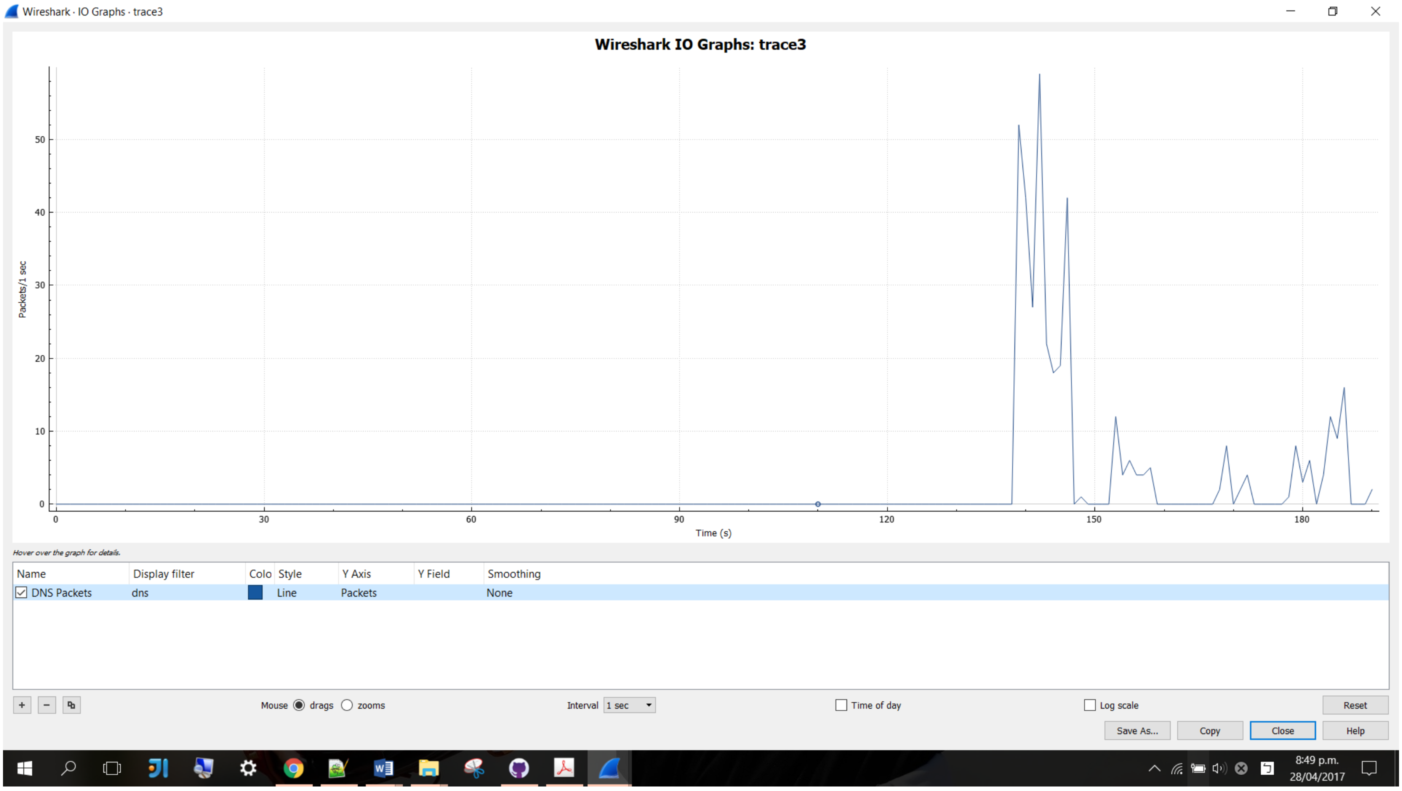
- DHCP filter:



Name	Display filter	Colo	Style	Y Axis	Y Field	Smoothing
☑ DHCP Packets	bootp	■	Line	Packets		None

Mouse ● drags ○ zooms Interval 1 sec □ Time of day □ Log scale

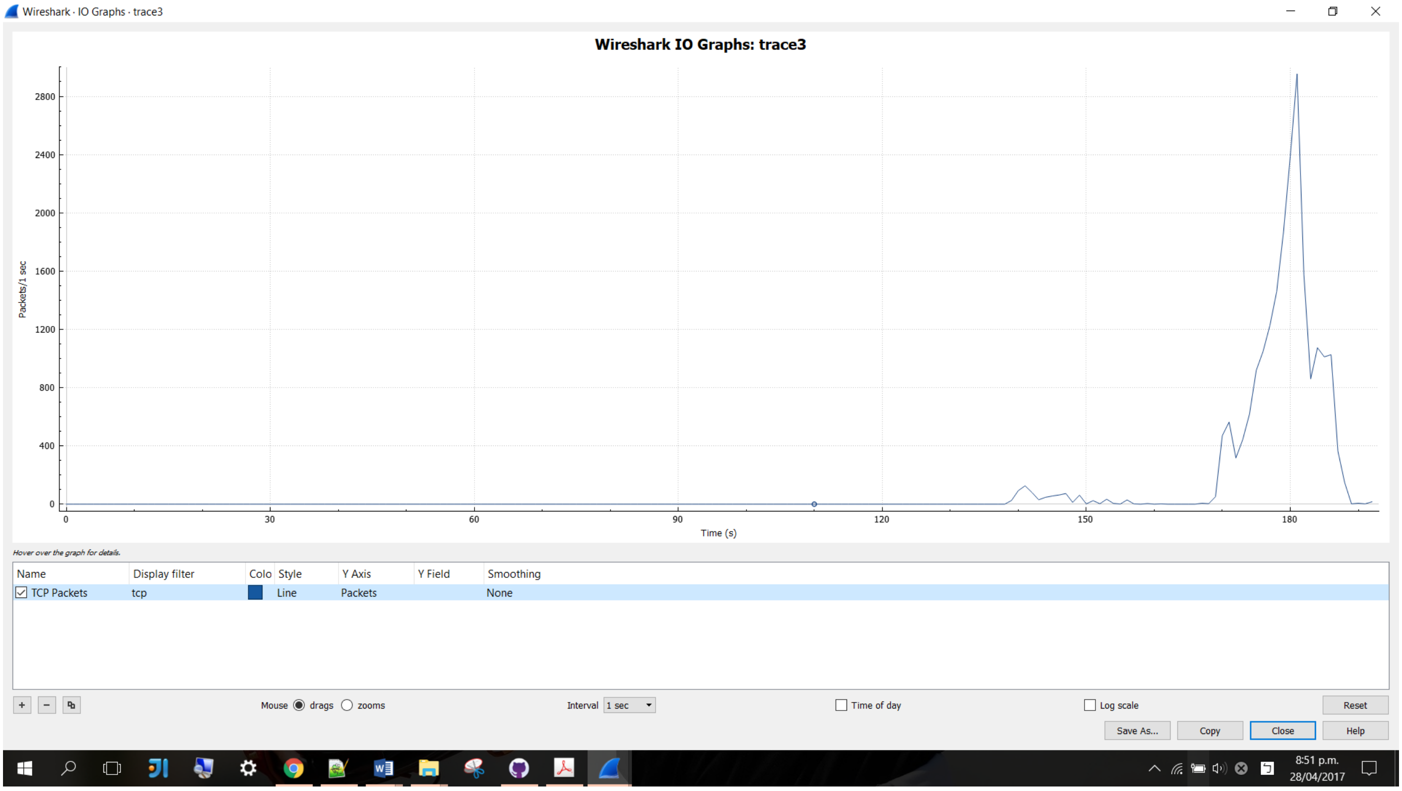
- DNS filter:



Name	Display filter	Colo	Style	Y Axis	Y Field	Smoothing
☑ DNS Packets	dns	■	Line	Packets		None

Mouse ● drags ○ zooms Interval 1 sec □ Time of day □ Log scale

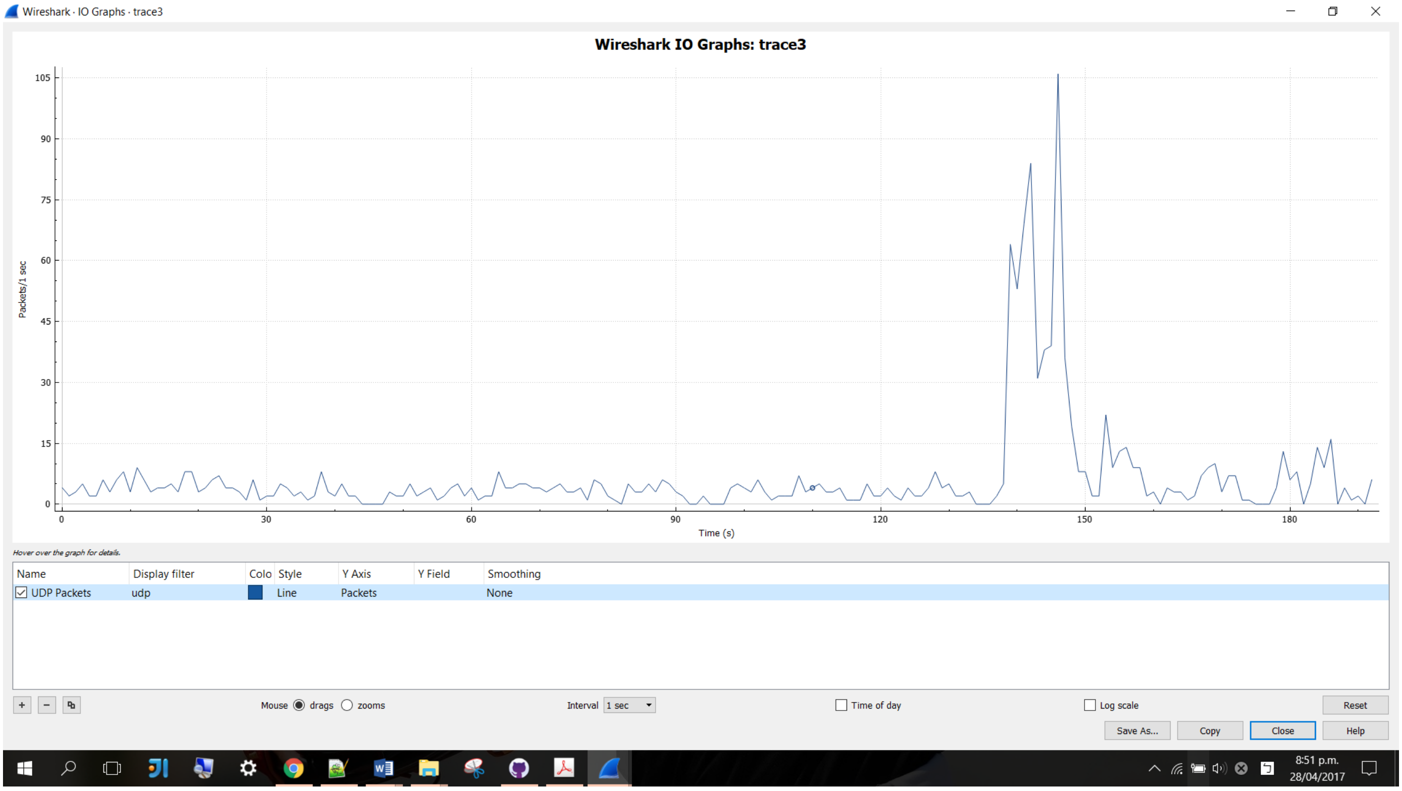
- TCP filter:



Name	Display filter	Colo	Style	Y Axis	Y Field	Smoothing
☑ TCP Packets	tcp	🟦	Line	Packets		None

Mouse ● drags ○ zooms   Interval 1 sec   □ Time of day   □ Log scale

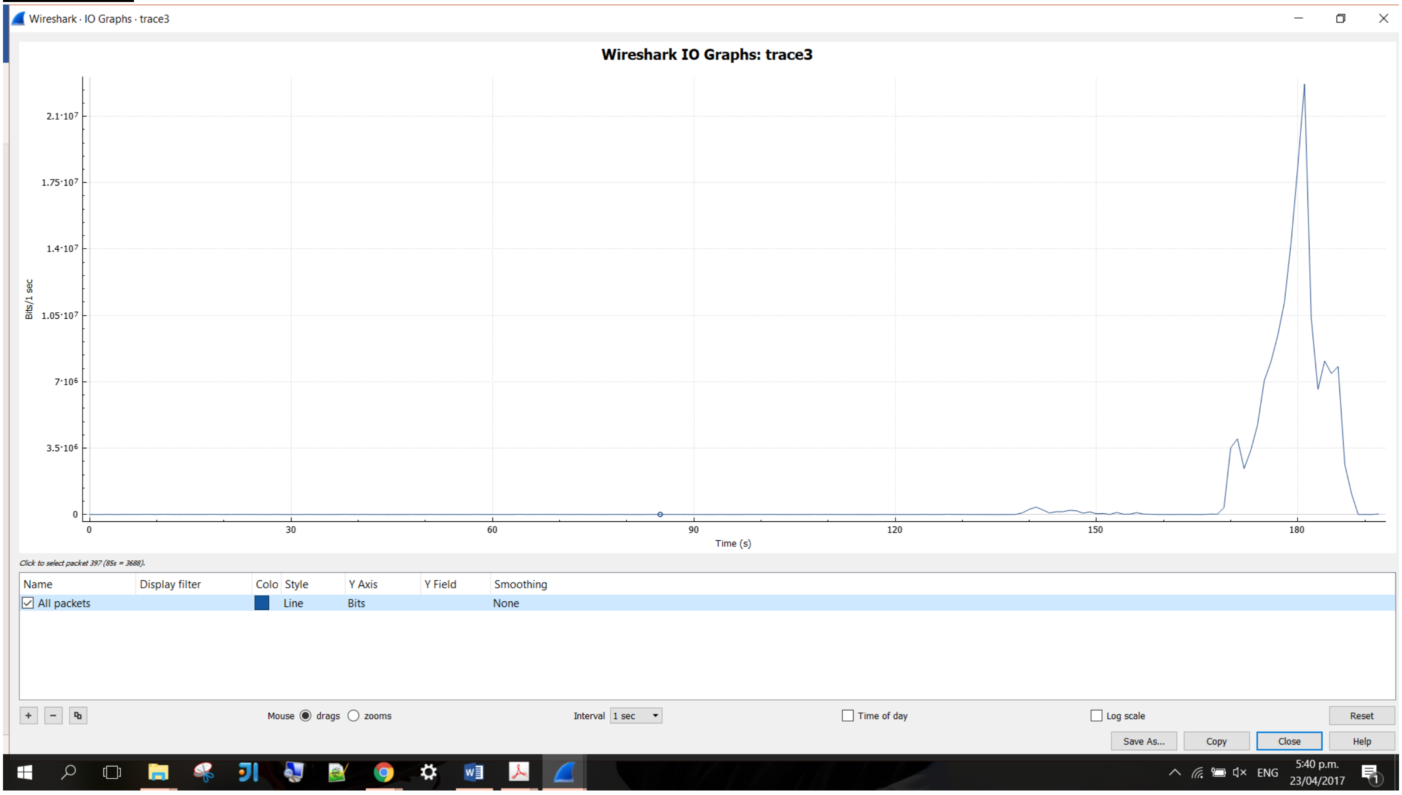
- UDP filter:



Name	Display filter	Colo	Style	Y Axis	Y Field	Smoothing
☑ UDP Packets	udp	🟦	Line	Packets		None

Mouse ● drags ○ zooms   Interval 1 sec   □ Time of day   □ Log scale

Task 3. Q3.



Click to select packet 397 (85s = 3688).

Name	Display filter	Colo	Style	Y Axis	Y Field	Smoothing
☑ All packets		■	Line	Bits		None

Mouse ● drags ○ zooms

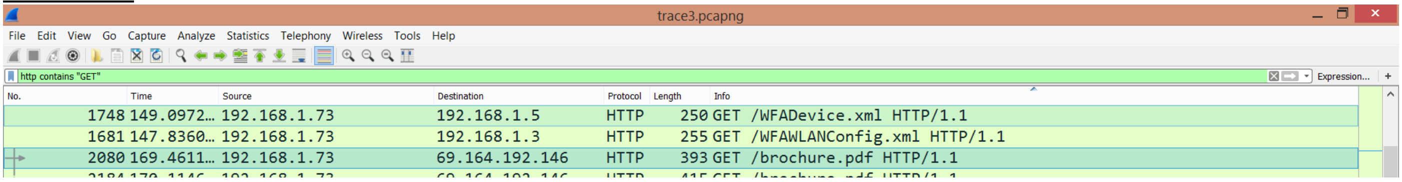
Interval 1 sec

□ Time of day

□ Log scale

The highest bits-per-second rate is 22700000 bits per second and it occurred in the 181st second.

Task 3. Q4.



http contains "GET"

No.	Time	Source	Destination	Protocol	Length	Info
1748	149.0972…	192.168.1.73	192.168.1.5	HTTP	250	GET /WFADevice.xml HTTP/1.1
1681	147.8360…	192.168.1.73	192.168.1.3	HTTP	255	GET /WFAWLANConfig.xml HTTP/1.1
2080	169.4611…	192.168.1.73	69.164.192.146	HTTP	393	GET /brochure.pdf HTTP/1.1
2184	170.1146…	192.168.1.73	69.164.192.146	HTTP	415	GET /brochure.pdf HTTP/1.1

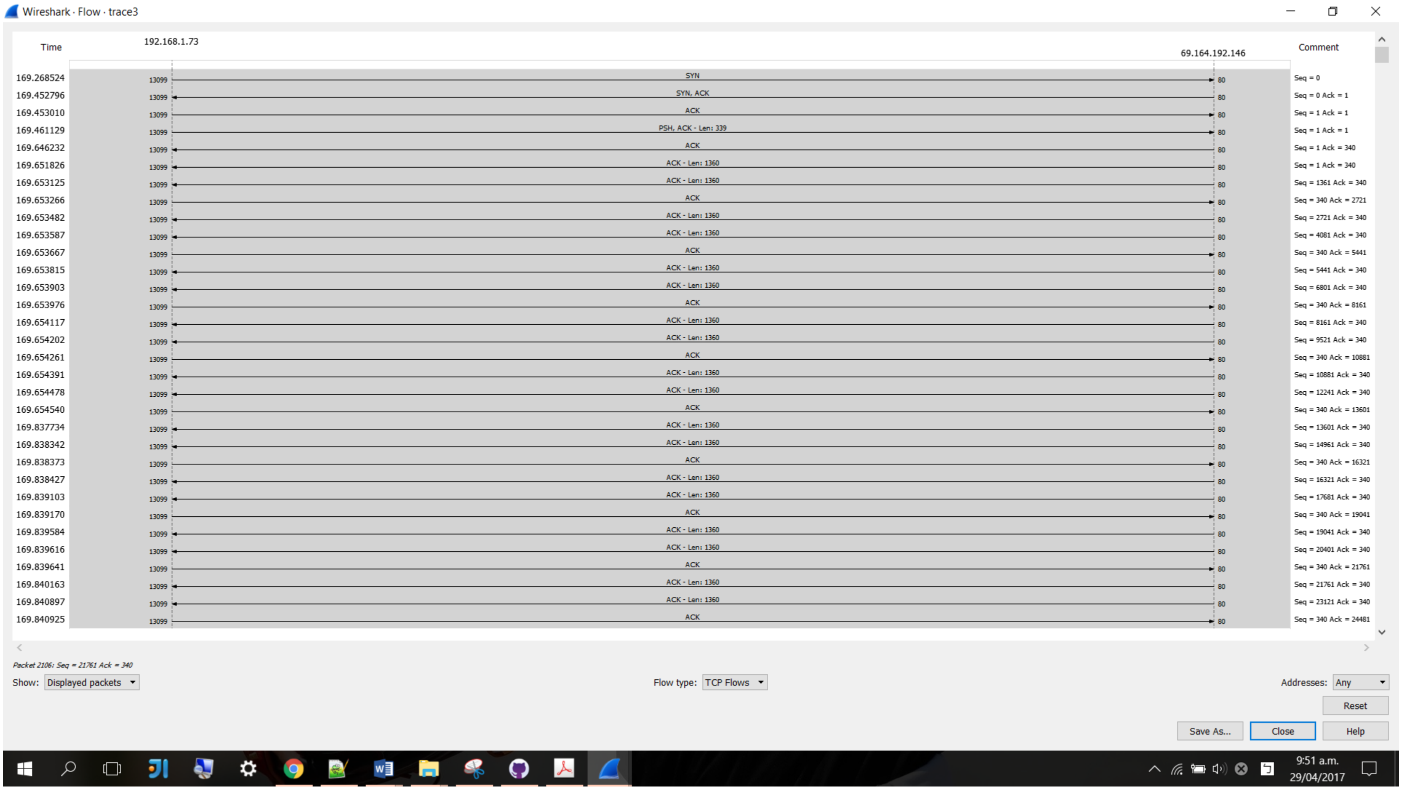
IP address of host: 192.168.1.73

IP address of client: 69.164.192.146

Task 3. Q5.

TCP conversation

- First 30:

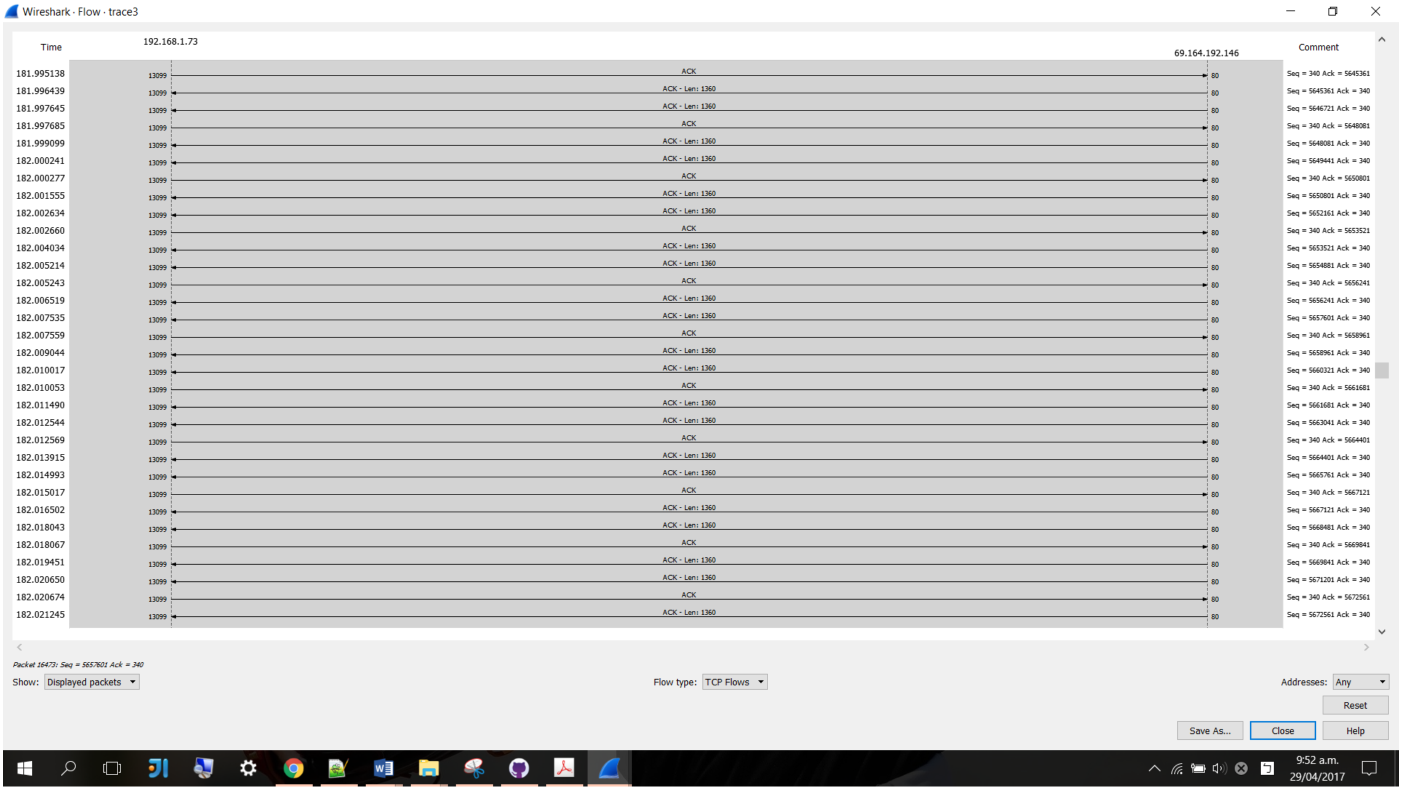


Wireshark · Flow · trace3

```
Time        192.168.1.73                                            69.164.192.146   Comment
169.268524     13099   SYN                                          80   Seq = 0
169.452796     13099   SYN, ACK                                     80   Seq = 0 Ack = 1
169.453010     13099   ACK                                          80   Seq = 1 Ack = 1
169.461129     13099   PSH, ACK - Len: 339                          80   Seq = 1 Ack = 1
169.646232     13099   ACK                                          80   Seq = 1 Ack = 340
169.651826     13099   ACK - Len: 1360                              80   Seq = 1 Ack = 340
169.653125     13099   ACK - Len: 1360                              80   Seq = 1361 Ack = 340
169.653266     13099   ACK                                          80   Seq = 340 Ack = 2721
169.653482     13099   ACK - Len: 1360                              80   Seq = 2721 Ack = 340
169.653587     13099   ACK - Len: 1360                              80   Seq = 4081 Ack = 340
169.653667     13099   ACK                                          80   Seq = 340 Ack = 5441
169.653815     13099   ACK - Len: 1360                              80   Seq = 5441 Ack = 340
169.653903     13099   ACK - Len: 1360                              80   Seq = 6801 Ack = 340
169.653976     13099   ACK                                          80   Seq = 340 Ack = 8161
169.654117     13099   ACK - Len: 1360                              80   Seq = 8161 Ack = 340
169.654202     13099   ACK - Len: 1360                              80   Seq = 9521 Ack = 340
169.654261     13099   ACK                                          80   Seq = 340 Ack = 10881
169.654391     13099   ACK - Len: 1360                              80   Seq = 10881 Ack = 340
169.654478     13099   ACK - Len: 1360                              80   Seq = 12241 Ack = 340
169.654540     13099   ACK                                          80   Seq = 340 Ack = 13601
169.837734     13099   ACK - Len: 1360                              80   Seq = 13601 Ack = 340
169.838342     13099   ACK - Len: 1360                              80   Seq = 14961 Ack = 340
169.838373     13099   ACK                                          80   Seq = 340 Ack = 16321
169.838427     13099   ACK - Len: 1360                              80   Seq = 16321 Ack = 340
169.839103     13099   ACK - Len: 1360                              80   Seq = 17681 Ack = 340
169.839170     13099   ACK                                          80   Seq = 340 Ack = 19041
169.839584     13099   ACK - Len: 1360                              80   Seq = 19041 Ack = 340
169.839616     13099   ACK - Len: 1360                              80   Seq = 20401 Ack = 340
169.839641     13099   ACK                                          80   Seq = 340 Ack = 21761
169.840163     13099   ACK - Len: 1360                              80   Seq = 21761 Ack = 340
169.840897     13099   ACK - Len: 1360                              80   Seq = 23121 Ack = 340
169.840925     13099   ACK                                          80   Seq = 340 Ack = 24481

Packet 2106: Seq = 21761 Ack = 340
Show: Displayed packets        Flow type: TCP Flows        Addresses: Any
```

- Middle:

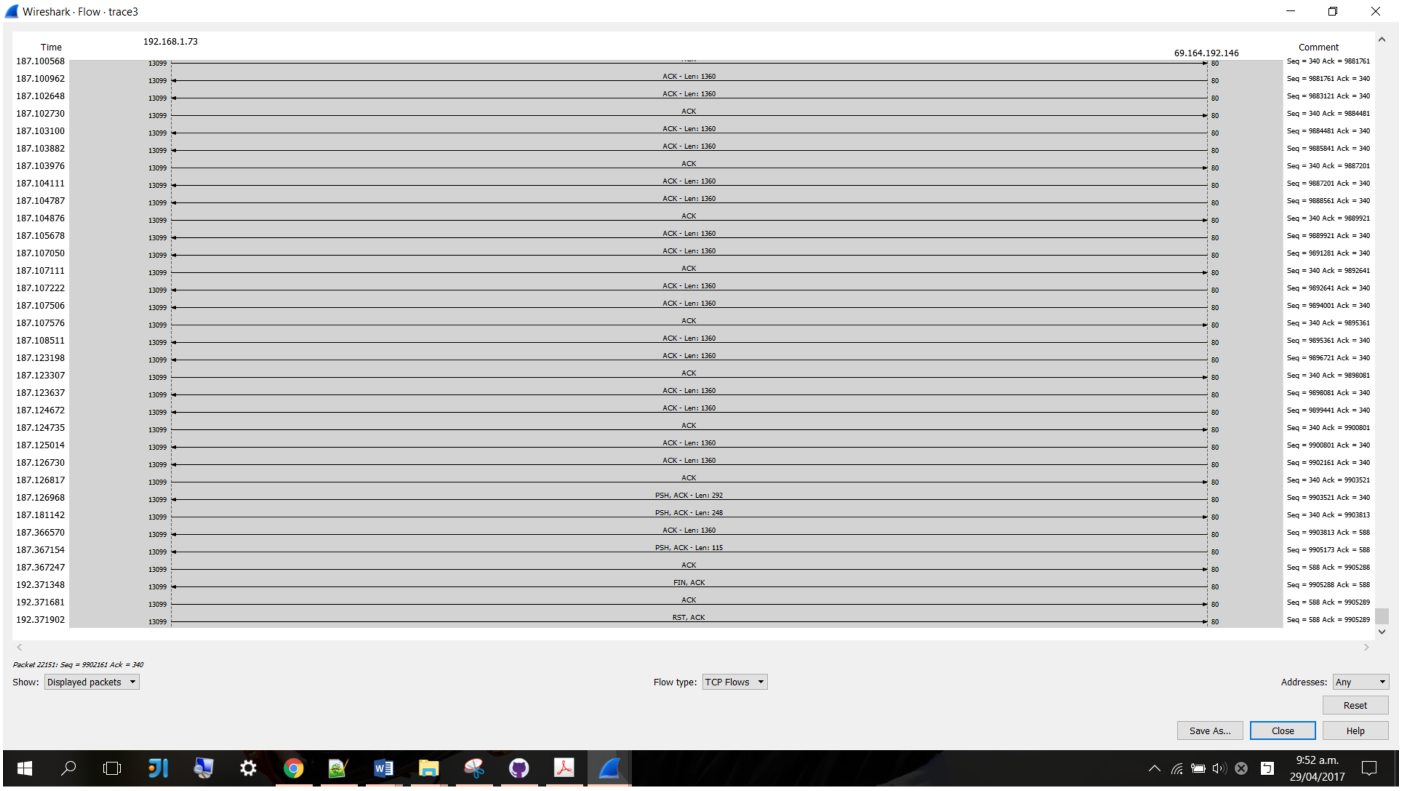


Wireshark · Flow · trace3

```
Time        192.168.1.73                                            69.164.192.146   Comment
181.995138     13099   ACK                                          80   Seq = 340 Ack = 5645361
181.996439     13099   ACK - Len: 1360                              80   Seq = 5645361 Ack = 340
181.997645     13099   ACK - Len: 1360                              80   Seq = 5646721 Ack = 340
181.997685     13099   ACK                                          80   Seq = 340 Ack = 5648081
181.999099     13099   ACK - Len: 1360                              80   Seq = 5648081 Ack = 340
182.000241     13099   ACK - Len: 1360                              80   Seq = 5649441 Ack = 340
182.000277     13099   ACK                                          80   Seq = 340 Ack = 5650801
182.001555     13099   ACK - Len: 1360                              80   Seq = 5650801 Ack = 340
182.002634     13099   ACK - Len: 1360                              80   Seq = 5652161 Ack = 340
182.002660     13099   ACK                                          80   Seq = 340 Ack = 5653521
182.004034     13099   ACK - Len: 1360                              80   Seq = 5653521 Ack = 340
182.005214     13099   ACK - Len: 1360                              80   Seq = 5654881 Ack = 340
182.005243     13099   ACK                                          80   Seq = 340 Ack = 5656241
182.006519     13099   ACK - Len: 1360                              80   Seq = 5656241 Ack = 340
182.007535     13099   ACK - Len: 1360                              80   Seq = 5657601 Ack = 340
182.007559     13099   ACK                                          80   Seq = 340 Ack = 5658961
182.009044     13099   ACK - Len: 1360                              80   Seq = 5658961 Ack = 340
182.010017     13099   ACK - Len: 1360                              80   Seq = 5660321 Ack = 340
182.010053     13099   ACK                                          80   Seq = 340 Ack = 5661681
182.011490     13099   ACK - Len: 1360                              80   Seq = 5661681 Ack = 340
182.012544     13099   ACK - Len: 1360                              80   Seq = 5663041 Ack = 340
182.012569     13099   ACK                                          80   Seq = 340 Ack = 5664401
182.013915     13099   ACK - Len: 1360                              80   Seq = 5664401 Ack = 340
182.014993     13099   ACK - Len: 1360                              80   Seq = 5665761 Ack = 340
182.015017     13099   ACK                                          80   Seq = 340 Ack = 5667121
182.016502     13099   ACK - Len: 1360                              80   Seq = 5667121 Ack = 340
182.018043     13099   ACK - Len: 1360                              80   Seq = 5668481 Ack = 340
182.018067     13099   ACK                                          80   Seq = 340 Ack = 5669841
182.019451     13099   ACK - Len: 1360                              80   Seq = 5669841 Ack = 340
182.020650     13099   ACK - Len: 1360                              80   Seq = 5671201 Ack = 340
182.020674     13099   ACK                                          80   Seq = 340 Ack = 5672561
182.021245     13099   ACK - Len: 1360                              80   Seq = 5672561 Ack = 340

Packet 16473: Seq = 5657601 Ack = 340
Show: Displayed packets        Flow type: TCP Flows        Addresses: Any
```

- Last 30:

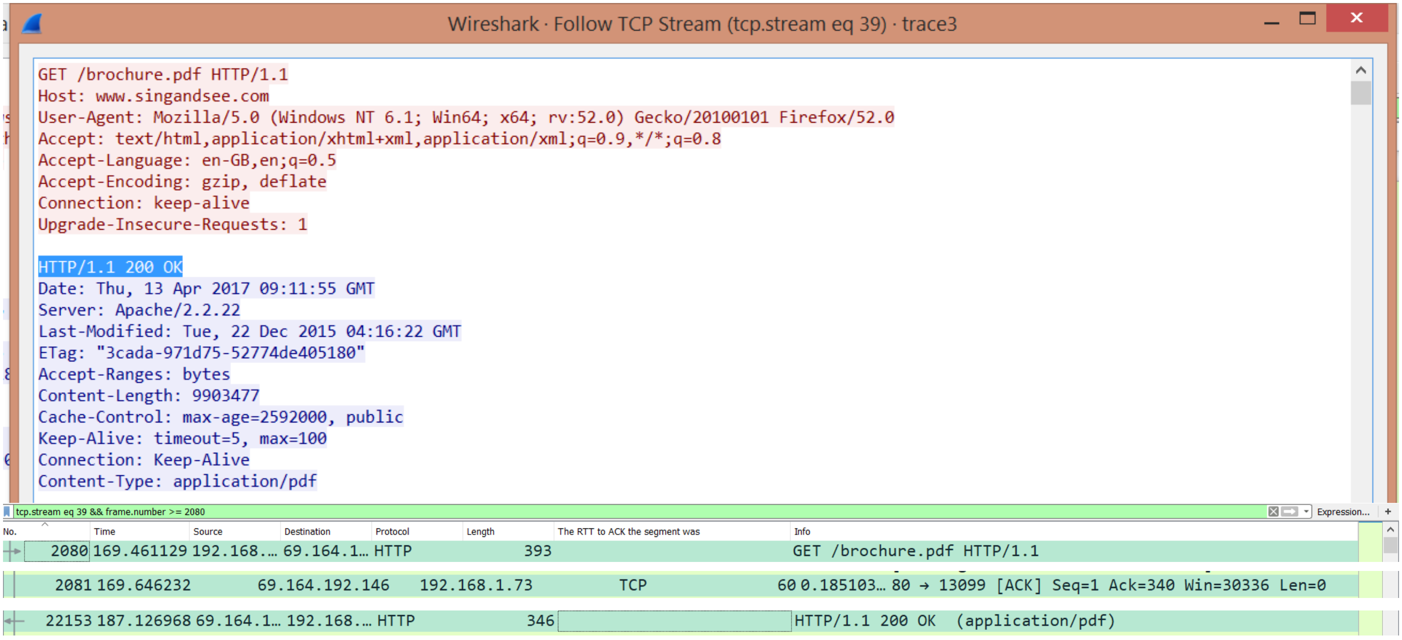


Wireshark · Flow · trace3

```
Time          192.168.1.73                                   69.164.192.146   Comment
187.100568    13099  ------------------------------------->  80   Seq = 340 Ack = 9881761
187.100962    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9881761 Ack = 340
187.102648    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9883121 Ack = 340
187.102730    13099  ---------------- ACK ----------------->  80   Seq = 340 Ack = 9884481
187.103100    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9884481 Ack = 340
187.103882    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9885841 Ack = 340
187.103976    13099  ---------------- ACK ----------------->  80   Seq = 340 Ack = 9887201
187.104111    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9887201 Ack = 340
187.104787    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9888561 Ack = 340
187.104876    13099  ---------------- ACK ----------------->  80   Seq = 340 Ack = 9889921
187.105678    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9889921 Ack = 340
187.107050    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9891281 Ack = 340
187.107111    13099  ---------------- ACK ----------------->  80   Seq = 340 Ack = 9892641
187.107222    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9892641 Ack = 340
187.107506    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9894001 Ack = 340
187.107576    13099  ---------------- ACK ----------------->  80   Seq = 340 Ack = 9895361
187.108511    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9895361 Ack = 340
187.123198    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9896721 Ack = 340
187.123307    13099  ---------------- ACK ----------------->  80   Seq = 340 Ack = 9898081
187.123637    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9898081 Ack = 340
187.124672    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9899441 Ack = 340
187.124735    13099  ---------------- ACK ----------------->  80   Seq = 340 Ack = 9900801
187.125014    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9900801 Ack = 340
187.126730    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9902161 Ack = 340
187.126817    13099  ---------------- ACK ----------------->  80   Seq = 340 Ack = 9903521
187.126968    13099  <-------- PSH, ACK - Len: 292 ---------  80   Seq = 9903521 Ack = 340
187.181142    13099  --------- PSH, ACK - Len: 248 -------->  80   Seq = 340 Ack = 9903813
187.366570    13099  <----------- ACK - Len: 1360 ----------  80   Seq = 9903813 Ack = 588
187.367154    13099  <-------- PSH, ACK - Len: 115 ---------  80   Seq = 9905173 Ack = 588
187.367247    13099  ---------------- ACK ----------------->  80   Seq = 588 Ack = 9905288
192.371348    13099  <------------- FIN, ACK ---------------  80   Seq = 9905288 Ack = 588
192.371681    13099  ---------------- ACK ----------------->  80   Seq = 588 Ack = 9905289
192.371902    13099  -------------- RST, ACK -------------->  80   Seq = 588 Ack = 9905289

Packet 22151: Seq = 9902161 Ack = 340
Show: Displayed packets          Flow type: TCP Flows          Addresses: Any
```

Task 3. Q6.



Wireshark · Follow TCP Stream (tcp.stream eq 39) · trace3

```
GET /brochure.pdf HTTP/1.1
Host: www.singandsee.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:52.0) Gecko/20100101 Firefox/52.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1

HTTP/1.1 200 OK
Date: Thu, 13 Apr 2017 09:11:55 GMT
Server: Apache/2.2.22
Last-Modified: Tue, 22 Dec 2015 04:16:22 GMT
ETag: "3cada-971d75-52774de405180"
Accept-Ranges: bytes
Content-Length: 9903477
Cache-Control: max-age=2592000, public
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: application/pdf
```

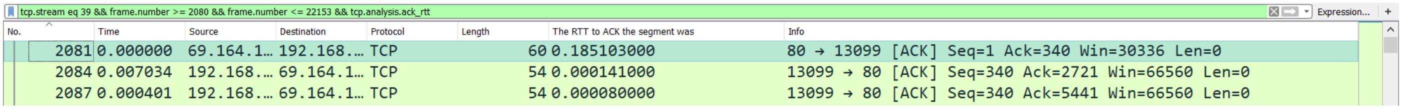
```
tcp.stream eq 39 && frame.number >= 2080
No.    Time        Source          Destination     Protocol  Length  The RTT to ACK the segment was  Info
 2080  169.461129  192.168.…       69.164.1…       HTTP      393                                     GET /brochure.pdf HTTP/1.1
 2081  169.646232  69.164.192.146  192.168.1.73    TCP       60      0.185103…                       80 → 13099 [ACK] Seq=1 Ack=340 Win=30336 Len=0
22153  187.126968  69.164.1…       192.168.…       HTTP      346                                     HTTP/1.1 200 OK  (application/pdf)
```

- Download time for the client side = 187.1269 – 169.4611 = 17.67 seconds
This includes the HTTP GET message because it is required for the client to set up the connection.
- Download time for the server side = 187.1269 – 169.646232 = 17.48 seconds
This excludes the HTTP GET message because the download session starts when the server starts sending the data segments.

Task 3. Q7.

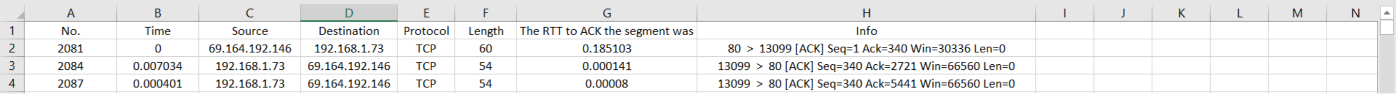
Apply display filter:

tcp.stream eq 39 && frame.number >= 2080 && frame.number <= 22153 && tcp.analysis.ack_rtt



```
tcp.stream eq 39 && frame.number >= 2080 && frame.number <= 22153 && tcp.analysis.ack_rtt
No.   Time      Source     Destination  Protocol  Length  The RTT to ACK the segment was  Info
2081  0.000000  69.164.1…  192.168.…    TCP       60      0.185103000                     80 → 13099 [ACK] Seq=1 Ack=340 Win=30336 Len=0
2084  0.007034  192.168.…  69.164.1…    TCP       54      0.000141000                     13099 → 80 [ACK] Seq=340 Ack=2721 Win=66560 Len=0
2087  0.000401  192.168.…  69.164.1…    TCP       54      0.000080000                     13099 → 80 [ACK] Seq=340 Ack=5441 Win=66560 Len=0
```

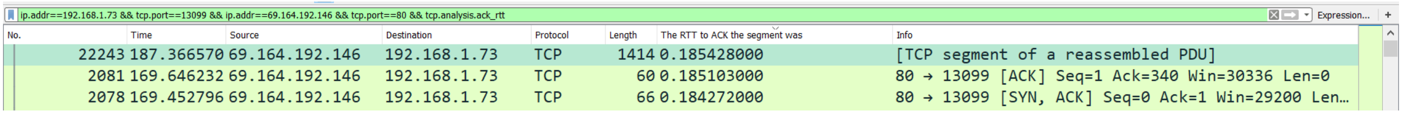
Export to a csv file and use the AVERAGE() function in column G.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	No.	Time	Source	Destination	Protocol	Length	The RTT to ACK the segment was	Info						
2	2081	0	69.164.192.146	192.168.1.73	TCP	60	0.185103	80 > 13099 [ACK] Seq=1 Ack=340 Win=30336 Len=0						
3	2084	0.007034	192.168.1.73	69.164.192.146	TCP	54	0.000141	13099 > 80 [ACK] Seq=340 Ack=2721 Win=66560 Len=0						
4	2087	0.000401	192.168.1.73	69.164.192.146	TCP	54	0.00008	13099 > 80 [ACK] Seq=340 Ack=5441 Win=66560 Len=0						

The average RTT was 0.000254041 seconds.

Task 3. Q8.



ip.addr==192.168.1.73 && tcp.port==13099 && ip.addr==69.164.192.146 && tcp.port==80 && tcp.analysis.ack_rtt

No.	Time	Source	Destination	Protocol	Length	The RTT to ACK the segment was	Info
22243	187.366570	69.164.192.146	192.168.1.73	TCP	1414	0.185428000	[TCP segment of a reassembled PDU]
2081	169.646232	69.164.192.146	192.168.1.73	TCP	60	0.185103000	80 → 13099 [ACK] Seq=1 Ack=340 Win=30336 Len=0
2078	169.452796	69.164.192.146	192.168.1.73	TCP	66	0.184272000	80 → 13099 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len…

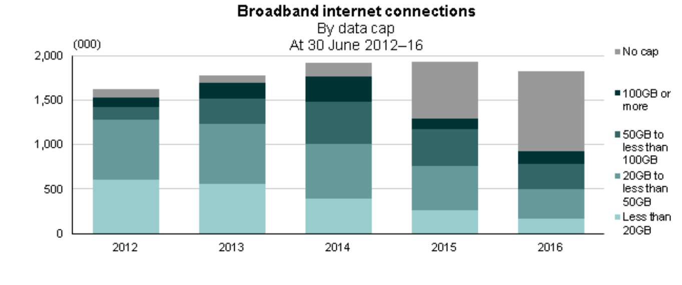
The longest RTT was 0.185103 seconds and it occurred in the 169.646232 th second. This occurred in packet 2081.

Task 4

Overview

As at 30 June 2016, the number of fibre-optic connections in New Zealand has increased more than double the number of connections at the same time last year. Almost 50 percent of all broadband connections in New Zealand had no data cap. This is in response to a tremendous increase in demand for streaming or on-demand services such as TV or movie streaming, online radio or music streaming, online gaming, and content creation and sharing. (Statistics New Zealand, 2016)

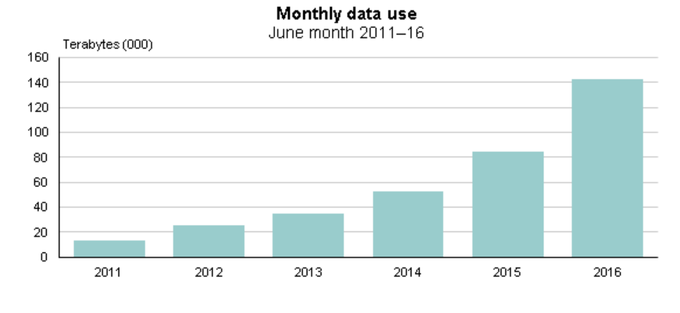
Furthermore, each New Zealand residential connection used approximately 88 gigabytes on average, which equates to approximately 85 hours of streaming TV or movies per month. Since 2013, New Zealand has seen a proliferation of video streaming services launching. These include global market leader Netflix, Sky-TV's Neon, Spark's Lightbox service, as well as Quickflix. (MacMahon, R., & Milner, M., 2015) Nearly a quarter (24%) of New Zealanders subscribe to digital content such as Netflix and Spotify, and consume content at a time that suits them. (Nielsen, 2016)



Source: Statistics New Zealand

Figure 1 Broadband internet connections between 2012 and 2016

(Statistics New Zealand, 2016)



Source: Statistics New Zealand

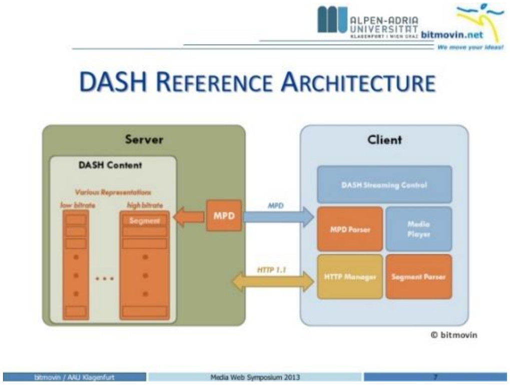
Figure 2 Monthly data usage between 2012 and 2016

(Statistics New Zealand, 2016)

Video traffic is the major consumer of internet bandwidth, because multimedia files are generally large in nature. Videos are a sequence of images displayed at a constant rate, while digital images are arrays of pixels.There exist a very diverse range of video communication and streaming applications, which have very distinctive operating conditions or properties. Video communication system designs significantly depend on the characteristics of the communication channel, such as bandwidth, delay, and loss.

Some video channels support Constant Bit Rate (CBR), such as ISDN or DTV. On the other hand, some channels support Variable Bit Rate (VBR), for example, DVD storage and communication over shared packet networks. (Apostolopoulos, J. G., Tan, W. T., & Wee, S. J., 2002) In a video streaming application, a client typically begins playout of the video while the file is being downloaded from the server. This means the client will be playing out the video from one location in the file while it is receiving later parts of the file from the server. Hence, streaming avoids having to download the entire file before beginning the playout, which can potentially incur a longer delay.

Several network protocols have been designed and standardised for communication between clients and streaming servers. The Internet protocol (IP) serves as the network layer protocol for Internet video streaming. The transport layer provides end-to-end network transport functions for streaming applications. Transport protocols include UDP, TCP, real-time transport protocol (RTP), and real-time control protocol (RTCP). UDP and TCP protocols support functions such as multiplexing, error control, congestion control, and flow control. RTP and RTCP are implemented on top of UDP/TCP. The application layer contains several protocols including HTTP and DASH. HTTP is used to exchange or transfer hypertext, while DASH is a technique that enables high-quality streaming of media content over the Internet to be delivered from conventional HTTP web servers. Moreover, DASH-based streaming is very different from UDP-based streaming as it involves adaptation both by the application and by TCP. (Martin, J., Fu, Y., Wourms, N., & Shaw, T., 2013)



DASH Reference Architecture

Figure 3 DASH Reference Architecture (Stefan Lederer, 2013)

The media content exists on the server in two parts: a Media Presentation Description (MPD) and segments. MPD essentially describes a manifest of the available content and their URL addresses in XML format, while the segments contain data segments of the actual multimedia bitstreams. By parsing the MPD, the DASH client learns about the content details and starts fetching the segments using HTTP GET requests. (Sodagar, I., 2011)

The DASH standards focus primarily on the syntax required to describe content manifests and video segments. In order to prevent rebuffering due to buffer starvation, a bit rate level less than the measured available bandwidth is usually chosen. Because of the benefits that HTTP-streaming based technology implies and due to the fact that DASH is a company-independent standard, YouTube and Netflix have implemented DASH as the preferred streaming technology rather than Flash Video streaming.

Furthermore, contend distribution networks (CDNs) replicate stored content and put the replicated content at the edges of the Internet. Techniques such as local caches are used to stream content to potentially millions of users. There are two main types of CDNs: ‘enter deep’ and ‘bring home’.

‘Enter deep’ CDNs push a large number of CDN servers deep into many access networks. They are closer to the users for improved user-perceived performance in terms of both delay and throughput. However, they are more expensive and more challenging to manage. It also involves sophisticated algorithms to shuffle data among the servers across the public Internet. On the other hand, ‘bring home’ CDNs use a small number of larger clusters in Point of Presences near the access networks, but not within them. Consequently, large content distribution centres are built at only a few key locations and private high-speed connections are used. A typical key location is simultaneously near the POPs of many large ISPs. This approach typically results in lower maintenance and management overhead, possibly at the expense of higher delay to the end users. (Huang, C., Wang, A., Li, J., & Ross, K. W., 2008) Given that a large portion of the traffic flowing through the Internet is stored content such as multimedia, CDNs can significantly reduce the traffic loads on the ISPs and the related interfaces between ISPs. Leading commercial CDN representatives of ‘enter deep’ and ‘bring home’ are Akamai and Limelight respectively.

Over-the-top content (OTT) refers to audio, video, and other media transmitted via the Internet as a standalone product. OTT content includes Youtube and Netflix. It bypasses the traditional operator’s network. These operators are so-called multiple-system operators. Hence, the OTT players do not require any business or technology affiliations with network operators for providing such services. OTT services generally rely on streaming protocols such as HTTP adaptive bitrate streaming, where a video is split into fixed length chunks and stored on one or more HTTP servers or cached within the CDN for quick access.

Controlling the perceived video quality or quality of experience (QoE) is a major challenge for OTT service providers. (Satti, S. M., Bitto, R., Keyhl, M., Obermann, M., & Schmidmer, C., 2016) Sometimes it is hard for OTT content to reproduce the same video resolution as traditional cable television, especially when compared side-by-side to a larger screen. Although OTT videos are distributed over dedicated CDN infrastructures, OTT content is delivered over the more congestion-prone and less managed open Internet. Commonly known distortions in OTT are initial-loading, stalling, and coding/quality-switching. (Satti, S. M., Bitto, R., Keyhl, M., Obermann, M., & Schmidmer, C., 2016)

Netflix

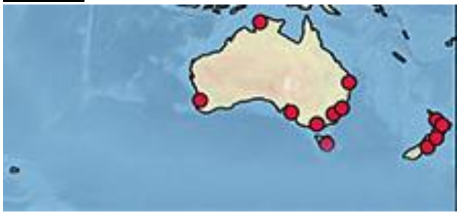


Figure 4 This map depicts the location of Netflix servers found in a recent research (Amy Nordrum, 2016)

Netflix is the leading subscription service provider for TV shows and online movies. It is the single largest source of Internet traffic in the US, consuming 29.7% of peak downstream traffic. (Adhikari, V. K., Guo, Y., Hao, F., Varvello, M., Hilt, V., Steiner, M., & Zhang, Z. L., 2012) In March 2016, Netflix claimed to deliver about 125 million total hours of viewing to customers per day. (Netflix Media Center, 2016) There are approximately 4 Netflix servers in new Zealand, while there are 4669 servers in 243 global locations. (Amy Nordrum, 2016) Netflix servers have two main functionalities. The first is to capture payment and user registration information, while the second is user redirection. The users who are successfully logged in and those who are not are redirected to different IP addresses. In addition, the servers do not interact with the clients directly during video streaming.

Netflix uses cloud services such as Amazon AWS cloud, CDNs, and other public services. Amazon cloud provides key functionalities such as CDN routing and mobile device support. Video streaming of Netflix is served out of multiple CDNs, and UltraDNS, which is a public DNS service used as its authoritative servers. The encoded and DRM protected videos are sourced in Amazon and copied to CDNs. The average bandwidth of CDNs can vary significantly over time and over geographic locations.

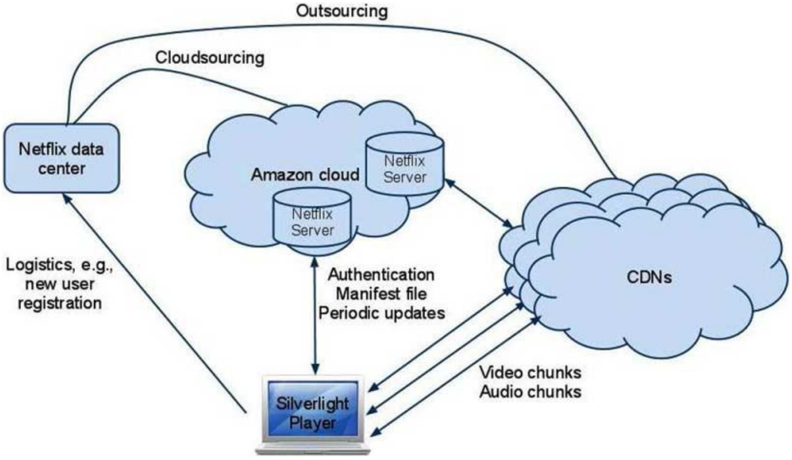


Figure 5 Netflix Architecture (Adhikari, V. K., Guo, Y., Hao, F., Varvello, M., Hilt, V., Steiner, M., & Zhang, Z. L., 2012)

Netflix's content delivery strategy stores content at both IXPs and ISPs. In the United States, Netflix is primarily delivered through IXPs and Akamai's 'enter deep' strategy is likely to be used. Meanwhile, there are no Netflix servers at IXPs in Canada or Mexico. Customers in those countries are served exclusively by servers within ISPs, as well as possibly through IXPs along the U.S. borders. (Amy Nordrum, 2016) Limelight's 'bring home' strategy is likely to be used in these countries. Netflix's centralised approach outside the U.S. is convenient because it has more control and scalability for the local market.

Netflix uses the DASH protocol over TCP for streaming. The DASH players of Netflix are allowed to freely switch between different quality levels at the chunk boundaries. However, Netflix clients do not try all possible available bitrates when trying to determine the optimal playback rate. Netflix video streaming is controlled by instructions in a manifest file that provides the DASH player metadata. The manifest files are client-specific and delivered via SSL connection. They also rank CDNs to indicate the preferred CDNs. (Adhikari, V. K., Guo, Y., Hao, F., Varvello, M., Hilt, V., Steiner, M., & Zhang, Z. L., 2012) In general, the Netflix client appears to stay with the same CDN as long as possible even if it has to degrade the quality level of the playback, while other CDNs appear to serve only as backups. Furthermore, Netflix's adaptation appears to default to TCP control during periods of heavy, sustained network congestion. However, the application algorithm is clearly intertwined with TCP control during periods of volatile network conditions. (Martin, J., Fu, Y., Wourms, N., & Shaw, T., 2013)

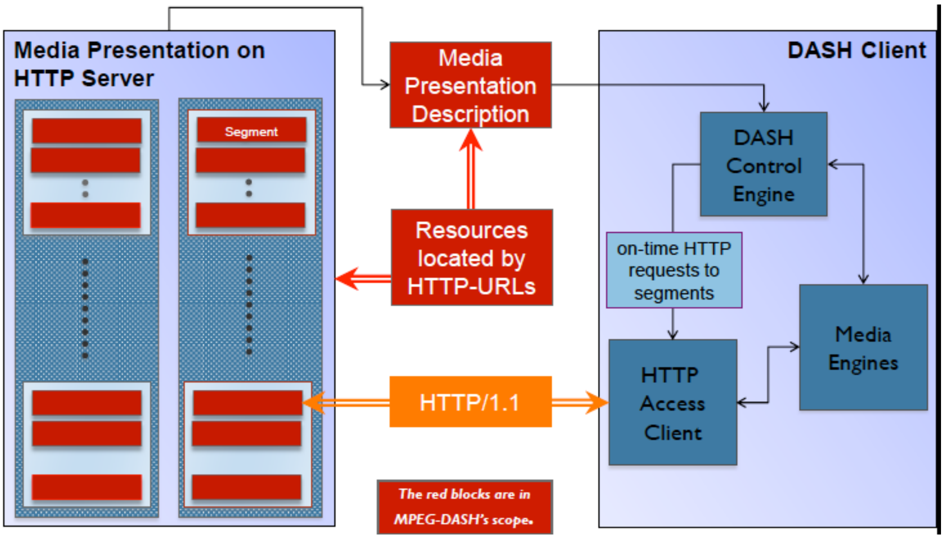

Media Presentation on HTTP Server
Segment
Media Presentation Description
Resources located by HTTP-URLs
HTTP/1.1
The red blocks are in MPEG-DASH's scope.
DASH Client
DASH Control Engine
on-time HTTP requests to segments
Media Engines
HTTP Access Client

Figure 6 DASH System Model (Stefan Lederer, 2013)

The DASH client of Netflix contains three main functional areas: the HTTP access, the media engine, and the control engine that monitors the arriving video stream and determines the quality of stream to request. (Martin, J., Fu, Y., Wourms, N., & Shaw, T., 2013) The client will maintain a playback buffer that smooths variable content arrival rates to support the video playback. The client requests a new segment tagged with the desired bitrate once the playback buffer drops below a certain threshold. When the playback buffer is almost empty, the client is likely to go into a 'buffering' state where it requests segments at a high rate. (Martin, J., Fu, Y., Wourms, N., & Shaw, T., 2013)

Hence, the session operates in one of two states: buffering or steady state. While in a buffering state, the client requests data at a rate up to the available bandwidth over the path between the server and the client. If conditions permit, the session will attempt to find a 'steady state' where it requests segments at a rate necessary to playback the content at a given encoded bitrate. (Adhikari, V. K., Guo, Y., Hao, F., Varvello, M., Hilt, V., Steiner, M., & Zhang, Z. L., 2012) Moreover, Microsoft Silverlight is used by Netflix to play, download, and decode its multimedia content. However, Netflix is gradually moving to HTML5 video.

<u>YouTube</u>

YouTube is one of the most popular services on the internet, being the most visited website in the world. Three hundred hours of video is uploaded to YouTube every minute. (FortuneLords, 2017)

YouTube's content delivery strategy prefers to store content at IXPs. (Amy Nordrum, 2016) YouTube uses a Flash-based client which is embedded in the web page. The communication between the client and server is done over HTTP. Consequently, YouTube does not have to cope with lost or reordered packets. However, a stalling of the video may be caused by transmission problems. HTTP over TCP provides two advantages. First, TCP is more firewall friendly compared to UDP as firewalls are more likely to block UDP. Second, the client manages HTTP streaming without having to maintain a session state on the server. (Sodagar, I., 2011)

YouTube used conventional HTTP streaming and progressive downloads prior to 2013. It later adapted DASH to reduce the overhead transmissions created by the previous two approaches. By using DASH, YouTube is able to switch the video quality based on the recent link capabilities. This enables YouTube's player quality parameter. If it is set to 'auto', YouTube can adapt the bitrate of the video based on the client's available bandwidth. (Añorga, J., Arrizabalaga, S., Sedano, B., Alonso-Arce, M., & Mendizabal, J. , 2015)

YouTube's streaming strategy has two phases: a burst phase and a throttling phase. The burst phase is the initial phase where there exists a significant burst of data. After this initial burst, the receiving download data rate of YouTube's player is considerably reduced. (Krishnappa, D. K., Bhat, D., & Zink, M., 2013) This strategy reduces congestion in the network by eliminating the transmission of unused data.

DASH provides the advantages in bandwidth and cost saving for YouTube, especially when the majority of users do not watch more than the first 20% of a video. However, one disadvantage of DASH is the amount of storage required to host all the videos in various bit rates and segment lengths on YouTube. (Añorga, J., Arrizabalaga, S., Sedano, B., Alonso-Arce, M., & Mendizabal, J. , 2015) Furthermore, YouTube uses both HTML5 video and Adobe Flash for multimedia streaming. HTML5 is defaulted for mobile devices, while Adobe Flash is defaulted for Pcs.

Lightbox

Lightbox is a New Zealand subscription video on demand (SVOD) service owned by Spark New Zealand. The service offers a selection of television shows over a wide range of devices. (Lightbox (New Zealand), 2017) Lightbox was initially launched as ShowmeTV in 2014 and currently faces competitions from other SVOD services including Quickflix, Netflix, and Neon. (Keall, Chris., 2014) (Damien Venuto, 2015) While over one million New Zealanders have access to Netflix, Lightbox shows the fastest growth of all SVOD providers in 2017. (StopPress, 2017)

According to our previous discussion, Lightbox is likely to utilise the centralised approach, where the customers are served exclusively by servers within ISPs. Akamai's 'enter deep' approach is likely to be used and the ISP here is likely to be the parent company, Spark.Similarly, this approach is convenient because it has more control and scalability for the New Zealand market.

At the heart of Lightbox's OTT services was Xstream's MediaMaker OTT platform that was used to provide customised OTT and TV everywhere services. The key features of MediaMaker include various management, automation, and scheduling services. (Rapid TV News, 2014)

Bibliography

Adhikari, V. K., Guo, Y., Hao, F., Varvello, M., Hilt, V., Steiner, M., & Zhang, Z. L. (2012). Unreeling netflix: Understanding and improving multi-cdn movie delivery. *INFOCOM* (pp. 1620-1628). IEEE.

Amy Nordrum. (2016, August 30). *Researchers Map Locations of 4,669 Servers in Netflix's Content Delivery Network*. Retrieved from IEEE Spectrum: http://spectrum.ieee.org/tech-talk/telecom/internet/researchers-map-locations-of-4669-servers-in-netflixs-content-delivery-network

Añorga, J., Arrizabalaga, S., Sedano, B., Alonso-Arce, M., & Mendizabal, J. . (2015). YouTube's DASH implementation analysis. *19th International Conference on Circuits, Systems, Communications and Computers (CSCC)*, (pp. 61-66).

Apostolopoulos, J. G., Tan, W. T., & Wee, S. J. (2002). *Video Streaming: Concepts, Algorithms, and Systems.* HP Laboratories, report HPL-2002-260.

Damien Venuto. (2015, January 20). *Spark continues to evolve, offers another subscriber perk with free Lightbox*. Retrieved from StopPress: http://stoppress.co.nz/news/telco-or-media-company-continued-evolution-spark

FortuneLords. (2017, March 23). *36 Mind Blowing YouTube Facts, Figures and Statistics – 2017*. Retrieved from FortuneLords: https://fortunelords.com/youtube-statistics/

Huang, C., Wang, A., Li, J., & Ross, K. W. (2008). Measuring and evaluating large-scale CDNs. *ACM IMC, 8*.

Keall, Chris. (2014, August 28). *Spark's Lightbox launches, with a little surprise from the boss*. Retrieved from National Business Review: http://www.nbr.co.nz/article/sparks-lightbox-launches-little-surprise-boss-ck-161495

Krishnappa, D. K., Bhat, D., & Zink, M. (2013). DASHing YouTube: An analysis of using DASH in YouTube video service. *2013 IEEE 38th Conference* (pp. 407-415). IEEE.

Kurose, J. F., & Ross, K. W. (2010). *Computer networking: a top-down approach* (Vol. 5). Addison-Wesley.

Lightbox (New Zealand). (2017, April 16). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Lightbox_(New_Zealand)

MacMahon, R., & Milner, M. (2015). Ultra-fast broadband in New Zealand: Progress accelerating. *Australian Journal of Telecommunications and the Digital Economy, 3*, 12.

Martin, J., Fu, Y., Wourms, N., & Shaw, T. (2013). Characterizing Netflix bandwidth consumption. *Consumer Communications and Networking Conference (CCNC)* (pp. 230-235). IEEE.

Netflix Media Center. (2016, March 17). *How Netflix Works With ISPs Around the Globe to Deliver a Great Viewing Experience*. Retrieved from Netflix Media Center: https://media.netflix.com/en/company-blog/how-netflix-works-with-isps-around-the-globe-to-deliver-a-great-viewing-experience

Nielsen. (2016, April 14). *NEW ZEALAND MEDIA TRENDS REPORT 2016*. Retrieved from Nielsen: http://www.nielsen.com/nz/en/insights/reports/2016/new-zealand-media-trends-2016.html

Rao, A., Legout, A., Lim, Y. S., Towsley, D., Barakat, C., & Dabbous, W. (2011). Network characteristics of video streaming traffic. *Seventh COnference on emerging Networking EXperiments and Technologies* (p. 25). ACM.

Rapid TV News. (2014, August 28). *Xstream switches on Lightbox New Zealand OTT service*. Retrieved from Rapid TV News: https://www.rapidtvnews.com/2014082835072/xstream-switches-on-lightbox-new-zealand-ott-service.html

Satti, S. M., Bitto, R., Keyhl, M., Obermann, M., & Schmidmer, C. (2016, December). Long-term quality evaluation in OTT video. *In Picture Coding Symposium (PCS)*, 1-5.

Sodagar, I. (2011). The mpeg-dash standard for multimedia streaming over the internet. *IEEE MultiMedia, 18*(4), 62-67.

Statistics New Zealand. (2016, October). *Internet Service Provider Survey: 2016*. Retrieved from Statistics New Zealand: http://www.stats.govt.nz/browse_for_stats/industry_sectors/information_technology_and_communications/ISPSurvey_HOTP2016/Commentary.aspx

Stefan Lederer. (2013, March 15). *MPEG-DASH open source tools and cloud services*. Retrieved from SlideShare: https://www.slideshare.net/slederer/mpegdash-open-source-tools-and-cloud-services

StopPress. (2017, April 21). *One in 10 New Zealanders have two or more SVODs' – Roy Morgan research*. Retrieved from StopPress: http://stoppress.co.nz/news/one-10-new-zealanders-have-two-or-more-svods

Sujata, J., Sohag, S., Tanu, D., Chintan, D., Shubham, P., & Sumit, G. (2015). Impact of Over the Top (OTT) Services on Telecom Service Providers. *Indian Journal of Science and Technology, 8*, 145-160.

Wu, D., Hou, Y. T., Zhu, W., Zhang, Y. Q., & Peha, J. M. (2001). Streaming video over the Internet: approaches and directions. *IEEE Transactions on circuits and systems for video technology, 11*, 282-300.

Table of Figures