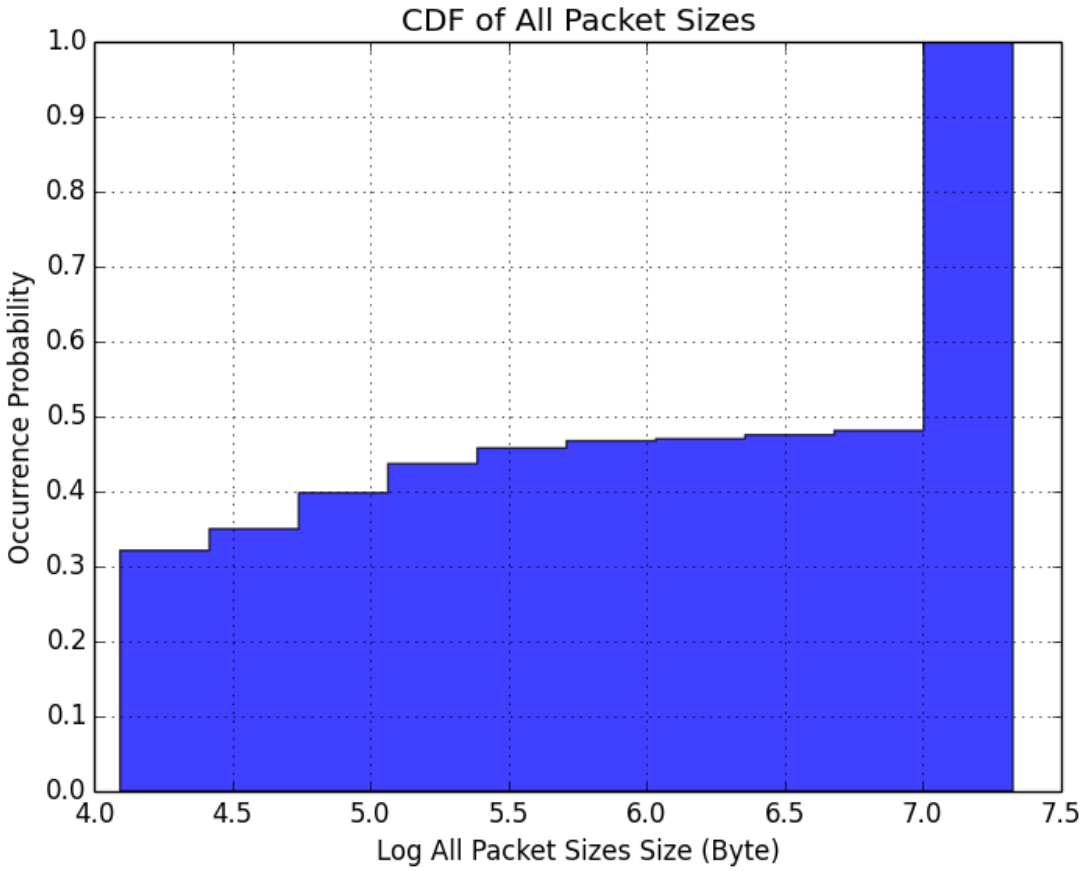
Tool: Wireshark
Language: Python
Introduction:
We use wireshark to select all the required fields and output them to csv file("row4.csv" in code.zip)
We then use a python script to generate all the diagrams based on the data read from the csv file

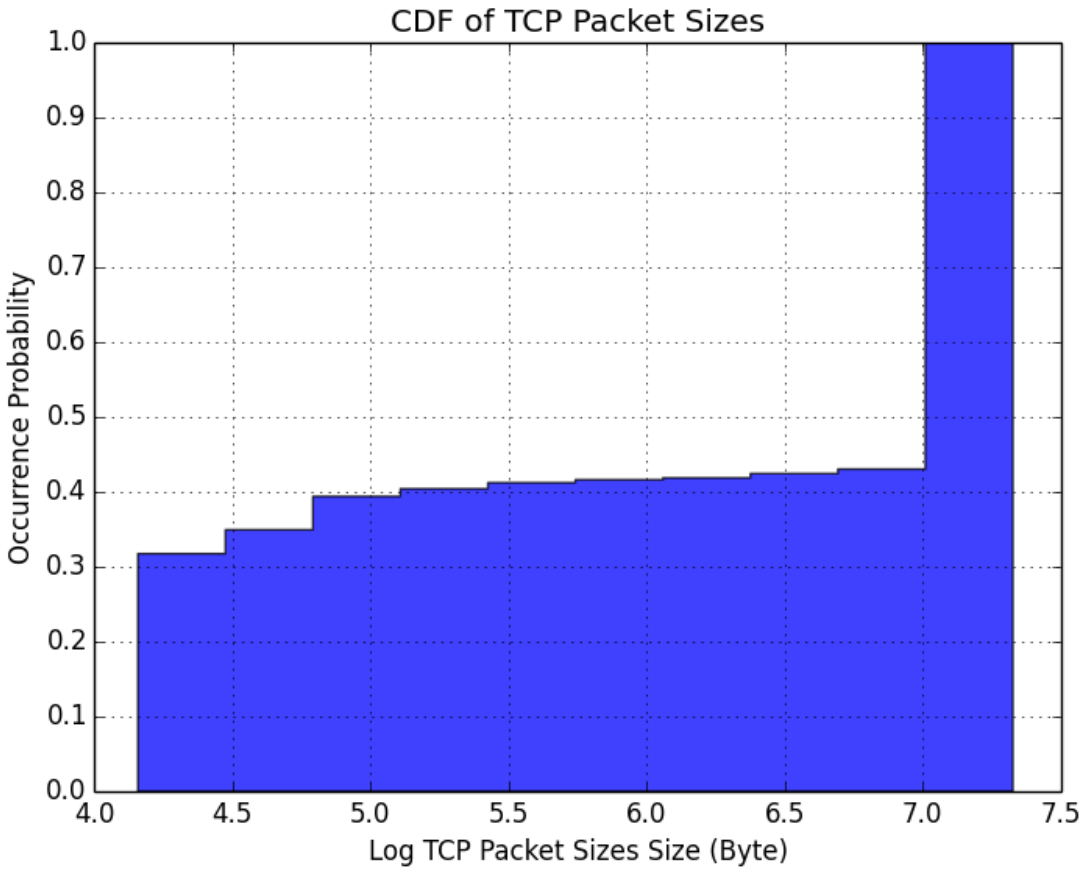
Per-packet statistics:

Type of packet	layer	quantity	percentage	Total bytes
Ethernet	Link layer	969165	100%	13568310
MX Network Load Balancing	Network layer	353	0.036423%	26274
Logical-link Control	Network layer	5436	0.560895%	267609
Internetwork Packet eXchange	Network layer	2	0.00020%	60
IPV4	Network layer	944734	97.4791%	18894752
IPV6	Network layer	4	0.00041%	160
Address Resolution Protocol	Network layer	19304	1.99181%	540512
Other	Network layer	546	0.05532%	22928
ESP (encapsulating security payload)	Transport layer	7215	0.74445%	418470
Virtual Router Redundancy Protocol	Transport layer	209	0.00029%	2272
	Transport layer	56473	5.82660%	451784
TCP	Transport layer	873001	90.07764%	42876904
PIM (Protocol independent multicast)	Transport layer	152	0.015683%	5776
OSPF	Transport layer	164	0.016922%	9358
Other	Transport layer	5767	0.595292%	229604

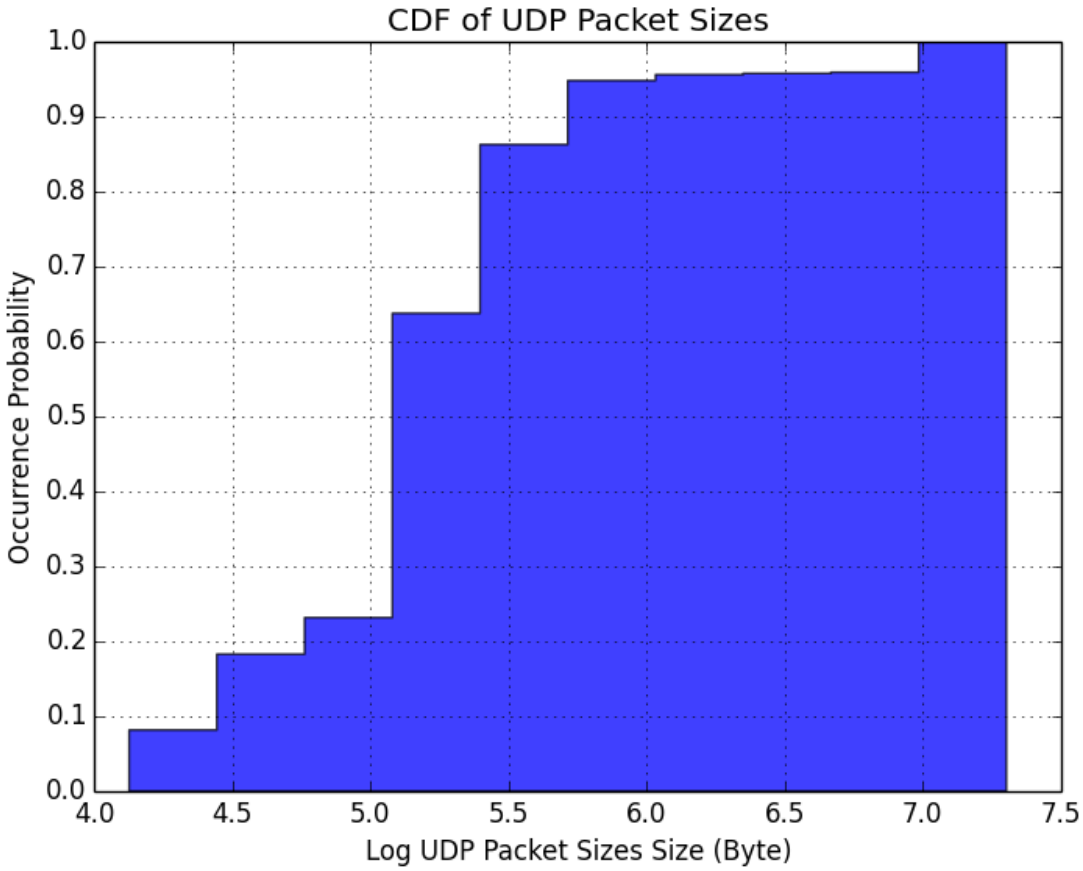
The Size of all Packets:
CDF of All Packets:



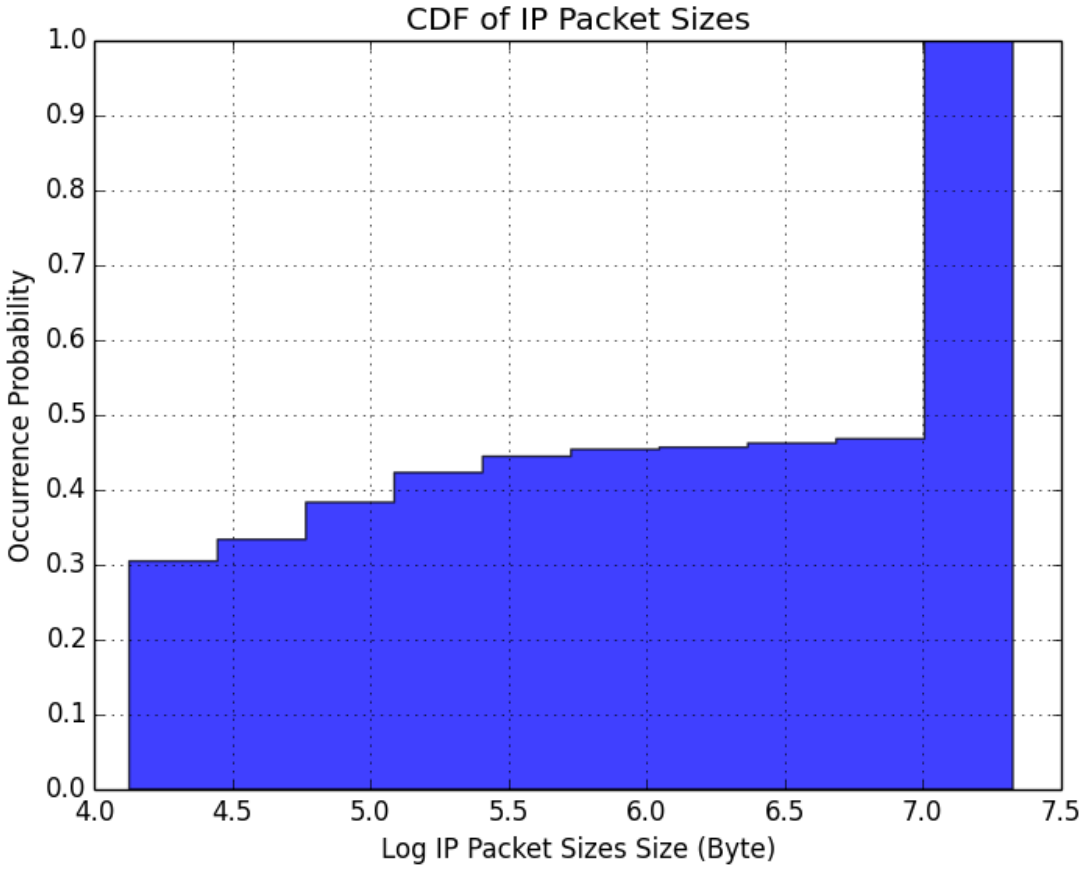
CDF of TCP packets



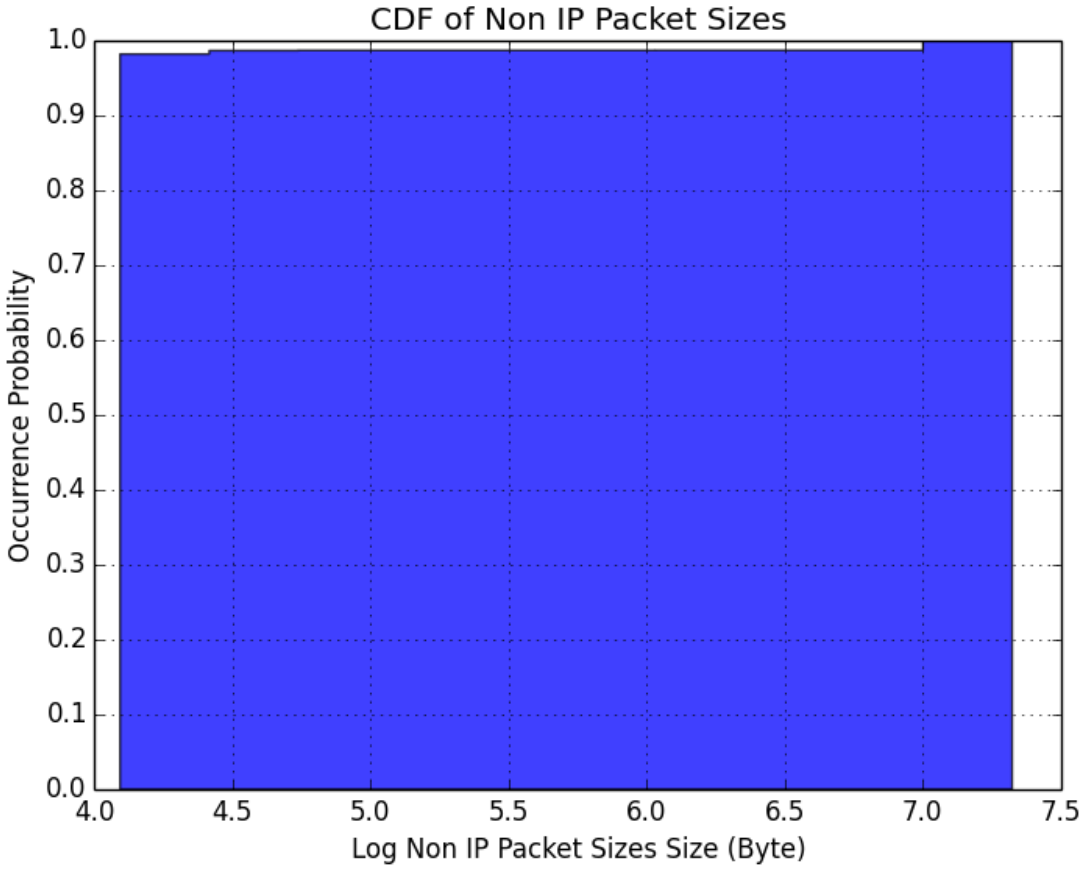
CDF of UDP packets



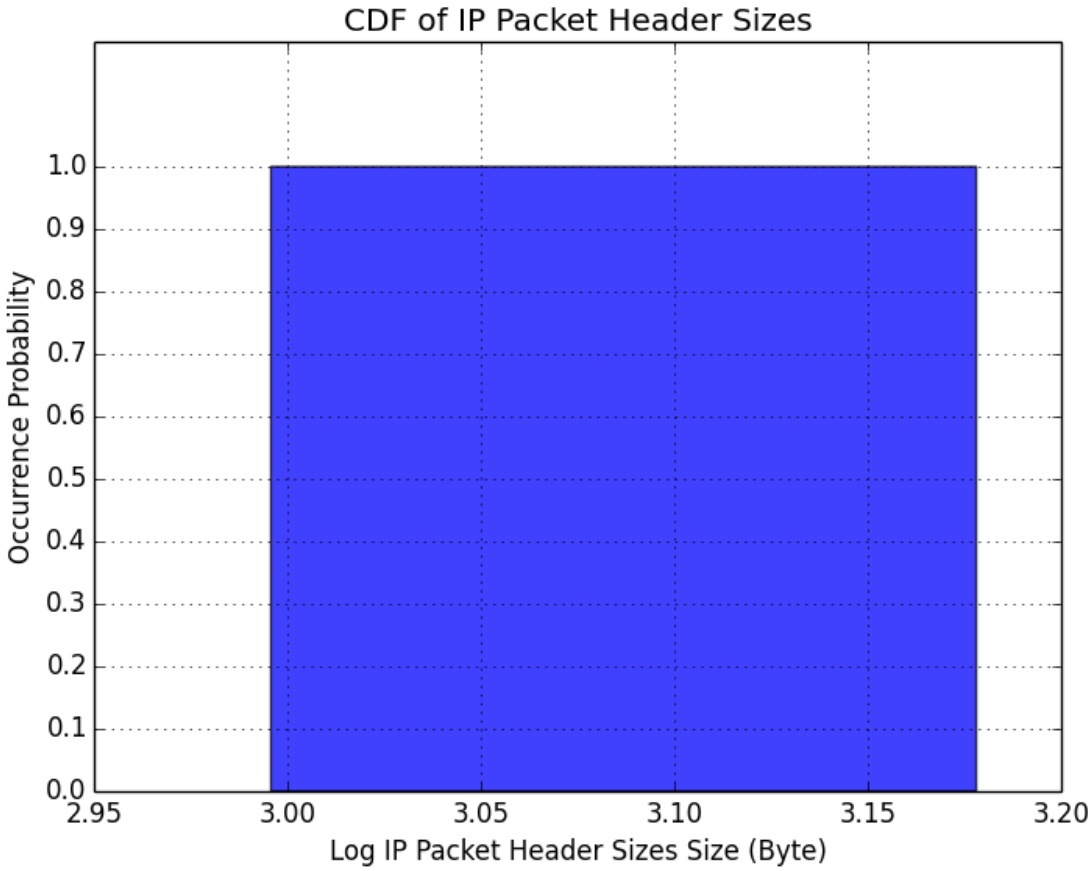
CDF of IP packets



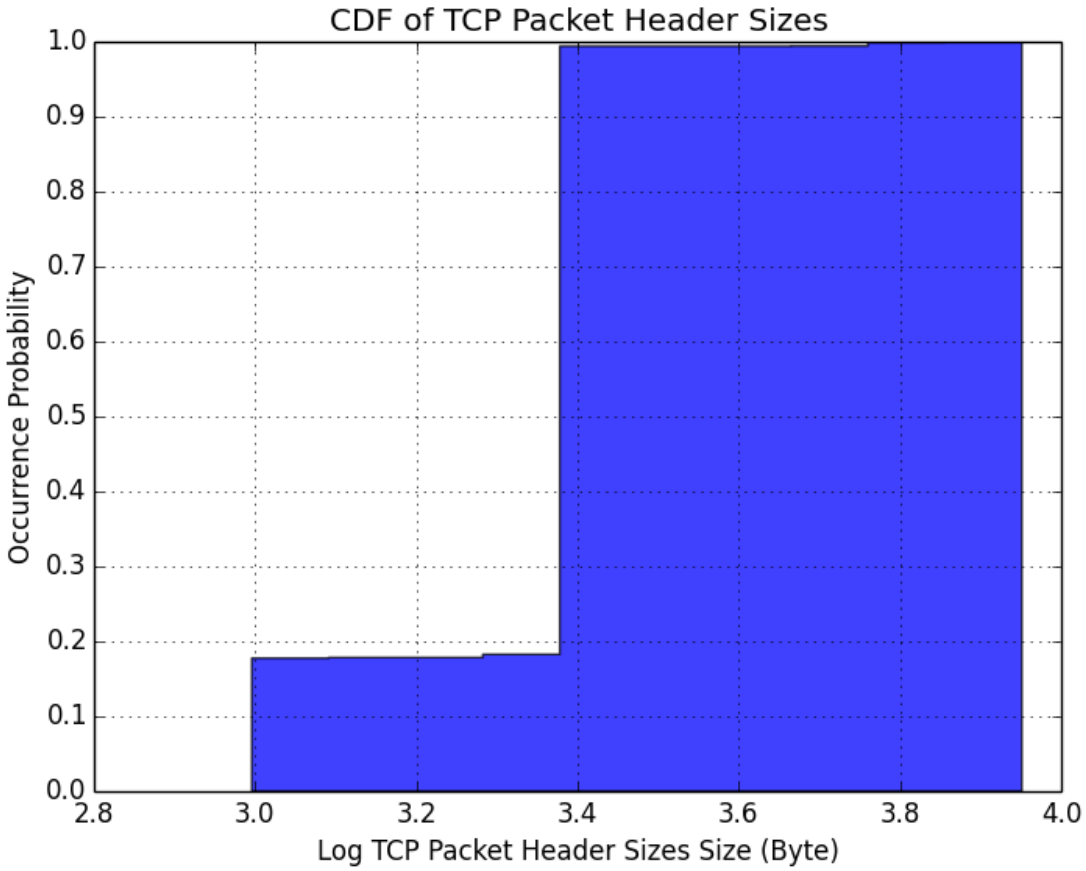
CDF of non-IP packets



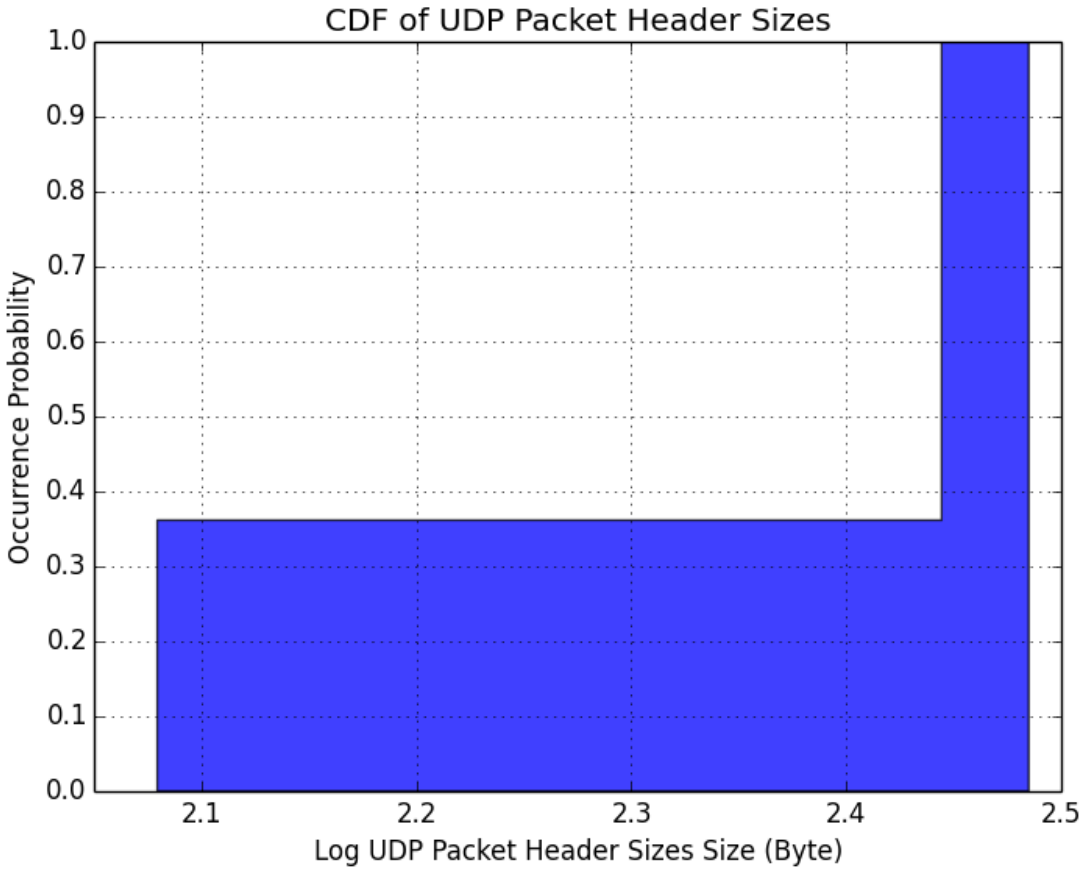
CDF of IP Packets Header Sizes



CDF of TCP Packets Header Sizes:



CDF of UDP packets Header Sizes



analysis:

As the pictures shown above, there's bigger variation of header size of TCP than that of UDP. The majority of UDP header size after logarithm is around 2.5. This is because UDP header generally has a fixed size while TCP's header size is varying from packet to packet.

why x-axis uses log:

Because there is no upper bound of the size of a packet, and we would like to shorten it into a period. Logarithmic function could do the job here. Also it could highlight the small difference between 0 and 1 and shorten large difference between 1 and infinity.

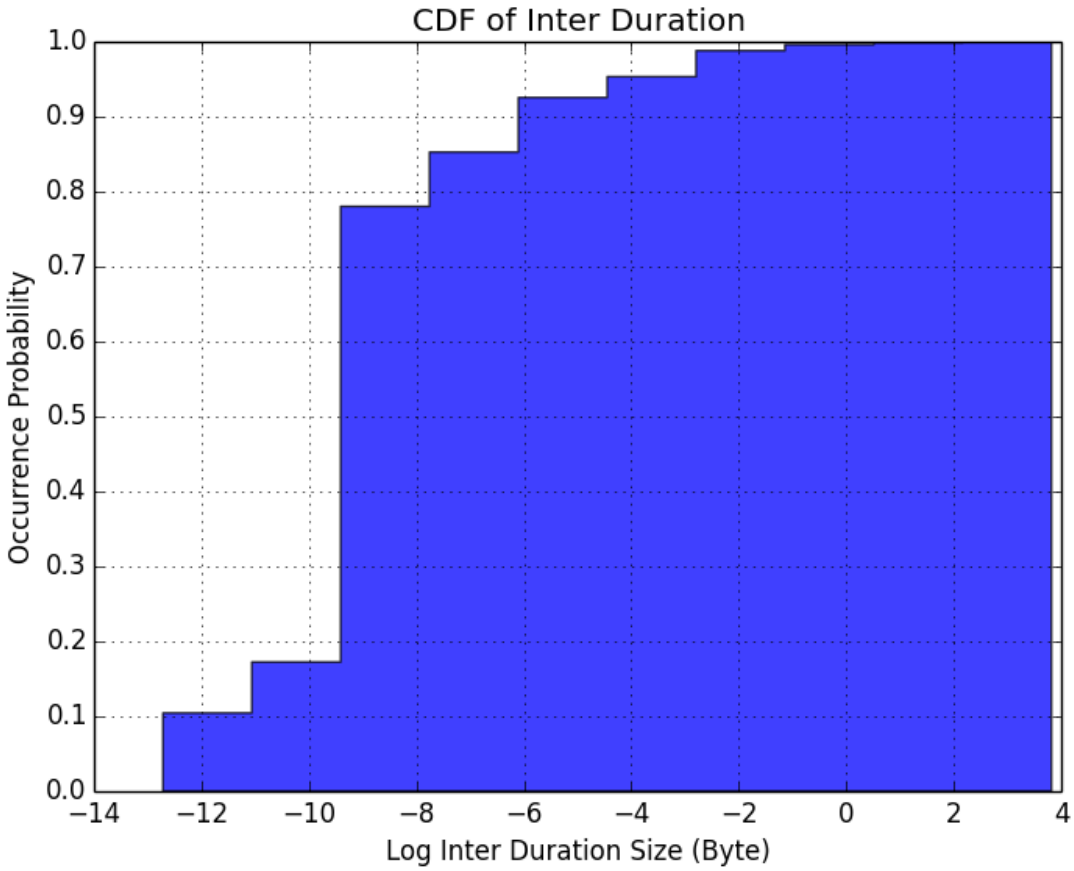
Flow Type:

Number of TCP flows	Number of UDP Flows
766	919

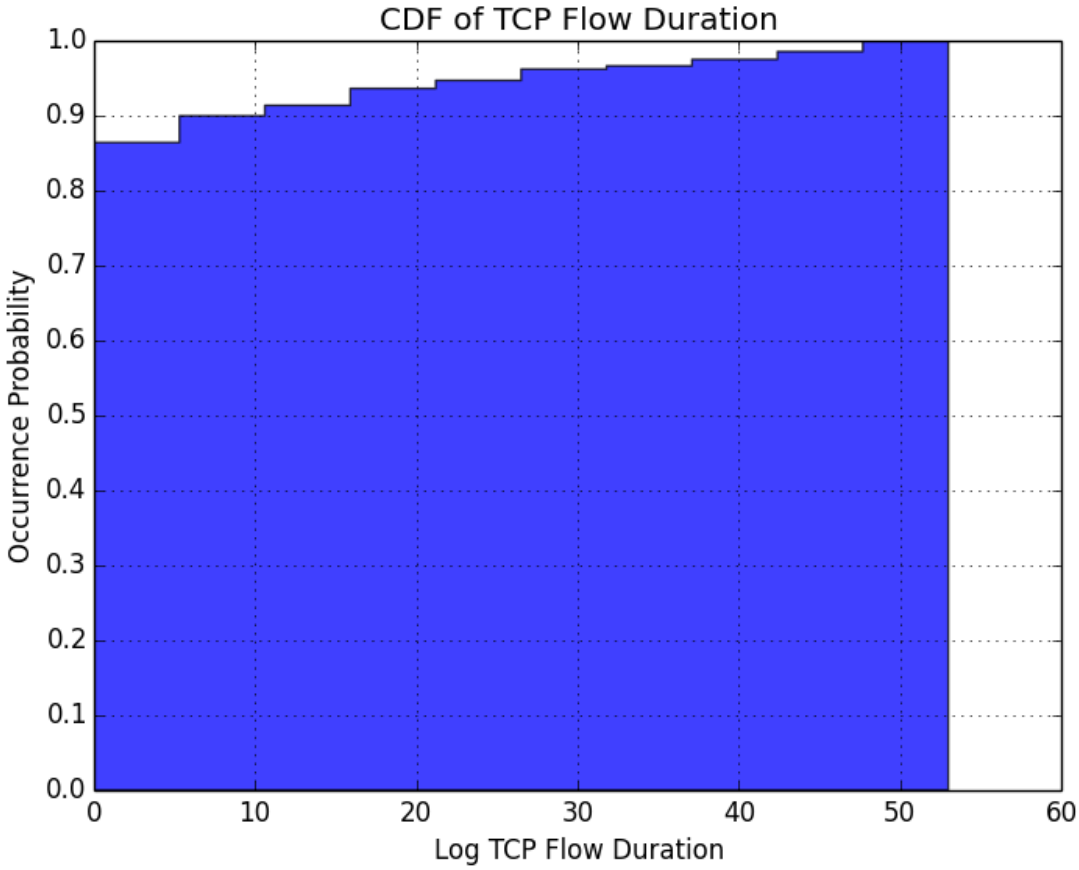
```
>>> keys = TCP_flows.keys()
>>> keys = list(keys)
>>> len(keys)
766

>>> udp_keys = UDP_flows.keys()
>>> udp_keys = list(udp_keys)
>>> len(udp_keys)
919
```

Flow Duration:

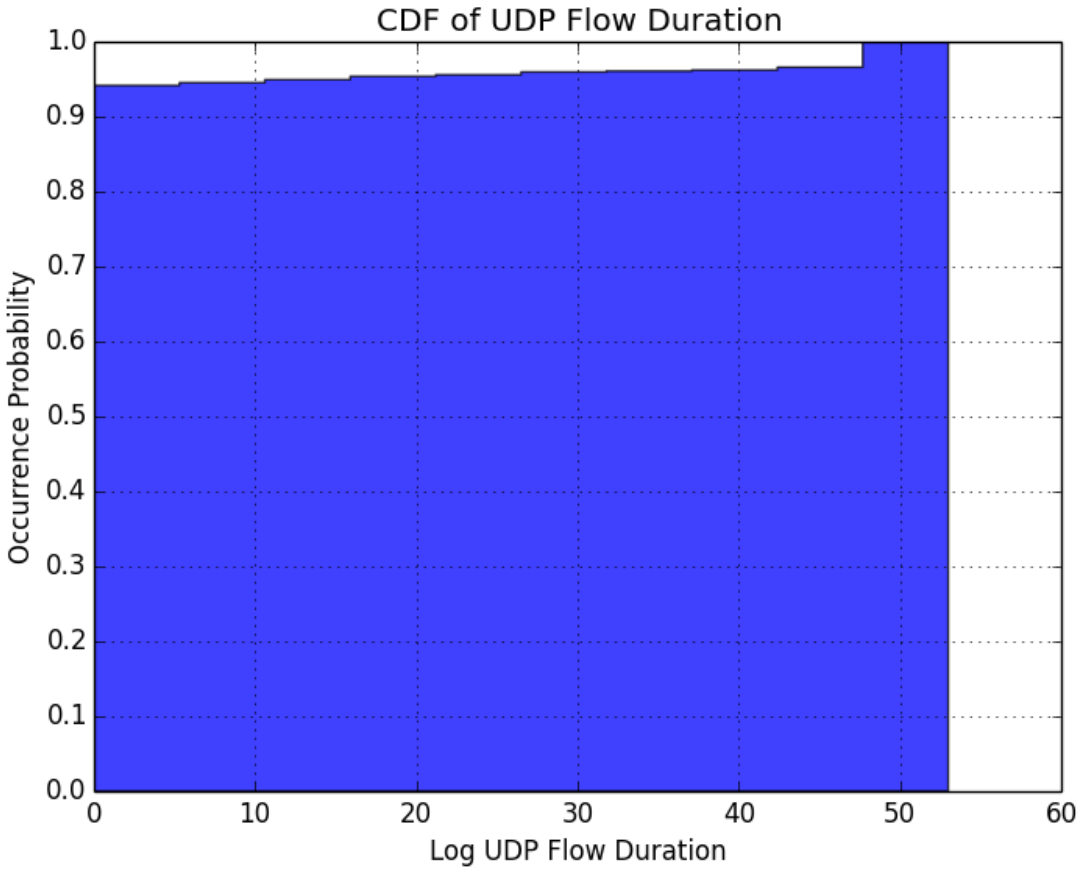
CDF of flow durations for all flows



CDF of flow durations of TCP flows:



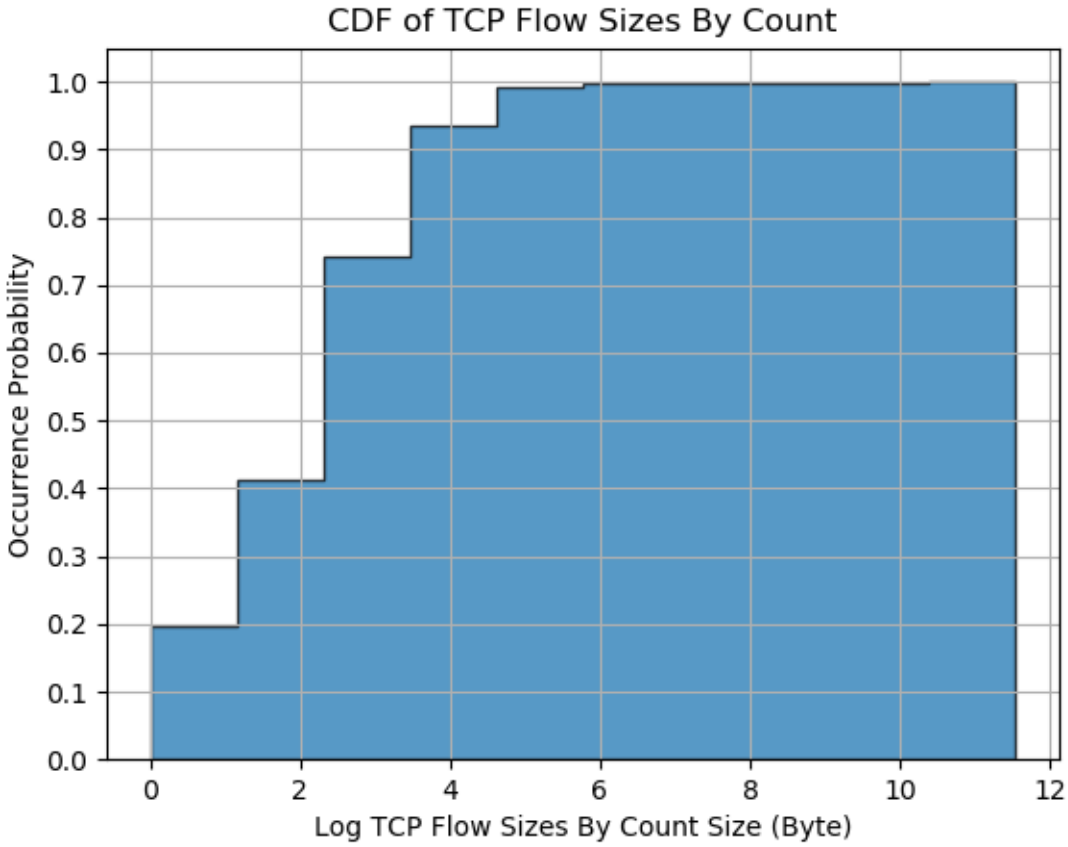
CDF of flow durations of UDP flows:

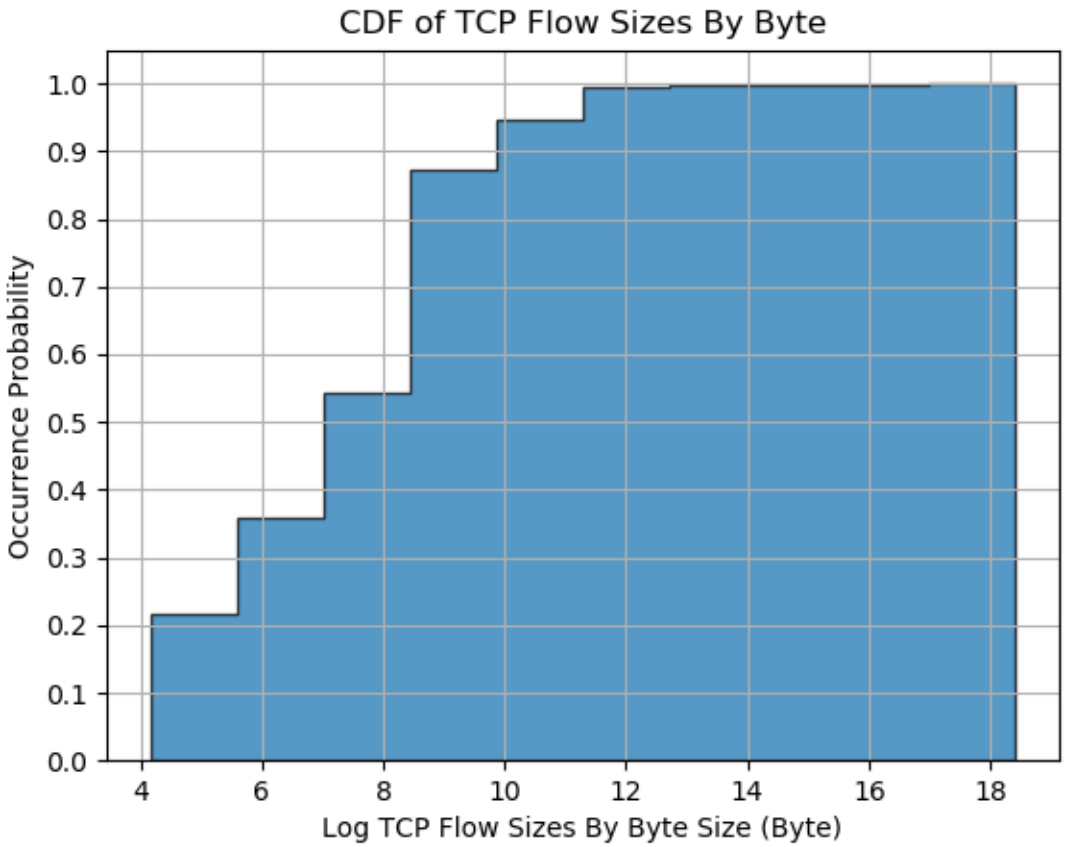


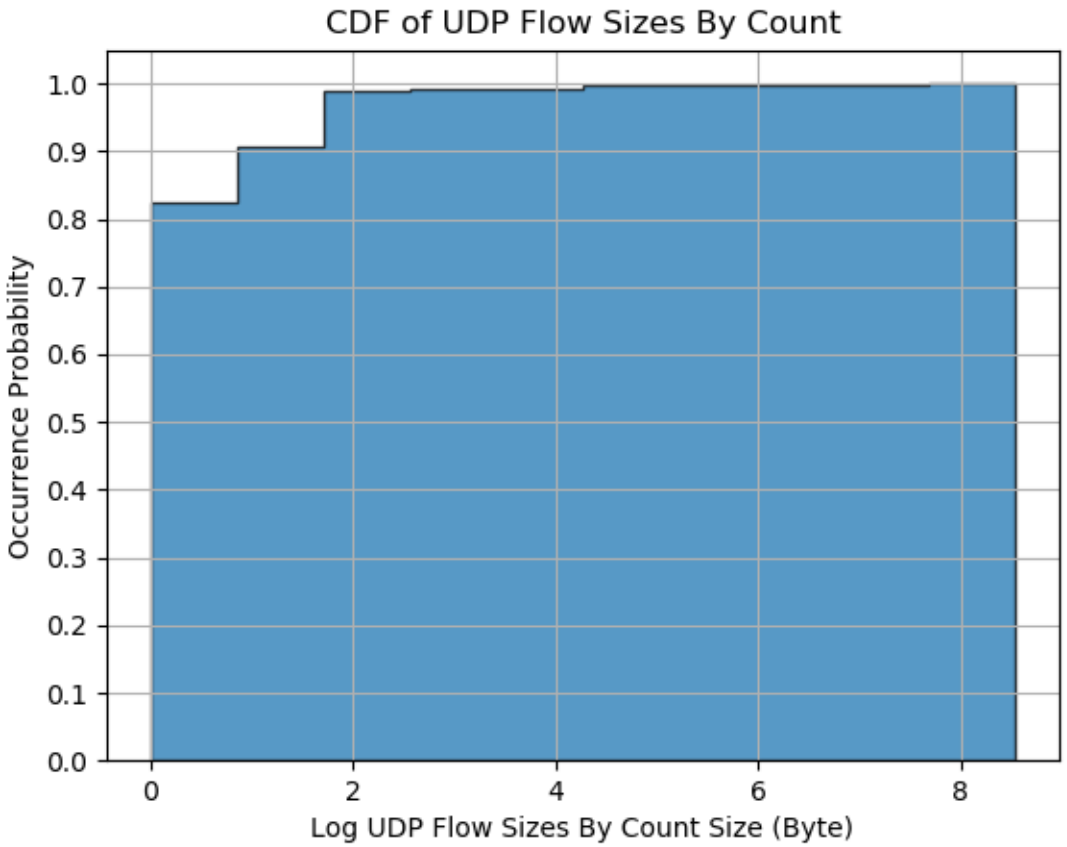
Is there any difference between TCP and UDP flows?

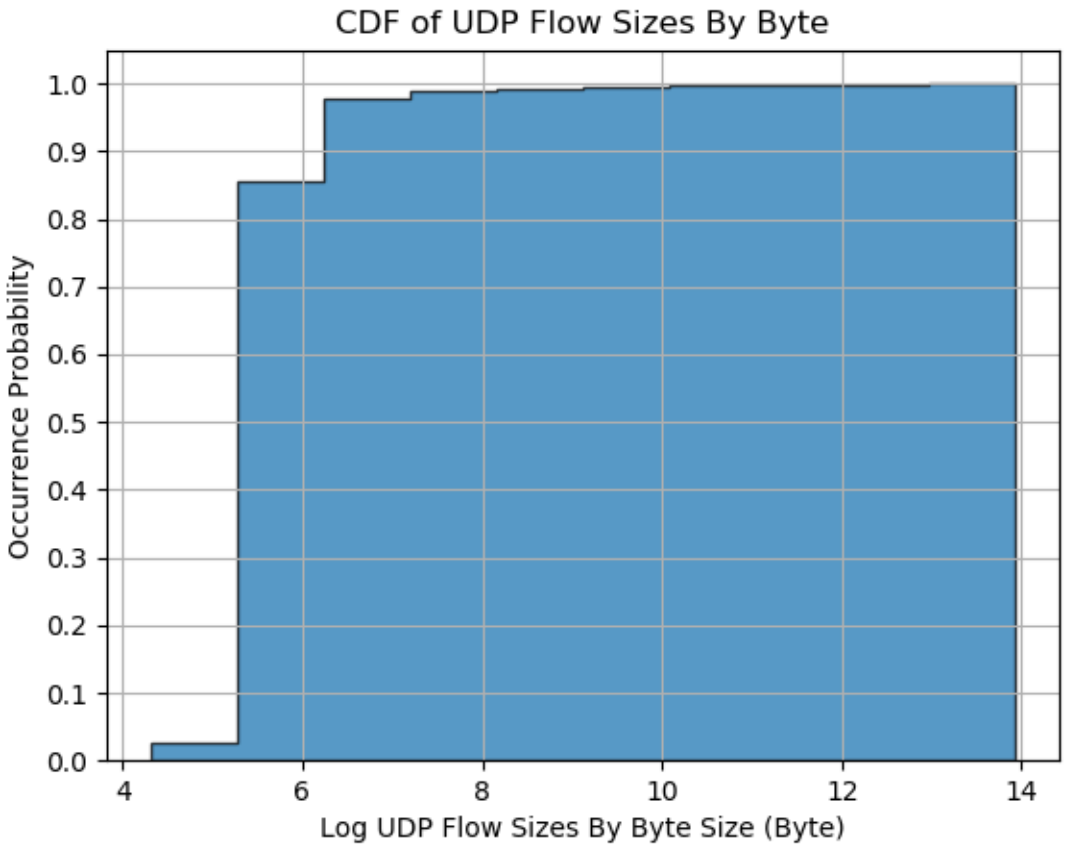
There is. The steps of increment of CDF of TCP Flow Durations are more obvious. This is because TCP implements retransmission, therefore there would be serveral packets of the same size, and the steps are more obvious, as for UDP, there is no restransmission mechanism, then the packet sizes tend to evenly distributed, we can see it by observing the more smooth increment.

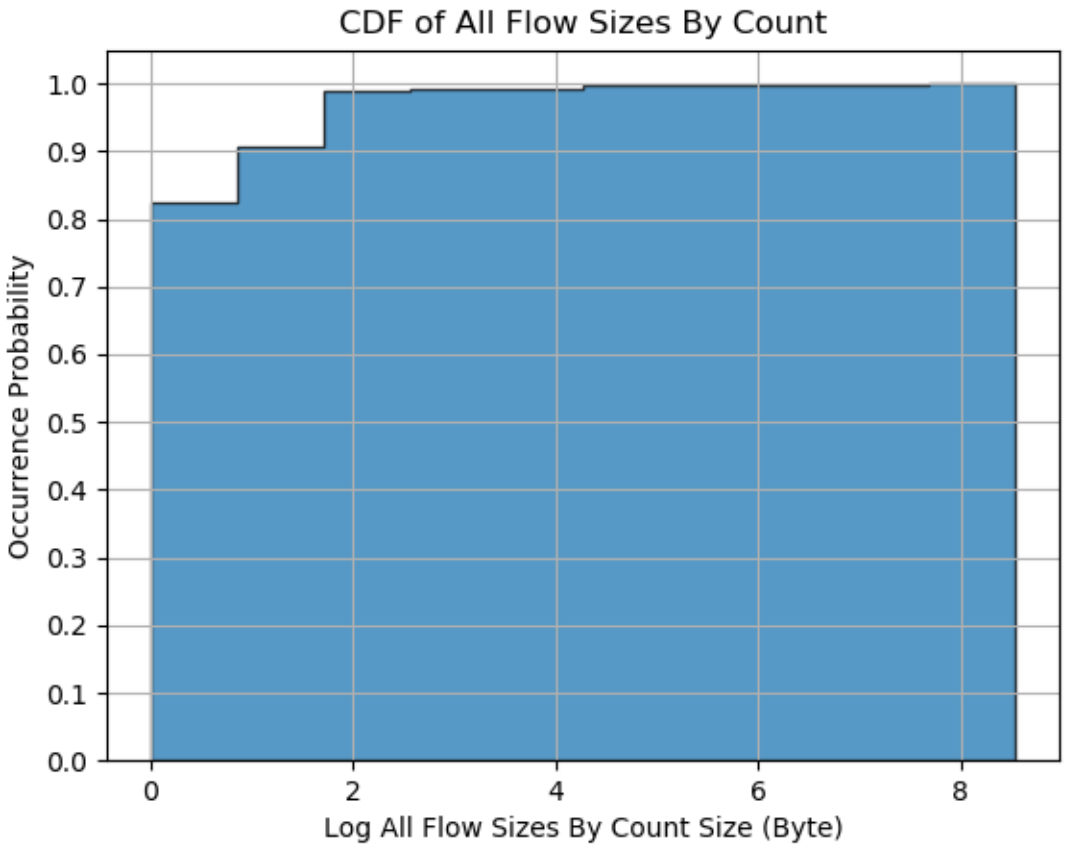
Flow Sizes

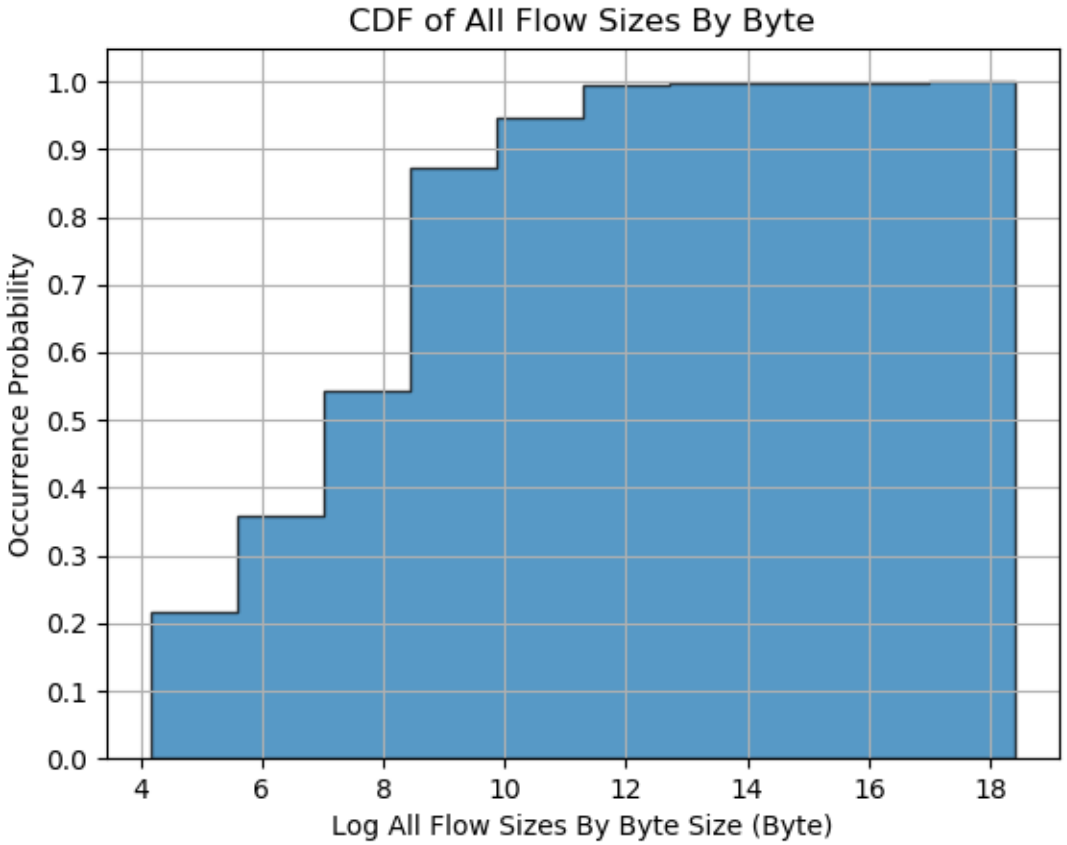

CDF of TCP Flow Sizes By Count
Occurrence Probability
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
2
4
6
8
10
12
Log TCP Flow Sizes By Count Size (Byte)


CDF of TCP Flow Sizes By Byte
Occurrence Probability
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
4
6
8
10
12
14
16
18
Log TCP Flow Sizes By Byte Size (Byte)


CDF of UDP Flow Sizes By Count
Occurrence Probability
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
2
4
6
8
Log UDP Flow Sizes By Count Size (Byte)


CDF of UDP Flow Sizes By Byte
Occurrence Probability
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
4
6
8
10
12
14
Log UDP Flow Sizes By Byte Size (Byte)


CDF of All Flow Sizes By Count
Occurrence Probability
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
2
4
6
8
Log All Flow Sizes By Count Size (Byte)



Do you see any difference between TCP and UDP?
What about the case of using packet count vs byte sum?

We noticed that TCP flows are of larger size. We think it is because TCP establish a connection, and UDP is a connectionless protocol. During a connection, content of larger sizes can be delivered and also TCP implements a retransmission mechanism, therefore both more packets and more bytes would be seen during tcp flows than during udp flows.

When using packet count, TCP packet count cdf is roughly the same as when using byte sum.

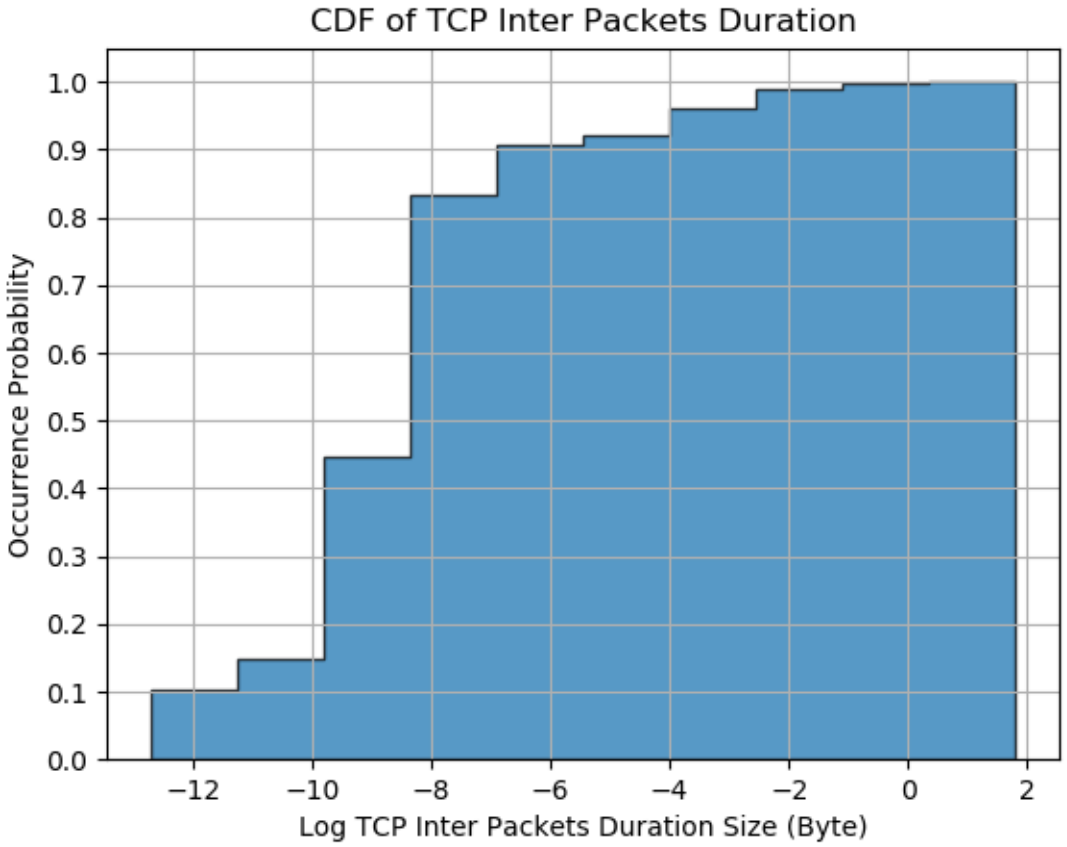
However, for udp, we can see that there are a lot of small packets. It can be seen from the small probability of packet count on X axis between 4 and 6 in CDF of UDP Flow Sizes by Byte.

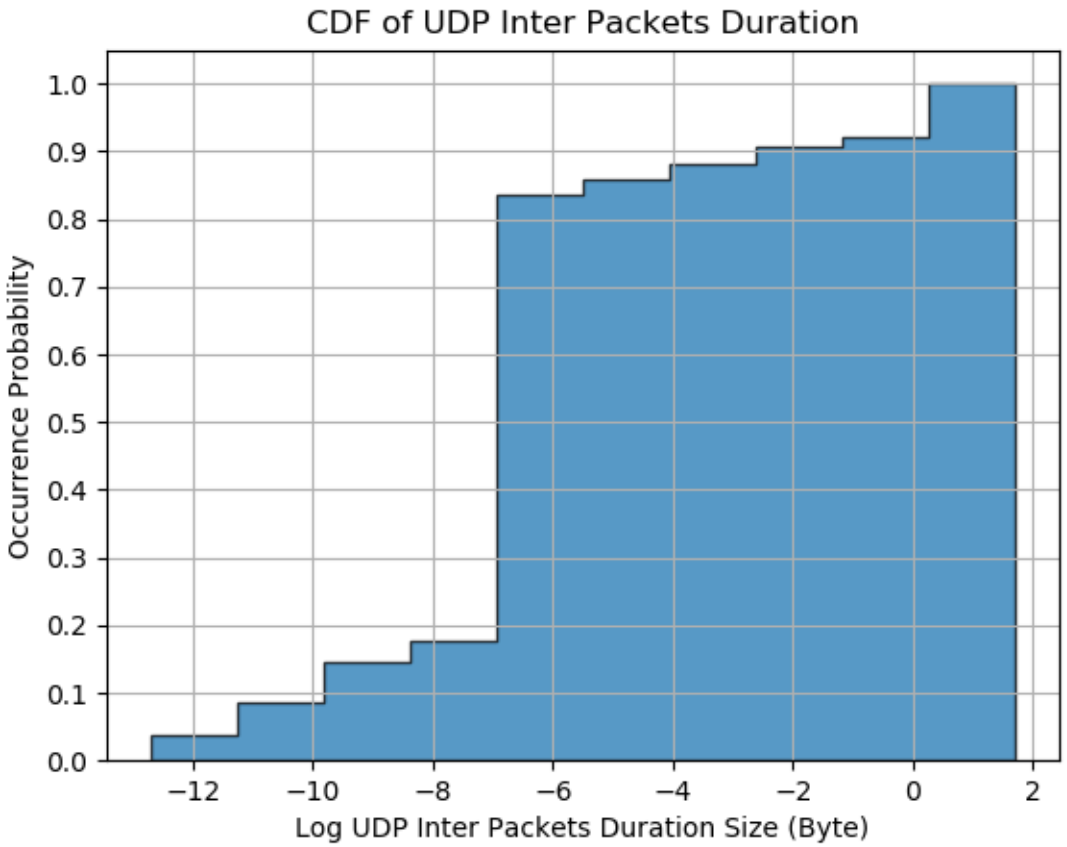
There is no similar thing in CDF of UDP Flow Sizes by Count

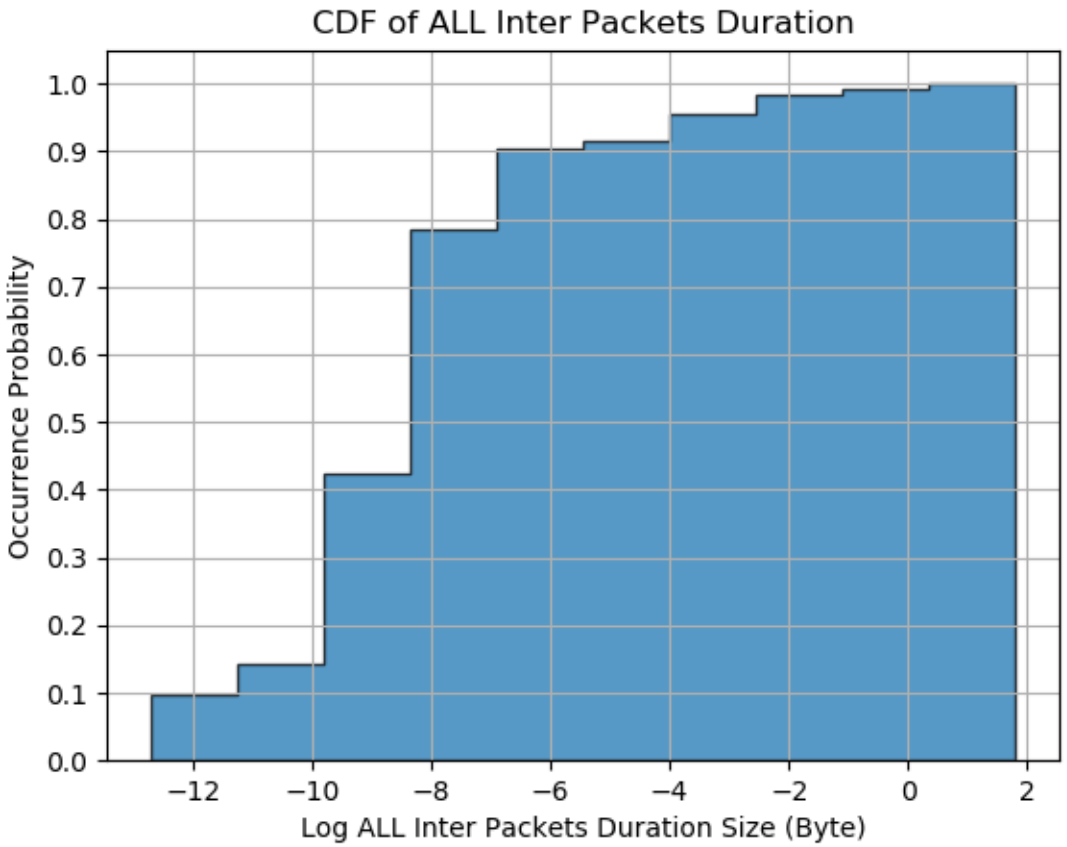
The overhead ratio as the sum of all headers (including TCP, IP, and Ethernet) divided by the total size of the data that is transferred by the flow:

0.07418569210819834

Inter-packet arrival time:




CDF of UDP Inter Packets Duration
Occurrence Probability
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
−12
−10
−8
−6
−4
−2
0
2
Log UDP Inter Packets Duration Size (Byte)



Is there any specific inter-arrival time that appears more commonly? If yes, is it present in all flows, TCP flows, or UDP flows? Do you see any difference between TCP and UDP flows?
Yes. We noticed that there is a huge increase around e^{-8}(X axis is using logarithmic).

It is present in all flows, TCP flows and UDP flows.We suspect this is because this pcap trace is done in a specific environment. That environment has a relatively stable latency and drop rate. Therefore a huge increae of a value less than 0.1 means that the network is relatively stable.

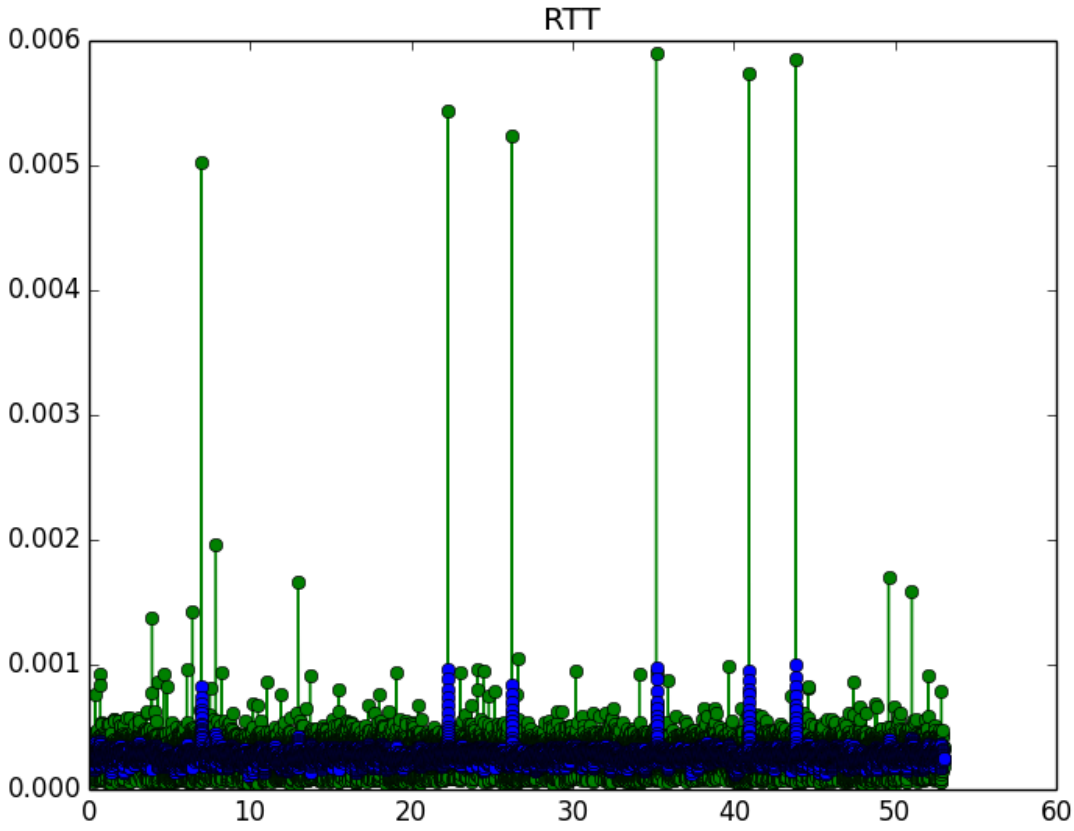
For this specific data set, the time interval between packets when using TCP is faster than that using UDP.

TCP State:
>>> print(get_tcp_state(TCP_flows))
(16, 164, 161, 425)

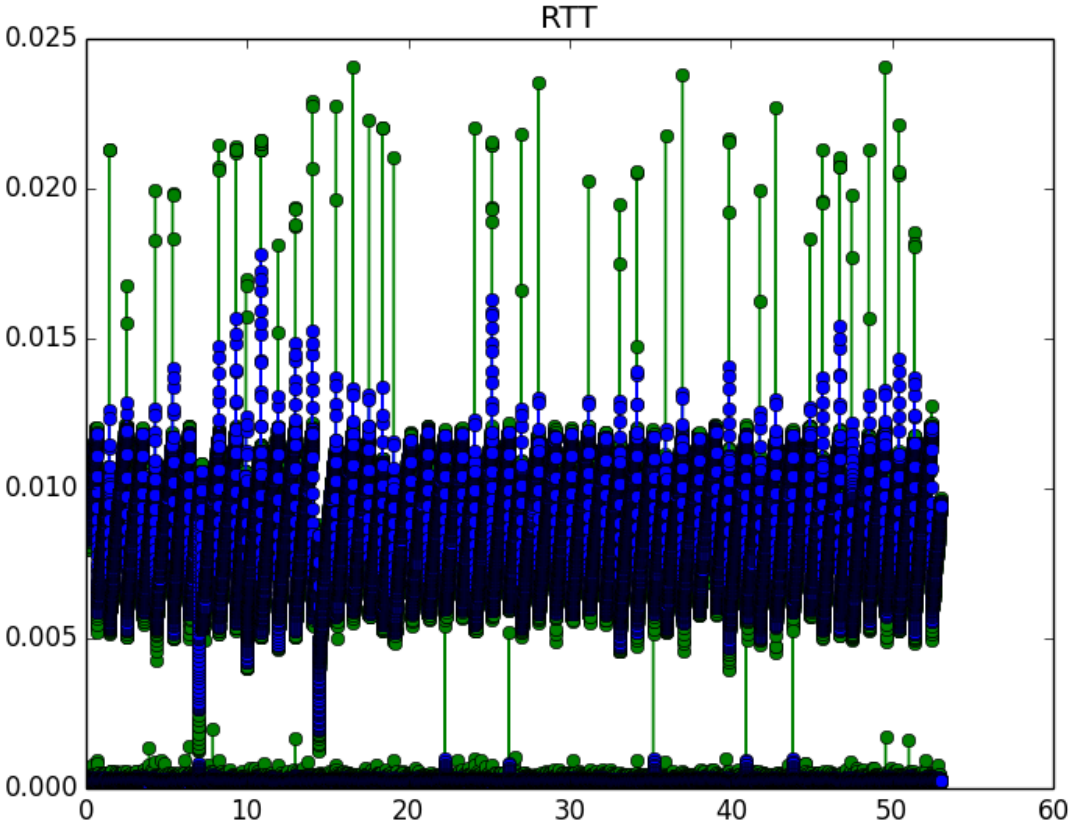
Number of Request	Number of Reset	Number of Finished	Number of Ongoing
16	164	161	425

Part 2
The top 3 longest TCP flows in terms of duration

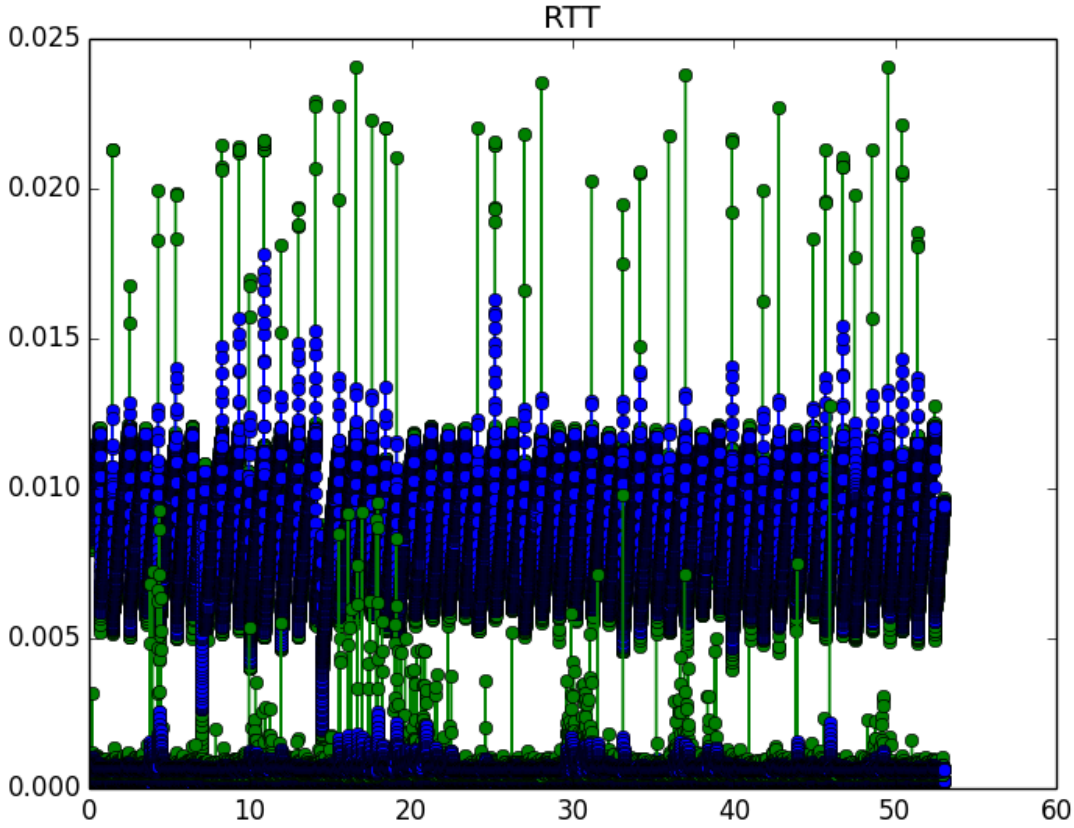
(Duration)First with one direction



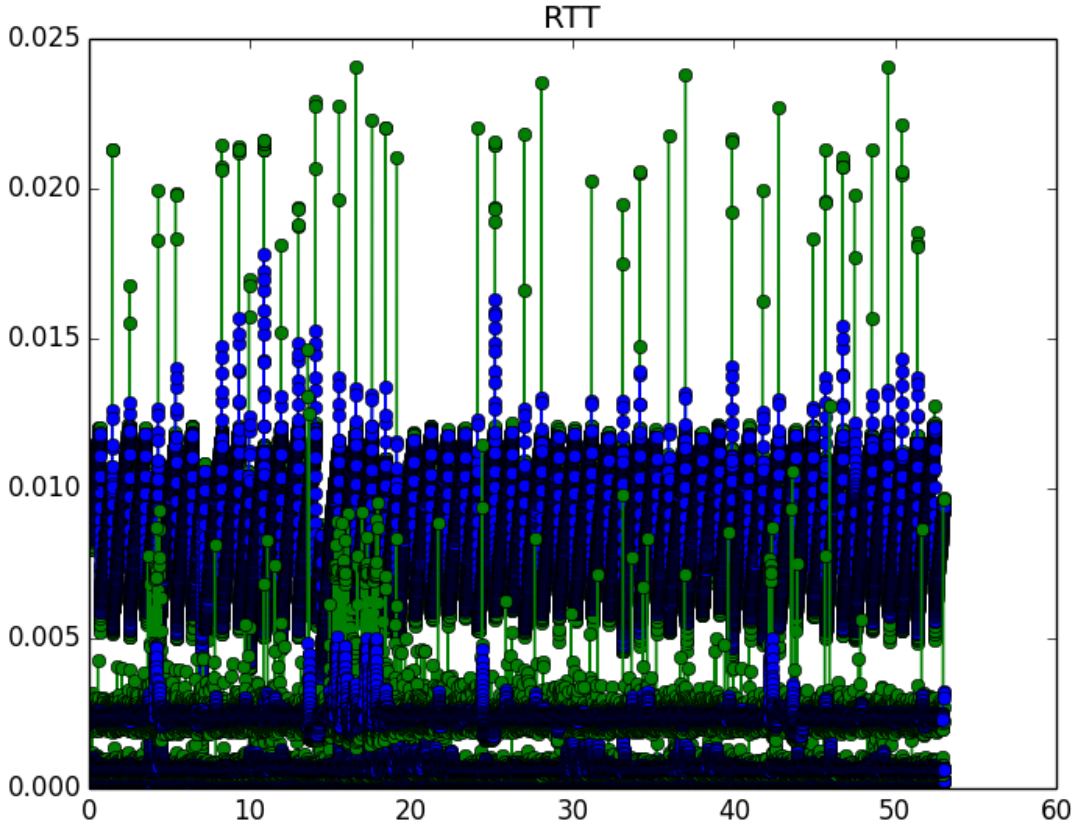
(Duration)First with the other direction



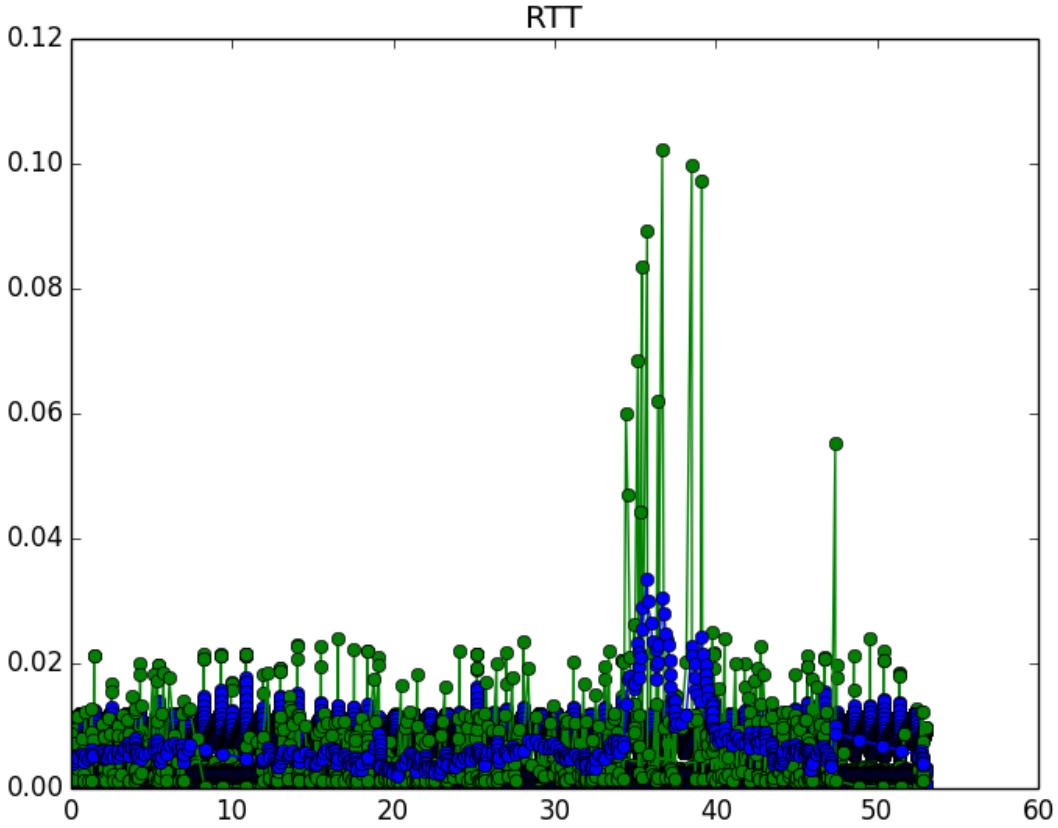
(Duration)Second with one direction



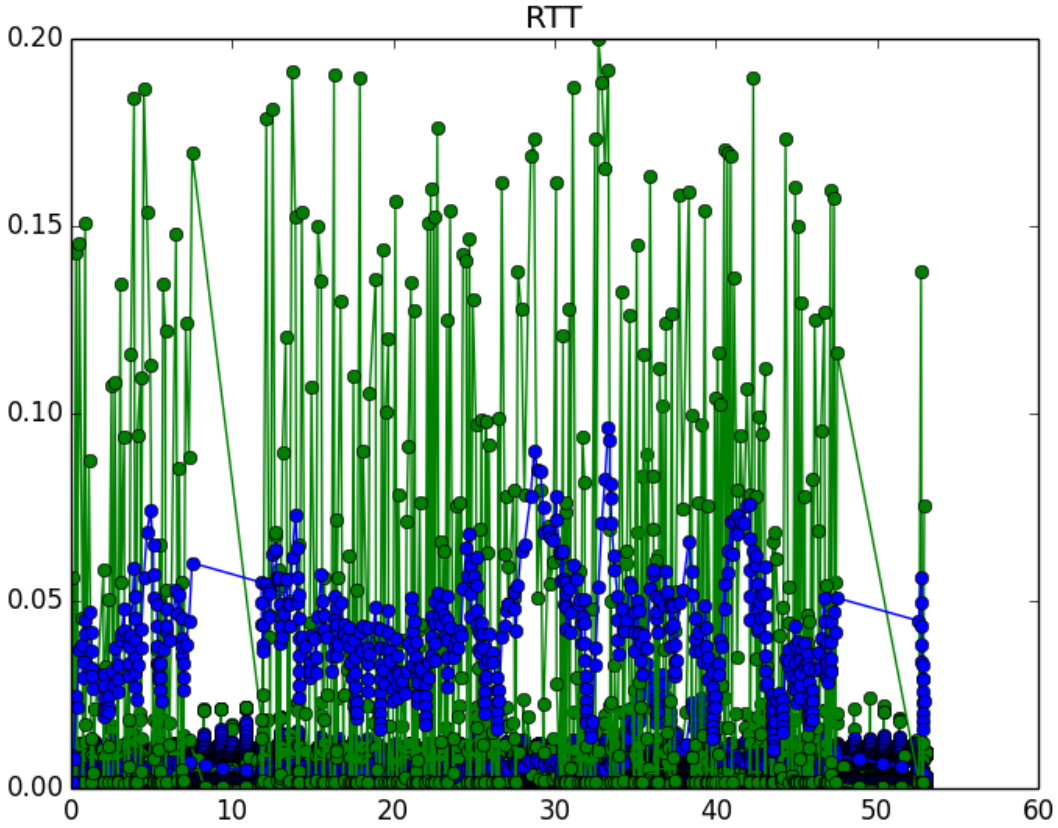
(Duration)Second with the other direction



(Duration)Third with one direction



(Duration)Third with the other direction



Is estimated RTT relatively stable? What about the sample RTTs? Do you see any increase or decrease in RTT?
If yes, what can be the reason that the RTT is changing during the life of a connection?

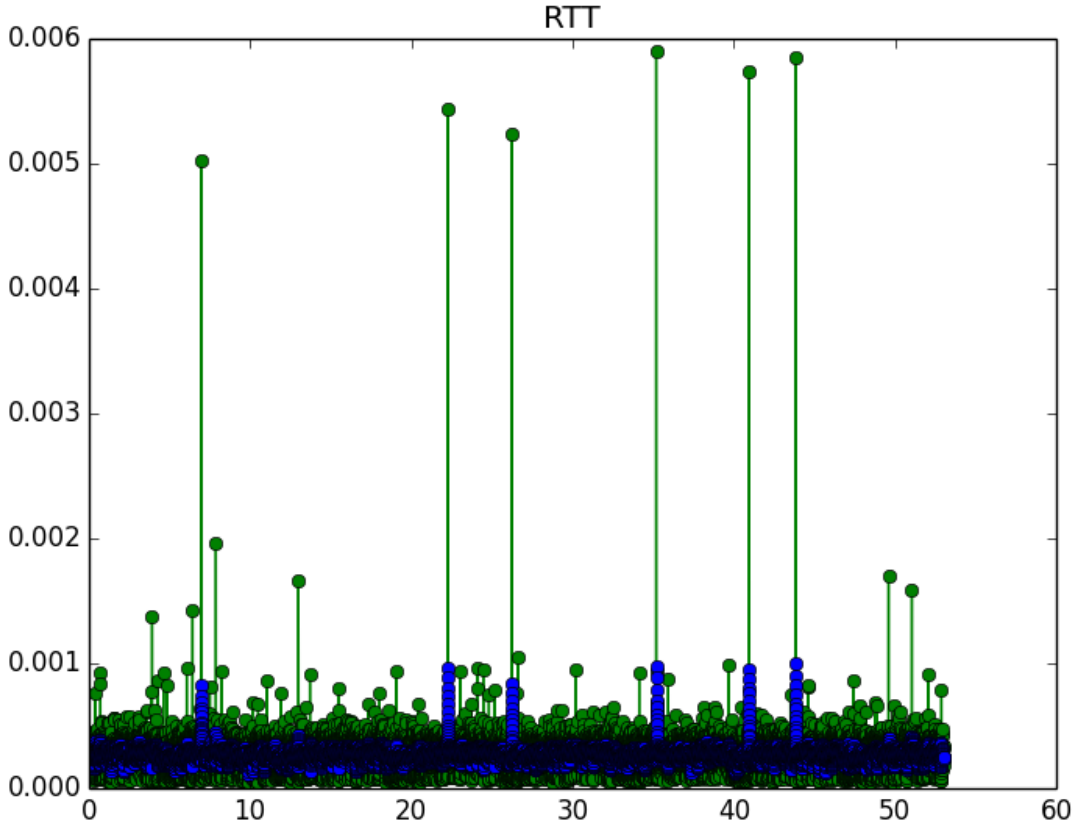
We can see that estimated RTT is relatively stable but not stable for the last one. There are sometimes some peaks that is drastically different from all other values. It's like a noise in Network communications.

On the other hand, sample RTTs are much more unstable than estimated RTTs

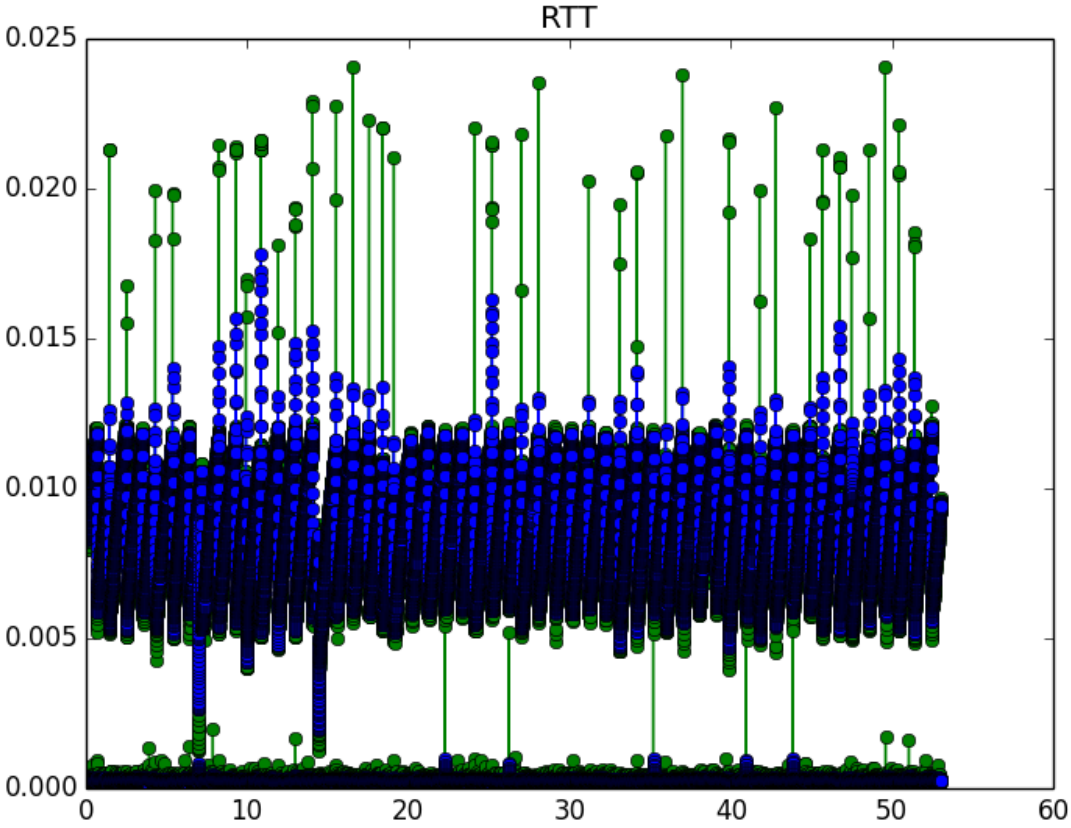
Yes. From the estimated RTT, we can clearly see the increase or decrease in RTT.

The Reasons could be related to network environment: for example, more people are using Ethernet now, you are walking away from a Wi-Fi router or ISP Internet service interruption

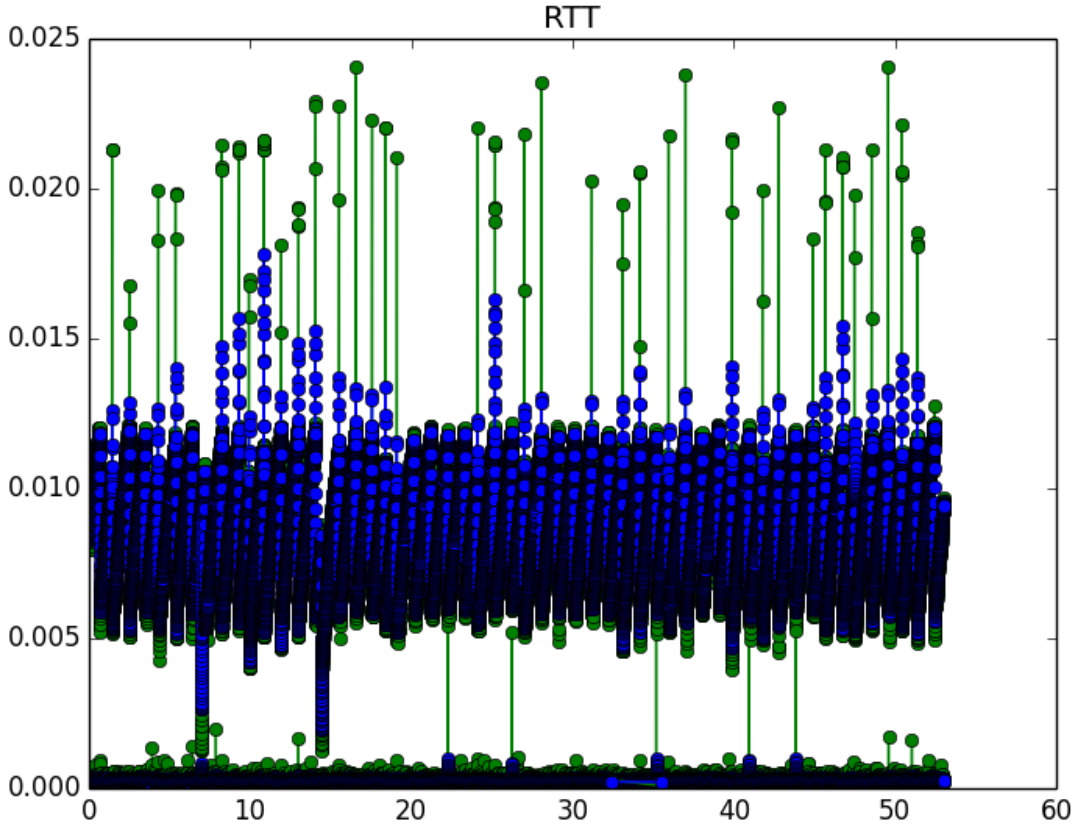
(Byte) First with one direction



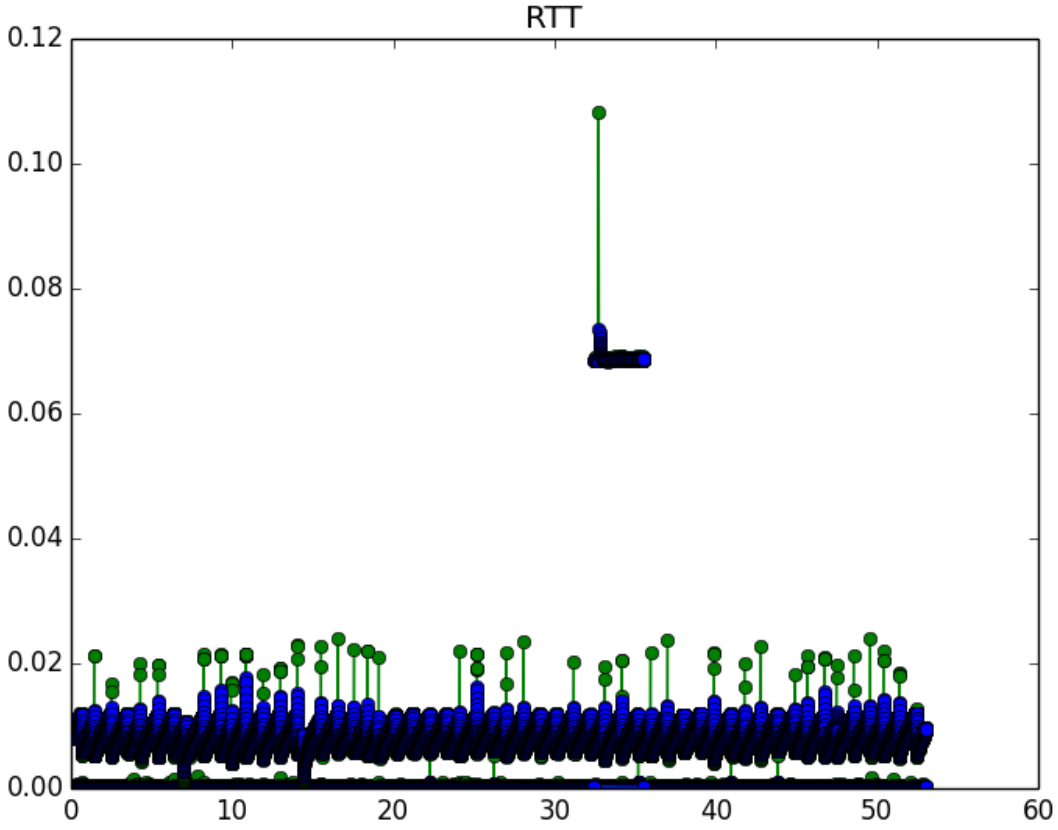
(Byte) First with the other direction



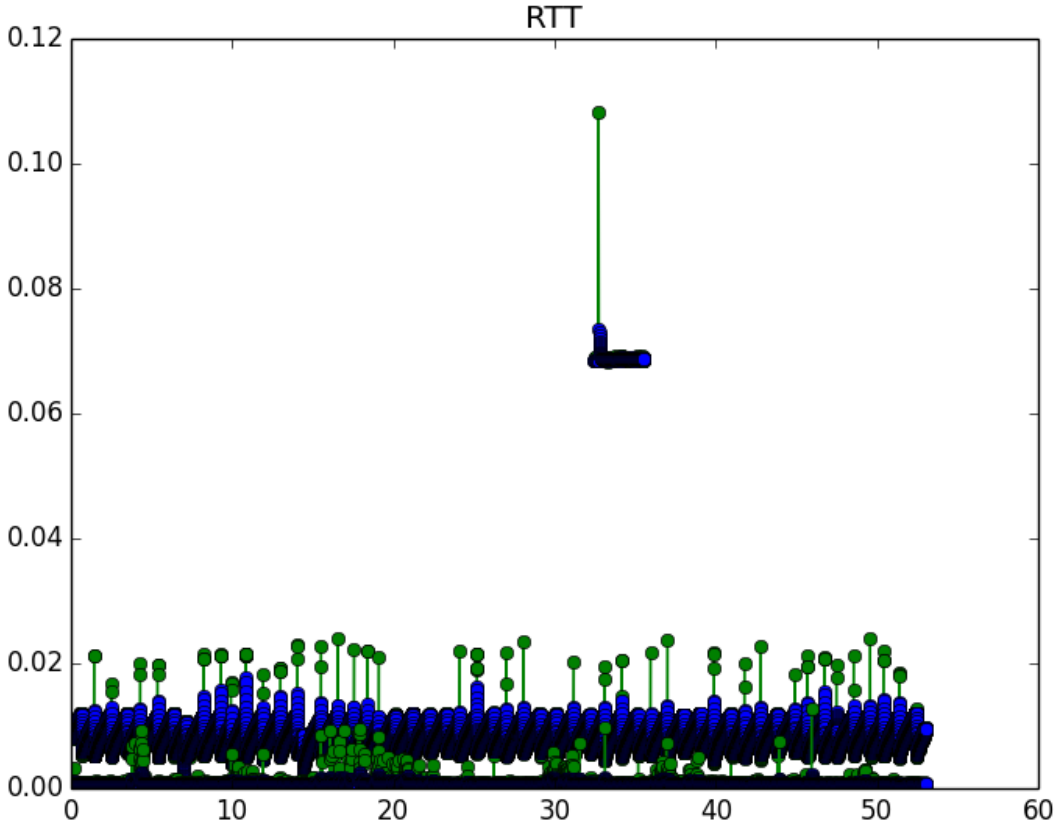
(Byte)Second with one direction



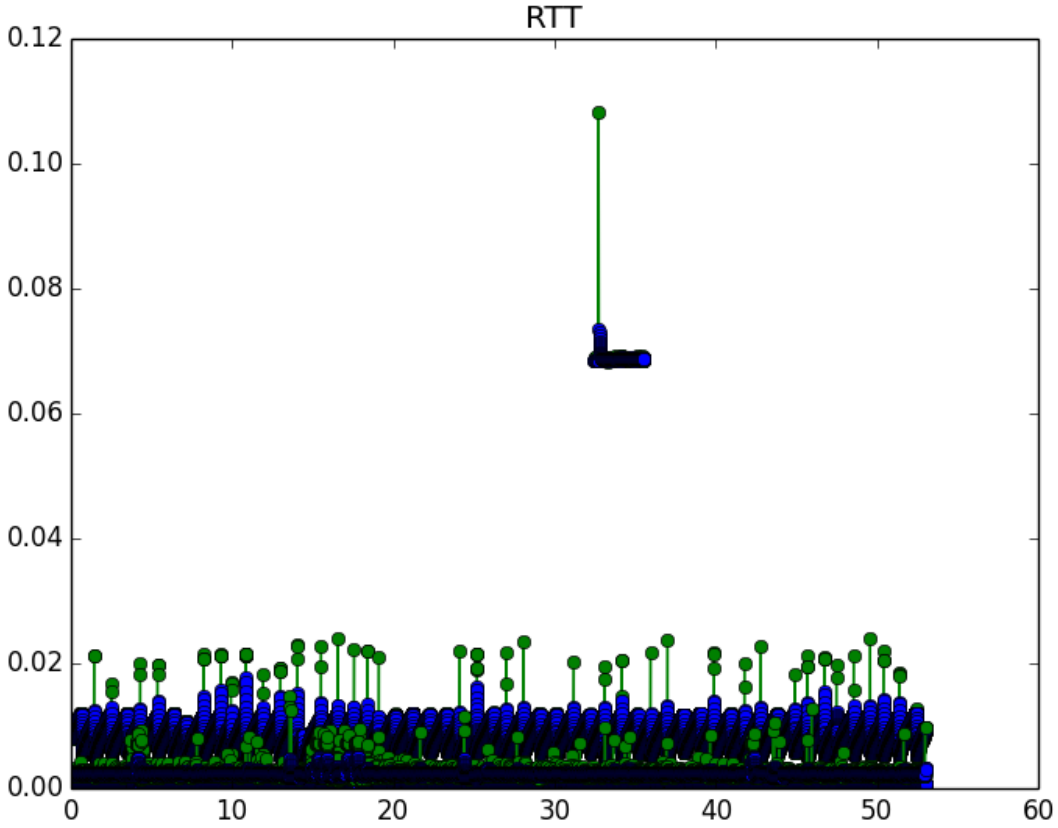
(Byte)Second with the other direction



(Byte)Third with one direction



(Byte)Third with the other direction



Is estimated RTT relatively stable? What about the sample RTTs? Do you see any increase or decrease in RTT?
If yes, what can be the reason that the RTT is changing during the life of a connection?

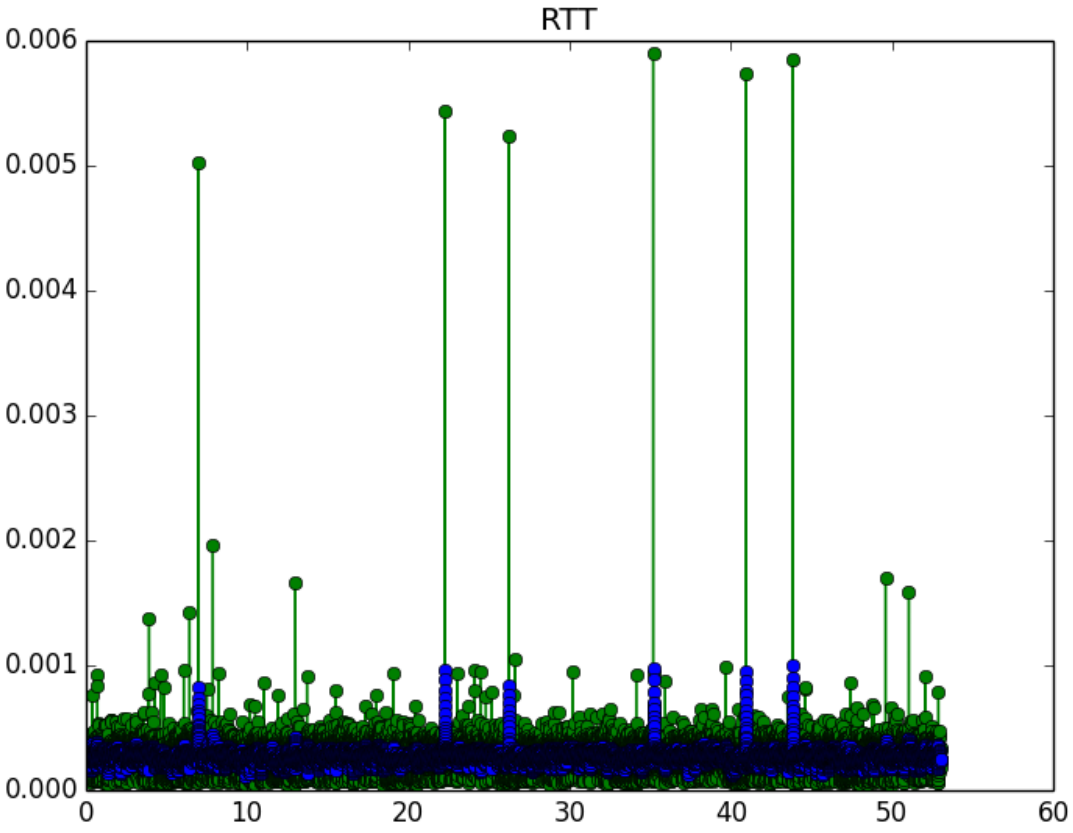
The estimated RTTs are also relatively stable. There are sometimes some peaks that are drastically different from all other values. It's like a noise in Network communications.

On the other hand, sample RTTs are much more unstable than estimated RTTs

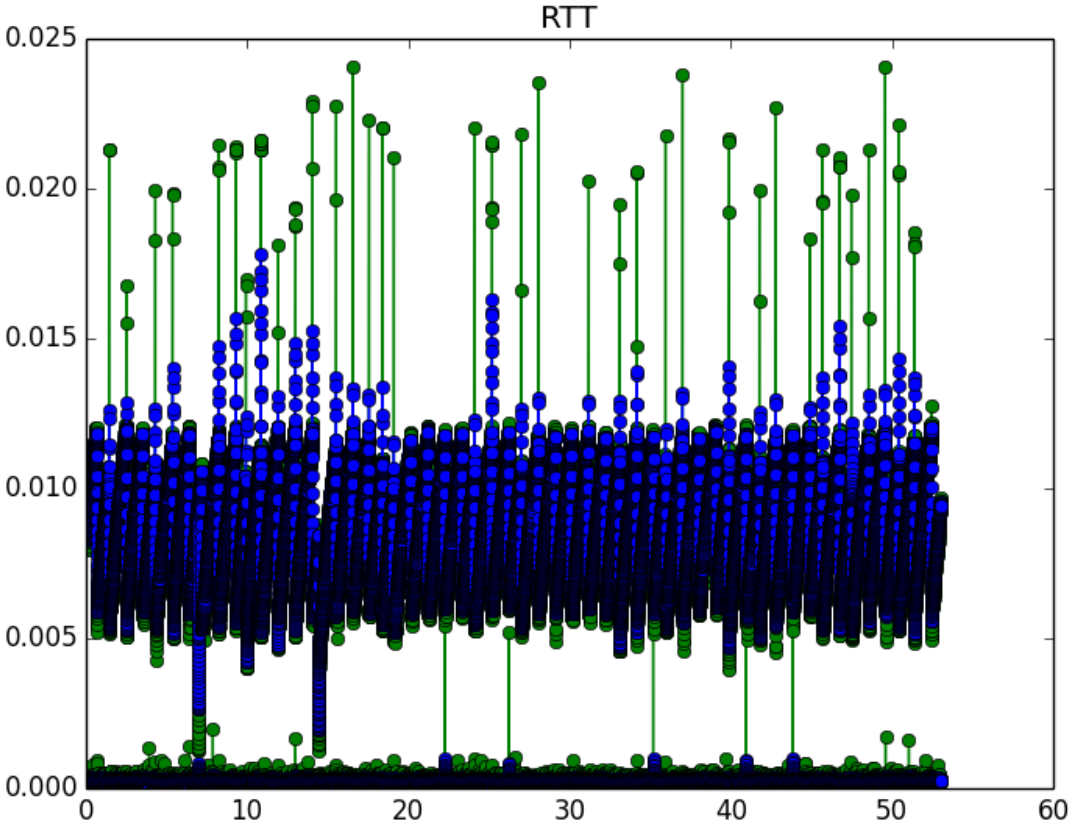
Yes. From the estimated RTT, we can clearly see the increase or decrease in RTT.

The Reasons could be related to network environment: for example, more people are using Ethernet now, you are walking away from a Wi-Fi router or ISP Internet service interruption

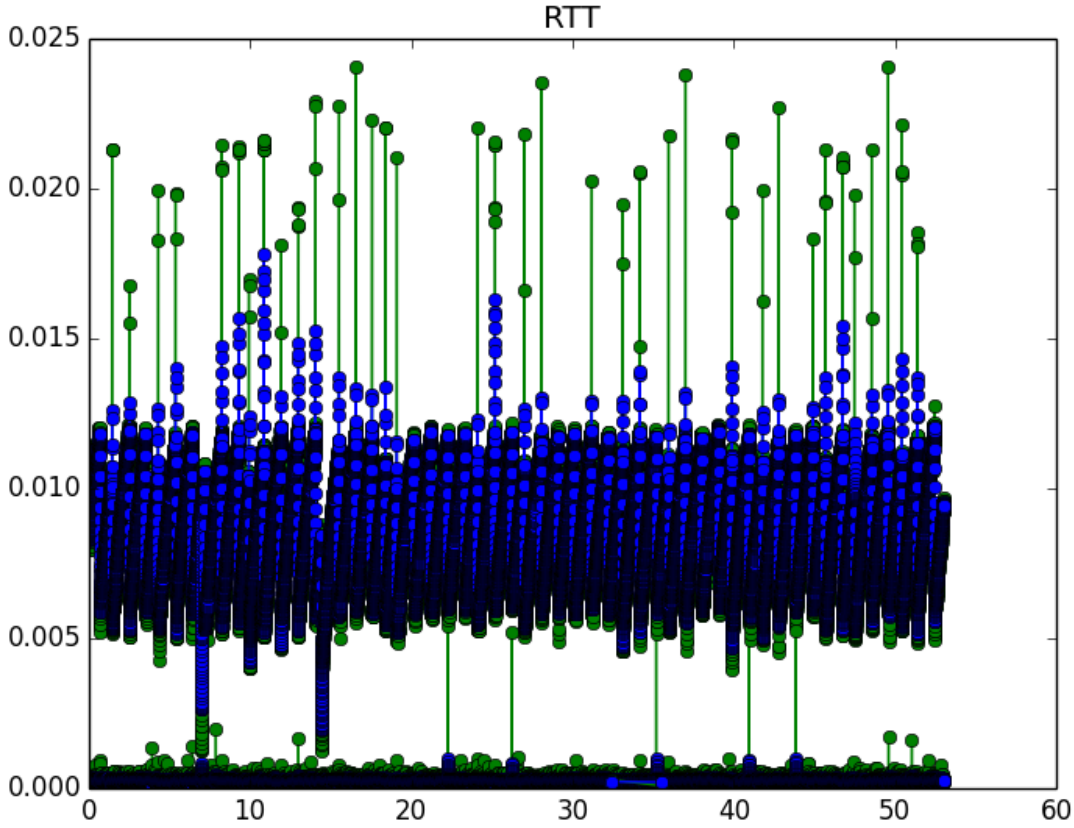
(Count)First with one direction



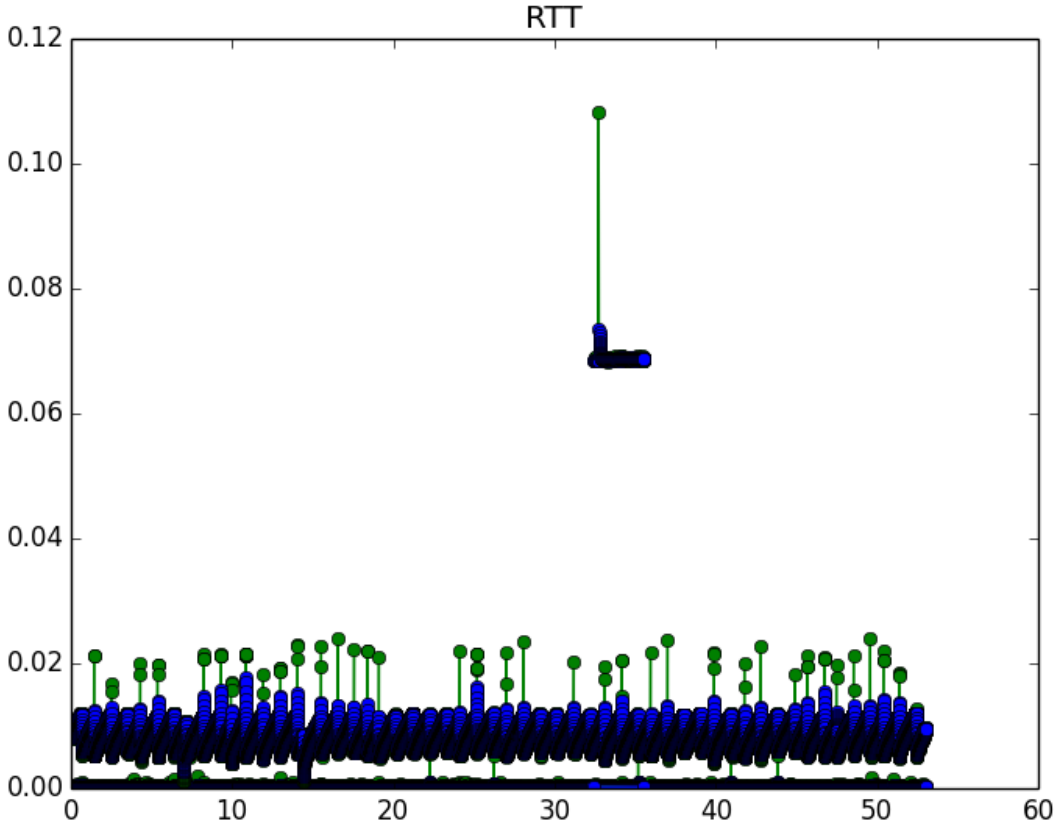
(Count)First with the other direction



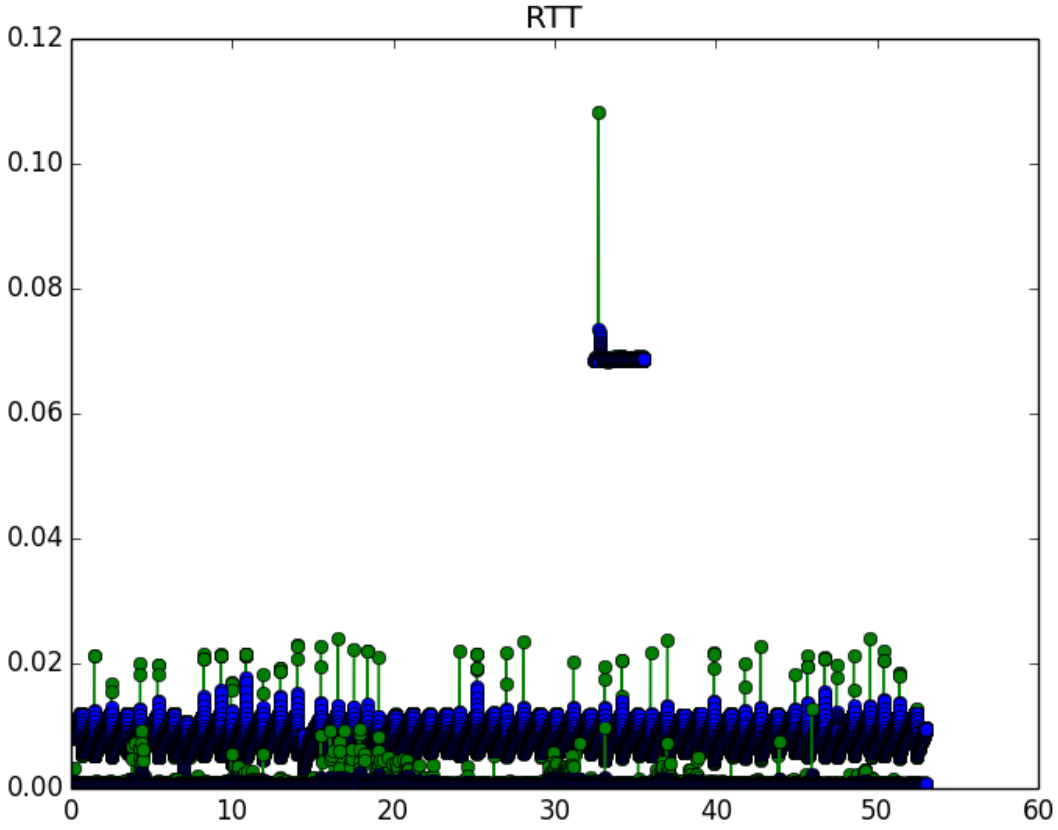
(Count)Second with one direction



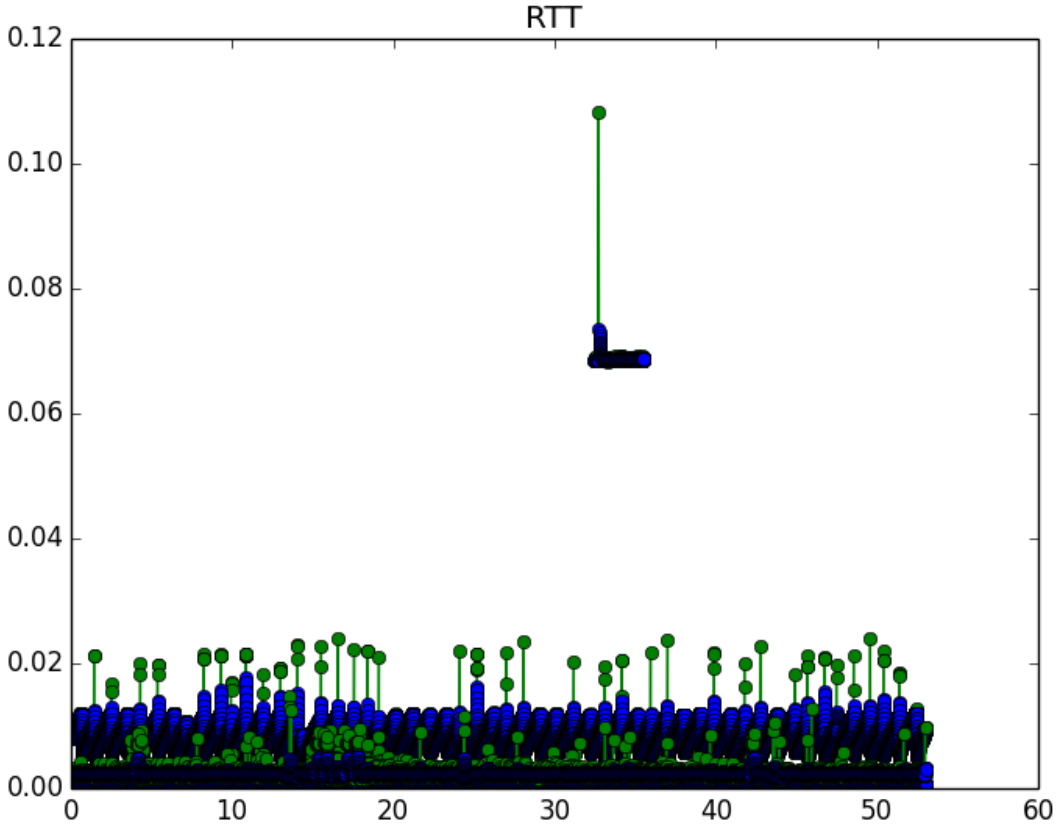
(Count) Second with the other direction



(Count)Third with one direction



(Count)Third with the other direction



Is estimated RTT relatively stable? What about the sample RTTs? Do you see any increase or decrease in RTT?
If yes, what can be the reason that the RTT is changing during the life of a connection?

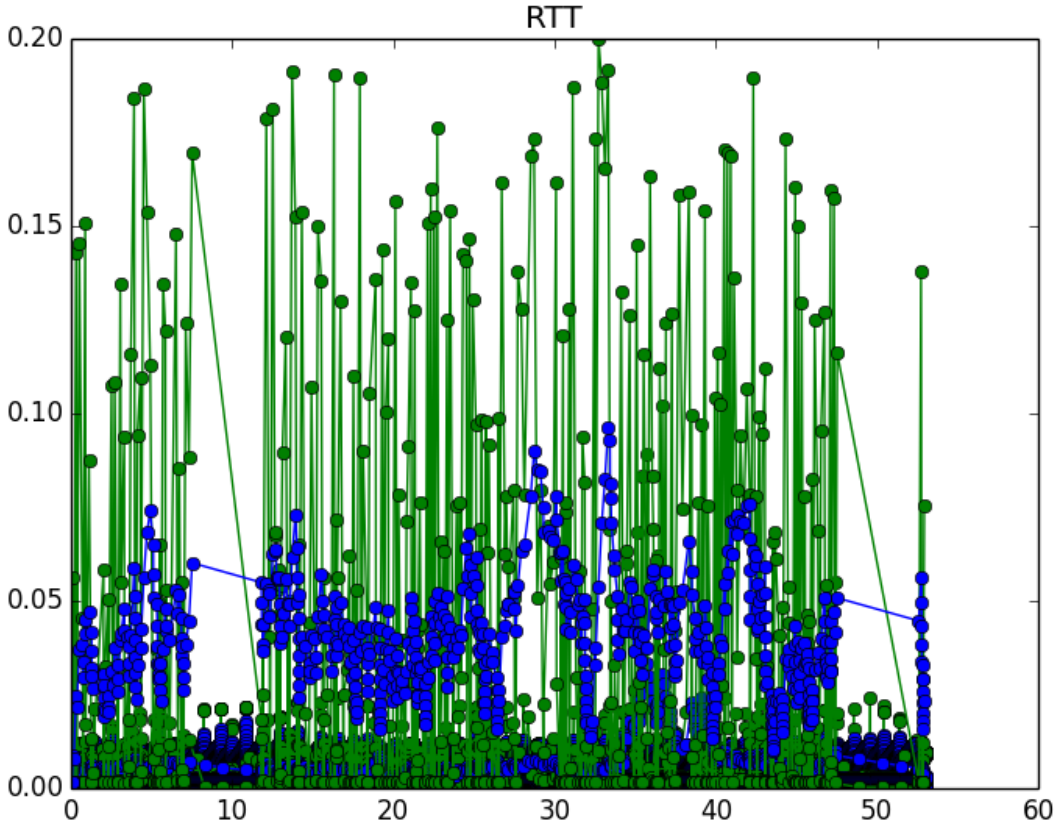
The estimated RTTs are also relatively stable. There are sometimes some peaks that are drastically different from all other values. It's like a noise in Network communications.

On the other hand, sample RTTs are much more unstable than estimated RTTs

Yes. From the estimated RTT, we can clearly see the increase or decrease in RTT.

The Reasons could be related to network environment: for example, more people are using Ethernet now, you are walking away from a Wi-Fi router or ISP Internet service interruption

Representative median RTT change with time plot



Do you see any specific pattern? Is the representative RTT changes over time? Is this change random or is it following a specific patter (e.g., first increase, and then decrease)? Explain what could be the reason of this change. Also compare the 3 different host pairs together. Is there any difference between them? If yes, what could be the reason.

No, median RTTs has no relationship with time, therefore it looks much like a noise pattern.

Yes, we think the representative RTT changes over time. We think it's random.

Reason for the random changes could be network instability.

We think there is some small difference between three hosts. It may be due to the difference in network chips or means to connect to Internet or simply because the ability/bandwidth to transfer data.