Overview.

All applications and commands, such as Wireshark and other command line tools, will be run on a Mac laptop. In addition, some general tools were set up to aid the data collection phase of this report. An iPhone was set up using `rvictl` and connected to a Mac laptop such that packets could be remotely captured on the mobile device and send to the laptop for analysis.

A shortcut was created on the command line using the command `ln -s /System/Library/PrivateFrameworks/Apple80211.framework/Resources/airport /usr/local/bin/airport` which would allow for the command `airport` to be used. Any instances of the `airport` command refers to this section.

Task One.

1.

The goal for Section 1.1 is to capture packets while streaming video from various video streaming services. The four services used in this report are YouTube, DailyMotion, Vimeo, and Odysee.

	Persistent/Non-persistent TCP	**Number of TCP connections**	**Number of packets captured in 60 seconds**
YouTube	Persistent	163	14711
DailyMotion	Persistent	374	85749
Vimeo	Persistent	83	15329
Odysee	Persistent	71	11815

Graphs of the TCP connections can be accessed by going to `Statistics > TCP Stream Graphs`. The number of TCP connections can be counted using this feature.
Although each platform opened a large number of TCP connections, not all of them were used to send or receive packets. By looking at the round trip time graphs in Wireshark, it can be noted that some of the TCP connections were created but remained idle during packet capture, as they never received any packets. Only a handful of connections were constantly receiving packets, and these connections were persistent since they were able to receive multiple objects over the course of the packet capture. Some other connections only received one or two packets before they were closed, which implies that there are some persistent connections that will be utilized a lot when transferring data, and there might be other connections on the side which are non-persistent which are used for transferring smaller amounts of data.

One method of looking for whether a connection was persistent was by searching for HTTP requests that would contain the `Connection: keep-live` or `Connection: close` in the header. However, the only packet capture which yielded an HTTP request was the one to YouTube, and there was no field specifying the connection status. This could be due to the default settings of HTTP 1.1 where if the connection field is not specified, then the default would be a persistent connection.

One outlier that occurred during the packet capturing was the DailyMotion video stream. Each packet capture lasted one minute, but DailyMotion captured around 6 times more packets than the other websites did. By looking at throughput graphs, it can be inferred that DailyMotion attempts to load a larger chunk of the video upon request, or that there is a larger gap between when the video loads and when the video begins to play.

2.

In this section, the goal is to determine whether each service uses constant bit-rate or variable bit-rate streaming and how that can affect the quality of the video.

	VBR or CBR?
YouTube	VBR
DailyMotion	VBR/CBR?
Vimeo	VBR
Odysee	VBR

The method used to check if a video stream is constant bit-rate or variable bit-rate was to track a specific TCP conversation and look at the packet throughput. If the video has a constant bit-rate, then the throughput graph should reflect that as packets are being transferred, and likewise for variable bit-rate.

Using this method, it seems that DailyMotion uses a mix between constant bit-rate and variable bit-rate stream. There is an initial period where a large number of bits are being transferred, but after the initial period, the same number of bits are being transferred in the same time intervals. This could be attributed to the constant bit-rate encoding, where the same number of bits will be sent for each frame of the video irregardless of any changes in the video. There is a small amount of variation in the number of bits being sent at each peak, and this could be attributed towards traffic in other TCP conversations or that DailyMotion uses some combination of VBR and CBR where there is a ceiling how many bits can be sent.

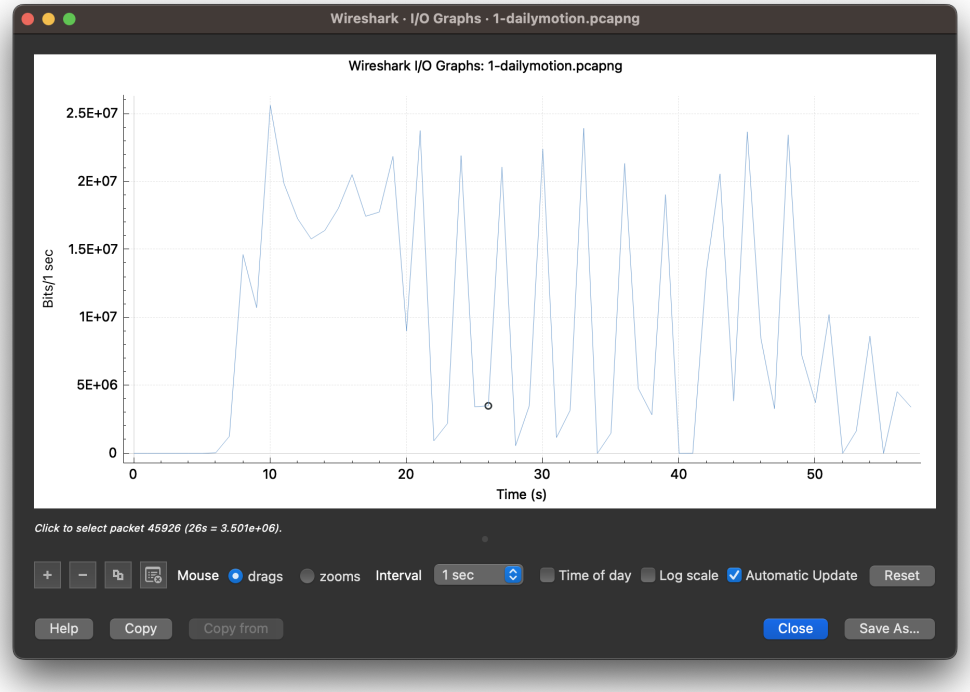

Figure 1: DailyMotion TCP graph

For the other three platforms, Vimeo, YouTube, and Odysee, the graphs were more erratic than the DailyMotion graph. It is likely that the other video streaming services use variable bit-rate since their graphs have less periodic spikes with varying peak heights.

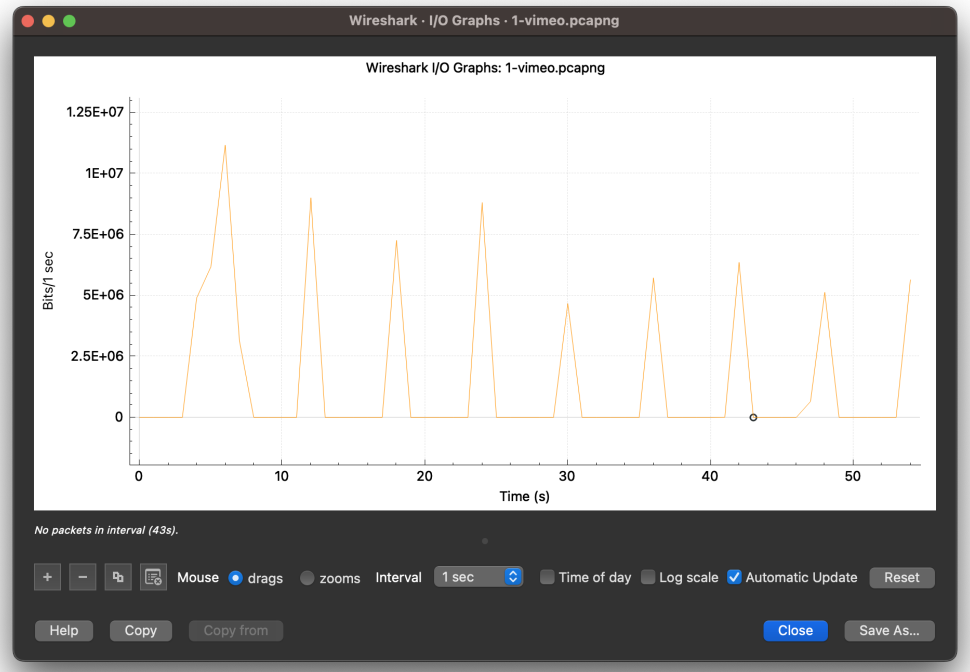

Figure 2: Vimeo TCP graph

It was noticed that the Vimeo TCP stream graph looked different than the other graphs. The graph seemed to have around 5 seconds between every burst of data. Since the peaks are not around the same height, it seems likely that this is not constant bit-rate, however, the graphs for YouTube and Odysee did not have a consistent rate at which data was being sent.

3.

For this section, the primary goal is to capture packets on a mobile device and compare how it is different when streaming video on a phone versus streaming video on a laptop. For comparison, the data from the previous two sections will be added to the table. The data is presented in the following format:

Persistent or non-persistent connection
Number of TCP connections
Number of packets during a 60 second interval
VBR or CBR

	Laptop (Wi-Fi)	**Phone (Wi-Fi)**	**Phone (Cellular)**
YouTube	Persistent 163 14711 VBR	Persistent 23 31425 CBR	Persistent 56 13125 CBR?
DailyMotion	Persistent 374 85749 VBR?	Persistent 38 59659 VBR?	Persistent 42 10738 VBR
Vimeo	Persistent 83 15329 VBR	Persistent 24 38341 VBR	Persistent 21 30879 CBR?
Odysee	Persistent 71 11815 VBR	Persistent 16 22145 CBR?	Persistent 18 26260 CBR?

All connections using Wi-Fi were believed to be persistent as there were many instances of TCP streams receiving more than one object during the duration of that TCP conversation.

Notice that when the video was streamed on the mobile device, there were usually less TCP conversations open but more packets captured (with the exception of DailyMotion). Additionally, streaming videos on a laptop almost always produced variable bit-rate streams, but streaming on a phone or using cellular made the graphs seem more like constant bit-rate stream graphs.

There were a few notable graphs when looking at specific TCP stream conversations to determine if a streaming service was using VBR or CBR. One graph in particular is the graph for YouTube while using a mobile device connected to Wi-Fi. This graph is a good example of what a constant bit-rate stream should look like. Notice that the rate at which bits are sent is constant, which implies that the same number of bits are sent for each frame, or that it has a constant bit-rate.

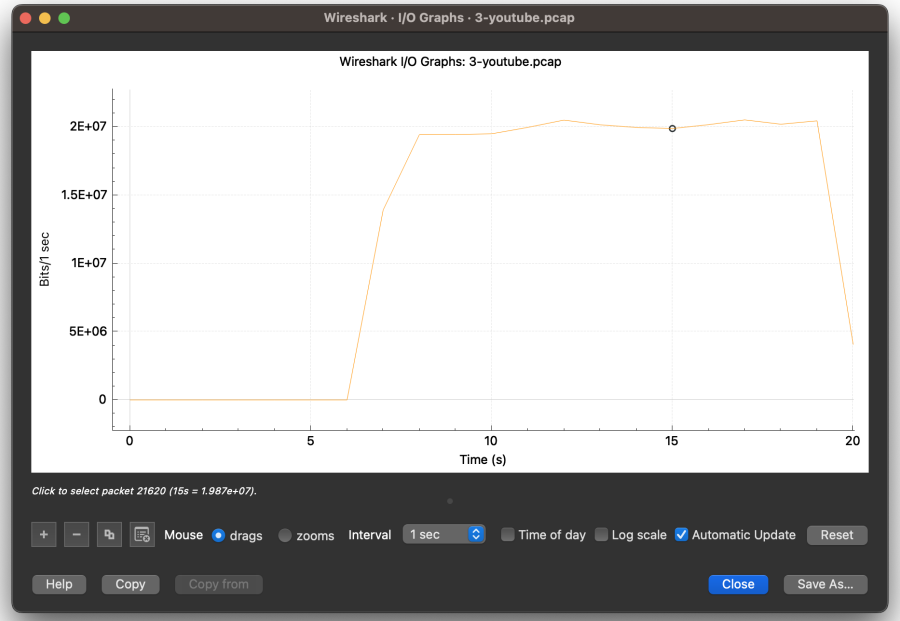

Figure 3: YouTube TCP stream graph for mobile Wi-Fi

However, this graph was the only graph where a constant bit-rate was clearly discernible. A variety of other TCP streams had graphs where only parts of the stream had constant bit-rate.

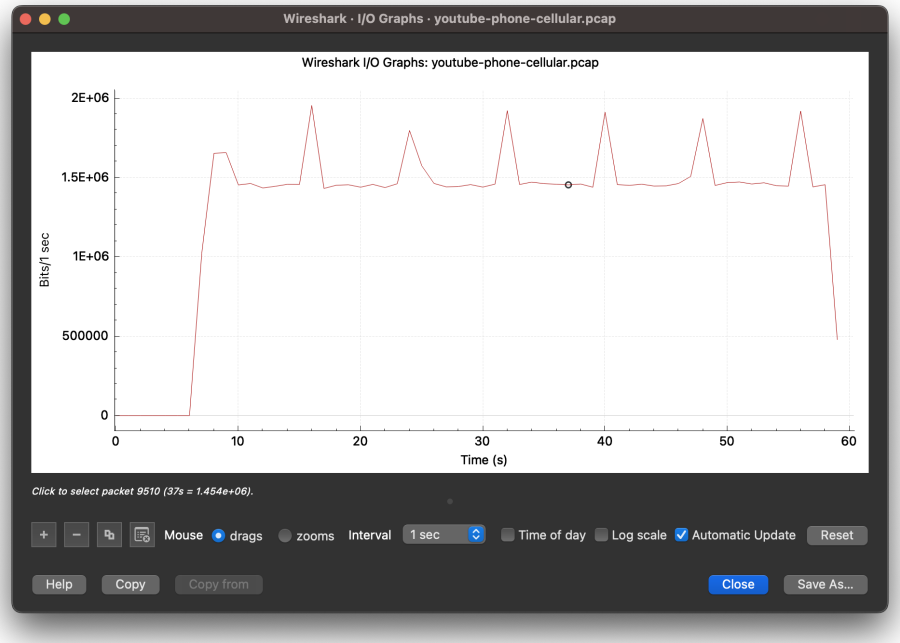

Figure 4: YouTube TCP stream graph for mobile cellular

Notably, the graph for the YouTube stream on mobile using a cellular network looked like a mix between VBR and CBR. For the most part, it maintains the constant bit-rate that is expected of CBR, but it also has periodic spikes where the bit-rate is higher. This could be attributed to the server needing to send more data at certain points as the video is playing, as even the spikes of data seem to happen with a constant interval.

If we set aside the specifics of the graphs, we can take a look at why the graphs differ between streaming services or Wi-Fi and cellular connection. For the laptop streams, the video quality on the laptop needs to be higher, since there is a larger screen and there are more pixels to fill. Presumably, this is why the laptop streams had more TCP connections open, because there was a larger quantity of data that needed to be transferred, due to the higher quality of the data. For mobile streams, the quality doesn't need to be as high, since the screen is small anyway. This is why the mobile streams have less TCP connections and are more likely to use CBR than VBR. If the video quality is lower, then the data can be sent all at one low rate of quality to make sure that packets are being delivered to the client on time.

It is worth mentioning the two Odysee graphs, as they differ from the other graphs.

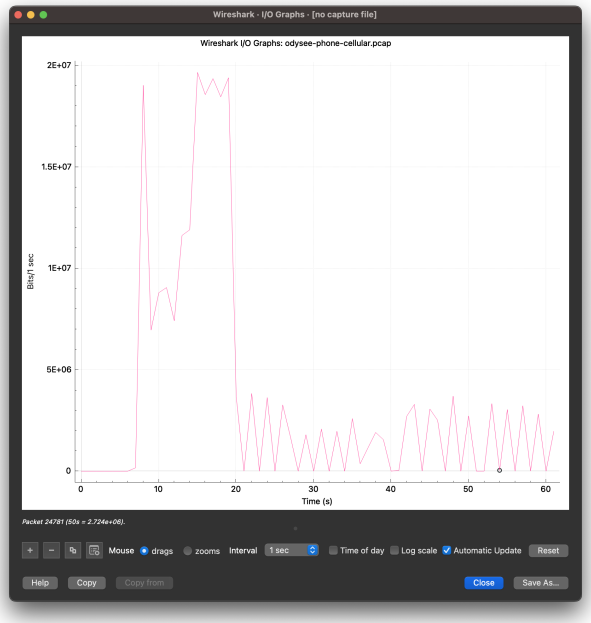

Figure 5: Odysee mobile cellular stream graph

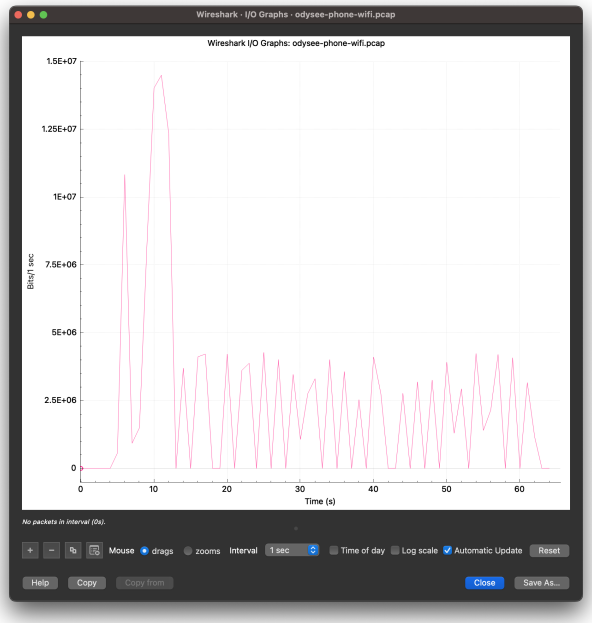

Figure 6: Odysee mobile Wi-Fi stream graph

The two graphs shown above appear to try and load a significant portion of the video at the very beginning. There is a large spike of bits being transferred around then, and the bit-rate drops after the initial period. Not only does the bit-rate drop lower, it also has the periodic spiking behavior shown in some other graphs like the Vimeo TCP graph. The reason for this behavior is unknown. It could be that Odysee uses a constant bit-rate, but only sends a burst of packets every few seconds. The behavior of the bit-rate dropping to zero at certain points can be seen in some other TCP conversations that were captured on the laptop. The behavior could also be affected by the buffer which has to process the packets on the client side, meaning that the server could send bursts of packets but the client is still processing them at the same rate.

The following paragraphs will be analyzing what happens when the link quality is changed as a packet capture is happening. A packet capture will run for one minute in which the phone or laptop is moved to a location with a weaker Wi-Fi or cellular signal.

	Phone (Low Wi-Fi)	**Phone (Low cellular)**
YouTube	Persistent 25 29444 CBR?	Persistent 22 10677 CBR?
DailyMotion	Persistent 48 42708 VBR	Persistent 48 12608 VBR
Vimeo	Persistent 23 31163 CBR?	Persistent 21 39688 VBR
Odysee	Persistent 38 39459 VBR	Persistent 29 30196 VBR

Between the weak Wi-Fi and weak cellular signals, there is not much difference. There is some variance in the number of packets captured, but that could be due to other applications running in the background. It did not matter if the connection was a weak cellular one or a weak Wi-Fi one, as the streaming service tended to open the same amount of TCP connections no matter which type of connection it was.

Comparing the low signal cellular results to the normal cellular results, there is not much difference. The results from the strong cellular signal seemed to have fewer packets sent, but not by much. They both have around the same number of TCP connections opened. It was noted that many of the low-signal cellular graphs looked more like variable bit-rate graphs, but this could be due to the weak connection interfering with the receiving strength of the data. It could be noted that the low cellular captures had lower throughput than the strong cellular captures did. This could be due to the streaming service choosing to send smaller packets and lower quality video, or it could be due to the weak connection causing less packets to be received.

Looking at the two YouTube graphs for mobile device on cellular data, there is not much of a difference. Both graphs seem to demonstrate that YouTube prefers to use constant bit-rate when streaming on a mobile device, although the strong cellular signal had the periodic peaks while the weak cellular signal did not have any uniform peaks. It is also noted that the strong signal had a higher throughput than the weak signal did, which is expected.

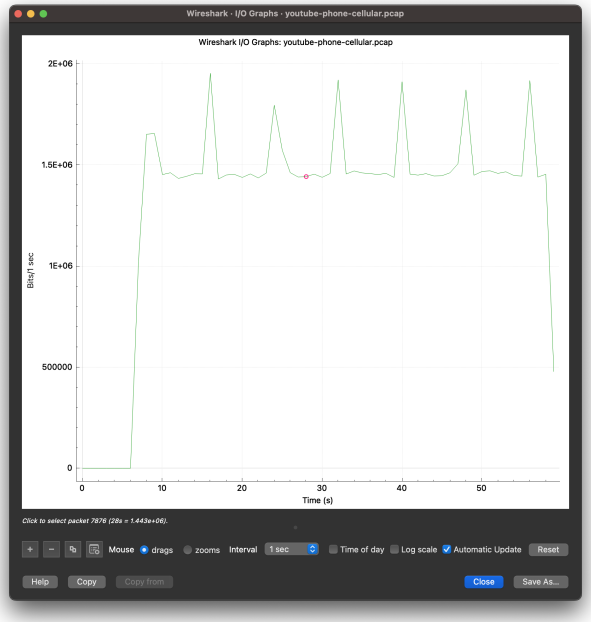

Figure 7: YouTube mobile cellular stream graph

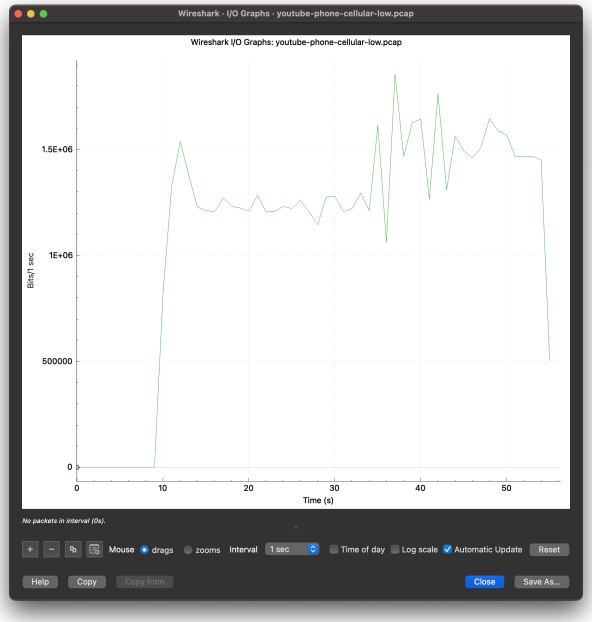

Figure 8: Youtube mobile low cellular stream graph

It seems that YouTube is the only platform which leans heavily towards using constant bit-rate streaming when it knows that the client is on a mobile device. It was more difficult to determine whether Vimeo and DailyMotion were using CBR or VBR just by looking at graphs, since they both had some stream graphs where the graph reflected VBR and CBR behavior. However, from the data gathered, it seems more likely that DailyMotion sticks to using VBR no matter how strong the connection is or what device the connection is coming from. It is more difficult to tell with Vimeo. A conclusion cannot be drawn from the data for Vimeo. Odysee is also difficult to tell, since it tends to have a large spike of bits sent at the beginning of the connection, and then periodic bits sent later throughout the connection. This behavior can be argued for CBR, which would explain why it is sending in constant bursts of the same bit-rate peaks, or it could be argued for VBR, where the bit-rate varies a lot which is why there are so many peaks. Regardless, it can be noted that YouTube, which is by far the most popular video sharing/streaming service, has the clearest data as to whether it uses VBR or CBR.

Finally, it seemed that the number of packets sent in the 60 second duration was not entirely dependent on the strength of the connection. For the YouTube packet captures, 3 out of 5 of them averaged somewhere near 15000 packets, but there were two that were higher than that. Both Vimeo and Odysee had an increasing number of packets that were captured, with the lowest packet capture rate being for their laptop Wi-Fi connections, and the highest packet capture rates for their low cellular and Wi-Fi connections respectively. This could be due to more packets being sent because the connection is poor, so there are dropped packets which need to be resent. On average, DailyMotion sent almost three times as many packets as the other platforms, and it is unsure why this behavior occured.