Networks Coursework 2: Monitoring Network Performance

Hemal Munbodh 100246776 and Robin Rai 100242165

4,611 Words

Contents

Introduction and Abstract 2
1 Routing Measurements 2
1.1 Methodology 2
1.2 Traceroute Analysis 3
2.2.1 nepalnews.com Traceroute Analysis 3
2.2.2 kbs.co.kr Traceroute Analysis 4
2.2.3 japantimes.co.jp Traceroute Analysis 4
2.2.4 nepallive.com Traceroute Analysis 5
2 Packet Loss and Round-trip time measurements 6
2.1 Methods for acquiring ping and packet loss data 6
2.2 Packet Loss and RTT Analysis 6
2.2.1 vok.rep.kp Packet Loss and RTT Analysis 7
2.2.2 nepallive.com Packet Loss and RTT Analysis 7
2.2.3 express24.uz Packet Loss and RTT Analysis 8
2.3 Lossy Target Packet Loss analysis 8
3 Throughput measurements 9
3.1 Throughput Affected by File Size 9
3.2 Throughput Over the Course of Two Weeks 9
3.3 Throughput Affected by Parallel Instances 10
3.4 Saturating Throughput 11
4 Conclusion 11
5 Credits 12
5.1 Group Participation 12
5.2 Acknowledgements 12
5.3 References 12

Introduction and Abstract

This report is an assessment of the performance of the 'real Internet' using metrics that describe routing stability, throughput, round trip time (RTT) of packets and rate of packet loss. It seeks to draw a correlation between the above metrics and changing environment variables including time of day, simultaneous requests and uses, geographical location amongst others. This will be done by monitoring selected websites, with common Unix network tools, such as ping (ping - Linux man page, 2020), traceroute (traceroute - Linux man page, 2020), wget (wget - Linux man page, 2020) and chronological tab (crontab) (crontab - Linux man page, 2020). Parsers were written in Java. Graphs and figures were made in Microsoft Excel.

The findings of this report will showcase interesting behaviour with possible explanations. It will serve as a lookup of explanations for common networking scenarios.

1 Routing Measurements

1.1 Methodology

Over 50 foreign websites were chosen and monitored at regular intervals over the course of 2 weeks. This was done with UNIX, running on a Raspberry Pi. A shell script was created that ran the "traceroute" command for every website specified. The results of this query were then appended to an output file, with parser friendly formatting (split by pound signs and date). This shell script was then automated to run every six hours, via crontab. This was then left to run independently for the 2 weeks specified.

```
###
Tue 12 May 00:45:01 BST 2020
###
traceroute to kbs.co.kr (210.115.192.7), 30 hops max, 60 byte packet
 1  139.222.0.1  0.934 ms  1.420 ms  1.414 ms
 2  10.0.0.1  3.964 ms  9.948 ms  6.855 ms
 3  193.62.92.124  0.846 ms  0.841 ms  0.821 ms
 4  193.62.92.71  1.326 ms  1.520 ms  1.174 ms
 5  193.62.92.70  1.143 ms  1.279 ms  1.423 ms
 6  146.97.41.201  3.963 ms  4.491 ms  4.417 ms
 7  146.97.35.245  4.426 ms  4.415 ms  4.266 ms
 8  146.97.33.30  16.843 ms  16.753 ms  16.749 ms
 9  146.97.33.61  7.621 ms  7.668 ms  7.477 ms
10  146.97.35.194  7.456 ms  7.283 ms  7.188 ms
11  212.187.216.237  7.862 ms  7.795 ms  7.739 ms
12  4.69.153.117  138.438 ms 4.69.153.121  138.368 ms 4.69.153.117
13  61.42.234.5  138.927 ms  138.461 ms  138.488 ms
14  203.233.117.165  138.788 ms 203.233.17.45  151.049 ms  150.917 m
15  1.208.175.17  281.328 ms 1.208.104.45  272.048 ms 1.208.146.17
16  * * *
17  1.208.145.17  277.894 ms  279.715 ms  277.933 ms
18  1.208.147.194  277.138 ms 1.208.148.34  280.945 ms 1.213.151.21
19  1.213.13.114  281.153 ms 1.208.13.118  271.781 ms 1.208.13.114
20  * * *
21  210.115.192.7  283.562 ms  286.548 ms *
###
```

Figure 1- Raw shell output with pound delimited dates

The first parser takes in the routes outputted, and associate geographical data to each of them, then group them by Uniform Resource Locator (URL). It then outputs all unique routes taken for every URL, along with the geographical locations of each hop in a formatted and readable output.

This was done by making a route object for every route, and a website object for each website URL. Each route would store its URL and an array of the Internet Protocol (IP) addresses of each traceroute hop, representing the route. It would also store the geographical data of each hop in an equally sized array. The website object would simply store its own route objects.

Each route object was compared to each other to remove duplicate routes. (If the arrays of IPs and Geographical addresses were identical.) While this works to some degree, it does not cater for small variations that could be considered the same route. For example, a route that makes an extra hop within a location to another would be seen as unique.

```
URL: kbs.co.kr
number of routes: 6
kbs.co.kr 139.222.0.1, 1.256, Norwich,Jisc Services Limited,Norfolk,United Kingdom,NR5,52.6413,1.2197
kbs.co.kr 193.62.92.124, 0.836, Belvedere,Jisc Services Limited,Bexley,United Kingdom,DA17,51.4911,0.1514
kbs.co.kr 193.62.92.71, 1.3399999999999999, Belvedere,Jisc Services Limited,Bexley,United
Kingdom,DA17,51.4911,0.1514
kbs.co.kr 193.62.92.70, 1.2816666666666665, Belvedere,Jisc Services Limited,Bexley,United
Kingdom,DA17,51.4911,0.1514
```

Figure 2- Example output of the first parser

Finally, to find the location for each IP address in a route, a third-party service, and Application Programming Interface (API) was used. This was originally GeoBytes' GeoIP2 Lite, which was recommended to us. This service, however, gave incorrect and misleading location data. A new service was found used, IPinfo.io. Using the API required providing it with an IP address, which then allowed for geographical data and IP data such as country and ISP name to be returned as strings. To plot this data on a map, a second parser was made and used. This would remove any sequential duplicate locations (i.e. 47,8 followed by another 47,8) and would leave only the longitude and latitude in individual CSV files. This process would naturally leave identical files. These were scanned and only a single copy left undeleted. This left us with completely unique geographical paths to plot.

1.2 Traceroute Analysis

We aimed to find websites that describe these characteristics for their routes: A variety of routes, fluctuations of RTTs, Geographical diversity, and interesting occurrences/behaviour (loops, parallelisation)

2.2.1 nepalnews.com Traceroute Analysis

nepalnews.com is a Nepalese news website hosted in Nepal. It features several distinct routes.

Figure 3- An example of a route that hops from the United Kingdom to the USA, then back to Sweden.

Figure 4 - A route going from the USA straight to Nepal

Figure 5 - A route that does not go through the USA

Comparing figure 3 with 4 shows 2 paths taken from router number 5 onwards - i.e. packets in figure 3 hop back to router 6 in Sweden while those in figure 4 go directly to Asia. This behaviour seems wasteful because packets routed to Los Angeles are routed back to Sweden leading to a large increase in RTT. This behaviour was frequent. The reason as to why this happens is unclear. One reason could be load distribution. However, it is not sensical for a router in Los Angeles to load-distribute with a router in Sweden. Interestingly, both routers 5 in figure 3. and router 6 in figure 4. are assets of a multinational Swedish telecommunications company, Telia.

Figure 3, 4 and 5 show some of the differing routes. This variability seems to point to unstable routing, unlike kbs.co.kr or japantimes.co.jp seen later. It is to be noted that kbs.co.kr and japantimes.co.jp are likely to be more popular websites than nepalnews.com - a possible reason, as similarly pointed by (Rexford, Wang, Xiao and Zhang, 2002) is that unpopular websites have unstable Border Gateway Protocol(BGP) routes.

Routes that go to America have significantly higher RTTs (~300ms), as opposed to the rest of the routes, which is not the most stable but hovers around (~200ms). The routes to America took place periodically, (14 - 16th,25 - 27th May). This could contribute to the DDoS protection theory, as it could be a routine inspection.

2.2.2 kbs.co.kr Traceroute Analysis

The website kbs.co.kr is South Korea’s public media broadcasting website. It features a single distinct route, composed of 15 near identical routes that differ only in network interface changes - routers at a single location transferring the packet amongst themselves. The traceroute packets all travel through the US, making a single hop in Kansas. A possible reason for travelling westwards through the US despite the further distance could be possible use of the Asia-America Gateway (AAG). The AAG is a stable route through the US to Asia and is less prone to frequent routing changes and instability unlike eastward routes like nepalnews.com and nepallive.com. This could be because packets must go through multiple boundaries, countries, interfaces and routers that enforce different routing policies.

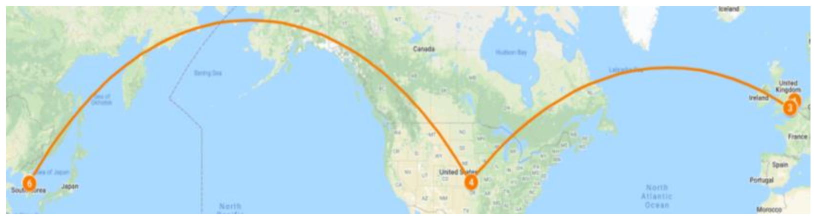


Figure 6 - The full path from Norwich to KBS

The route is provided by three *services*, Jisc Services Limited in the UK, Level 3 Communications in the US, and LG DACOM in South Korea. It is to be noted that parallel routing takes place with the US and Korean providers - the three probes per traceroute are sent to three separate routers at the same location. This is likely done to spread load over multiple routers.

```
12  4.69.153.121  138.302 ms 4.69.153.117  138.796 ms 4.69.153.121  138.714 ms
```

Figure 7 - An example of each probe on a hop

Also, the single hop covering the entirety of America shows the possibility that there is hidden routing. Routers might not be decrementing the traceroute packet’s Time to live (TTL), a Virtual Private Network (VPN) could be in use, or a MPLS network could be used are all possible reasons for only a single hop showing.

The RTTs measured to kbs.co.kr are very consistent, but quite (expectedly) high. The RTTs increase by a large expected degree when reaching the US, and expectedly increase further when reaching South Korea. The consistency of the RTTs can be attributed to the very stable route, and high-quality hosting. The route taken to kbs.co.kp is roughly 17965 kilometres. Given that fibre cabling has a refractive index of 1.48 (Steenbergen, n.d.), light/data can travel through it roughly at 200 kilometres per millisecond (1/1.48 = ~0.67c = ~200,000KM/s one way). This can be halved to account for the return trip. With roughly 17965 kilometres, an expected RTT of 180 milliseconds can be calculated. In reality, the RTT is between 275 and 345ms, which can also account for packet congestion and routing lookup.

2.2.3 japantimes.co.jp Traceroute Analysis

www.japantimes.co.jp is Japan’s oldest and largest English newspaper. While it is hosted/routed to by Amazon, it has a single stable host destination and two distinct routes. This was constructed from 53 routes. One route goes through Cogent infrastructure, while the other goes through NTT Group. When travelling through NTT’s infrastructure, the overall RTT is lower, by roughly 20 milliseconds (~230 compared to ~250ms). This could be from the ingress route, but could also be from the return route, which is not visible to traceroute. The reverse route can differ greatly from different providers, as they will have

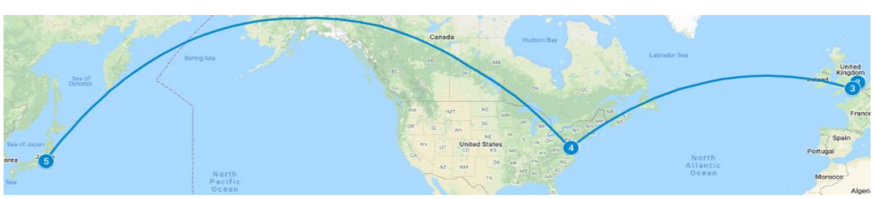


Figure 8 – The false American hop

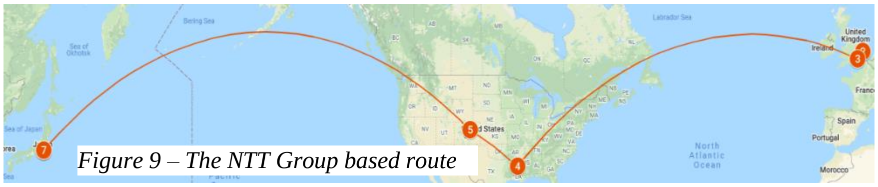


Figure 9 – The NTT Group based route

different rules and policies. This can affect RTT.

While the two similar routes being taken could indicate load sharing, the route chosen was determined by date. For the first two days of monitoring the route went through NTT (12th-14th May), and the remaining 10 days went through Cogent.

For both routes, the fourth hop goes to Louisiana. This is false location data, as the router's name, ae-2-3204.ear2.london2.level3.net, lists London, and its RTTs indicate a very small geographical distance change. The hop after it seems to be genuinely in America though, as RTTs increase. Like kbs.co.kp, a single hop covering the entirety of America seems unlikely and could be explained by routers not be decrementing the traceroute packet's Time to live (TTL), a Virtual Private Network (VPN) could being in use, or a MPLS network could be in use.

```
www.japantimes.co.jp 212.187.216.237, 7.7763333333333335, London,9-2-2.ear3.london2.level3.net
AS3356 Level 3 Parent, LLC,England,null,51.5085,-0.1257
www.japantimes.co.jp 4.69.143.194, 7.548666666666667, Monroe,ae-2-3204.ear2.london2.level3.net
AS3356 Level 3 Parent, LLC,Louisiana,null,32.5530,-92.0422
```

Figure 10 – The Cogent based route

Like KBS, the route goes westwards, potentially through the Asia America Gateway. It also features very stable RTTs, with an expected increase in RTTs when looking at distance travelled. It is to be noted that the ping tool reported 100% packet loss. This is because the final hop/host server does not support ICMP (traceroute times out).

Both routes are in the ballpark of 17000 and 18000km. Based on the previous calculations, an RTT of ~170 and ~180ms can be gained. This comes quite close to the measured result of ~230 and ~250ms.

2.2.4 nepallive.com Traceroute Analysis

nepallive.com is another Nepalese news website hosted in Nepal. It too features many different routes, including new visible behaviour - DDoS mitigation by Cloudflare. Most routes (33 distinctions in total) are mostly consistently routing through Europe and India, with some travelling through Singapore before reaching Nepal. The routes go through Jisc services and Telia Company through Britain, then Tata Communications. Some routes however, diverted to America, going straight from Jisc Services to Cloudflare - where the route promptly ended. This is highly likely to be Distributed Denial of Service (DDoS) protection as it is one of Cloudflare's biggest services. A DDoS attack is where many distributed devices maliciously send ICMP packets to a server to the point at which it is overloaded and cannot perform its duties. This is because processing ICMP is slow and usually not hardware accelerated, unlike regular packets (Steenbergen, n.d.). Since traceroute works by sending ICMP packets in an automated and regular manner, it could be potentially seen as an attack.

Figure 11 – The route to Cloudflare

The routes that get stopped in America took place on the first two days of testing (12th - 14th May). From then on, regular and complete routing took place. This could be from the system fingerprinting our monitor as a non-threat.

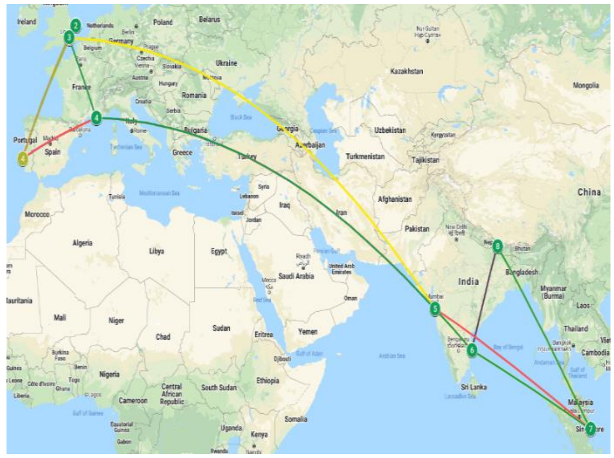


Figure 12 – Many of the true routes to Nepal

The routes that do not get stopped are distributed over several routers spread over Europe and Asia. These routes change very frequently (daily). Due to this high frequency and close geographical distance to these routers, this is most likely done for load distribution. This is where traffic is spread out over nearby servers, to reduce the possibility of overloading and congesting one.

Aside from the trips to Cloudflare (very low and consistent RTTs), the RTT varies greatly among these routes, ranging between 180 and 310ms. However, geographical hops do not affect the RTT. For example, similar routes that travelled through the detours of both Lisbon and Singapore still shared a large range of RTT (180 - 300ms). Since the website is small, this could be attributed to server load. This could also be from the egress route.

2 Packet Loss and Round-trip time measurements

2.1 Methods for acquiring ping and packet loss data

To acquire the ping data, another shell script was written to run the “ping” command for the same list of websites. Each ping command ran 120 measurements. The script would then append the results to a file with special formatting for parsing. This was then left to run independently for 2 weeks.

The parsing program’s goal is to collate the results of each website, then calculate the average, median, maximum, and minimum RTTs for each set of 120 pings of a given website. The packet loss for each set of pings was also extracted. This was then outputted to a CSV file to be further processed in Microsoft Excel. This was done in a similar way to the route process.

2.2 Packet Loss and RTT Analysis

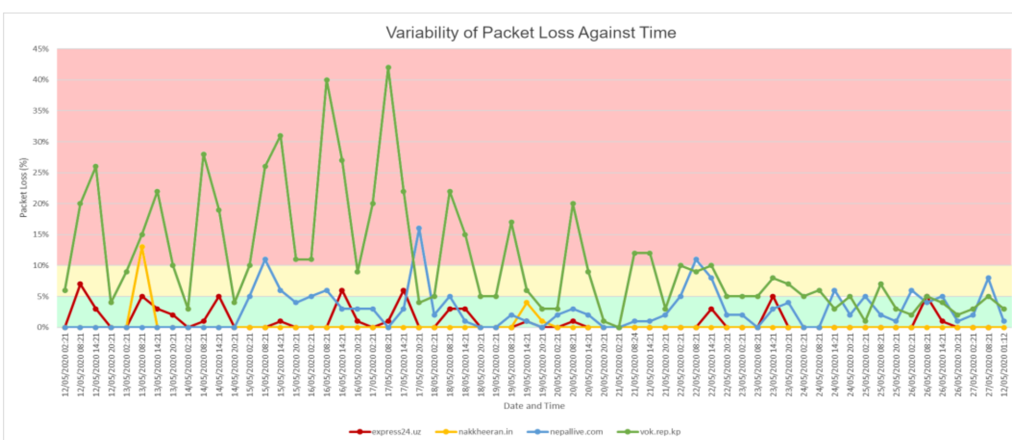


Figure 13 – Variance and classification of packet loss over a two-week period. Minor loss – green, Significant loss – yellow, Major loss - green

2.2.1 vok.rep.kp Packet Loss and RTT Analysis

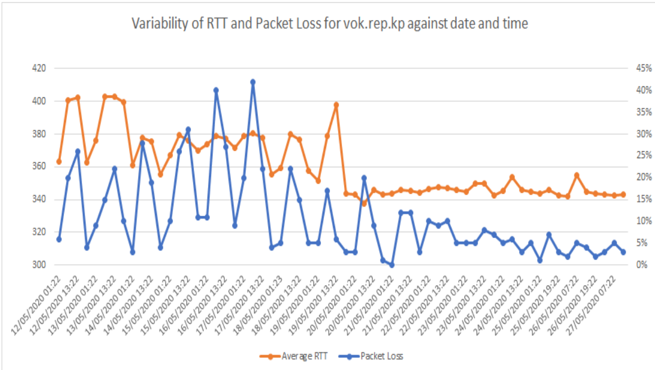


Figure 14 – Variation in average RTT and packet loss for vok.rep.kp against time

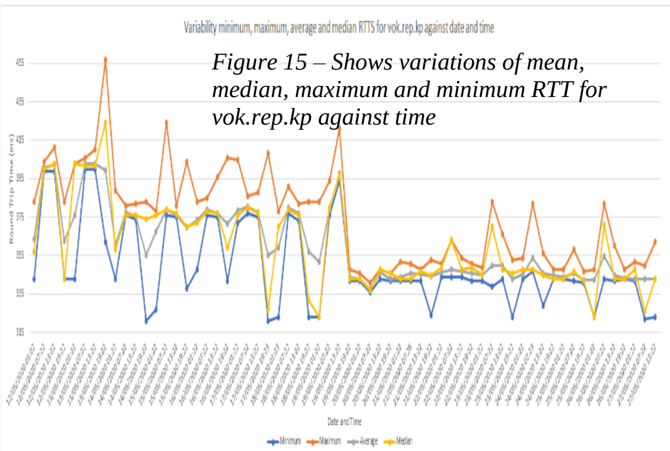


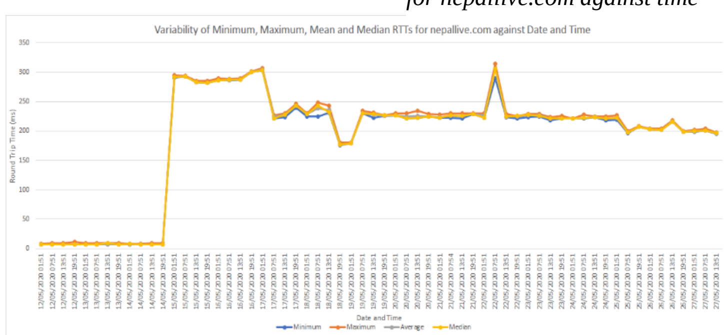
Figure 15 – Shows variations of mean, median, maximum and minimum RTT for vok.rep.kp against time

The Voice of Korea (VoK) is the Democratic People's Republic of Korea national radio broadcaster. Most notably, this website experiences very frequent packet loss with one exception (21st May). This is not a surprising result as vok.rep.kp is hosted within China by Unicom, a Chinese telecommunications company. China is well known for heavily censoring the inbound internet traffic and tampering with it. In fact, the term Great Firewall of China is a blanket term for a series of techniques used to monitor and censor traffic. This explains constant packet loss but not very high losses above the 10% threshold, which are fairly frequent between the 12th and 21st May. Notably, both RTTs and package loss decrease after the 21st May. From figure 15, we notice an expected relationship between mean and median RTT i.e. when median increases, mean also increases and vice versa. Median gives an indication of the skew of a dataset. So, if the median RTT is relatively small, the mean RTT is likely to be small as well. The variability of average RTT in figure 14 seems to point to congestion. The latter can be supported by the claim in Xu's, 2016 article that Unicom's infrastructure is congested and underdeveloped.

2.2.2 nepallive.com Packet Loss and RTT Analysis

Like its traceroute, nepallive.com features interesting data. As mentioned in the traceroute analysis, the first two to three days monitored went to Cloudflare in America. This can be seen in the ping measurements, as both ping and packet loss are at a minimum/is not present. This can be attributed to the higher quality infrastructure with the western route. When the route actually does go to Nepal, noticeably higher packet loss and RTTs are seen. RTTs are quite consistent, while packet loss fluctuates slightly. There are two periods where both RTT and

Figure 16 – Shows variations of mean, median, maximum and minimum RTT for nepallive.com against time



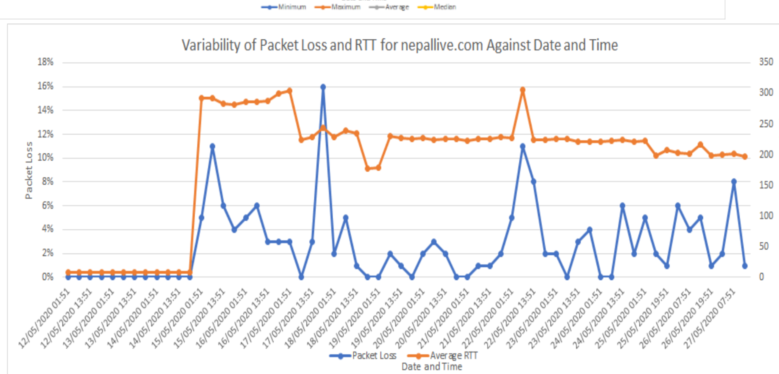


Figure 17 – Variation in average RTT and packet loss for nepallive.com against time

packet loss spike. A major spike took place at ~8AM UK time. Adding 4 hours and 45 minutes, a time of ~1PM Friday Nepal time is found. This can reasonably suggest a large load of requests to the site. Differences between minimum, maximum, median, and average RTTs is minimal. This suggests that the different routes taken are indeed quite close to each other geographically.

2.2.3 express24.uz Packet Loss and RTT Analysis

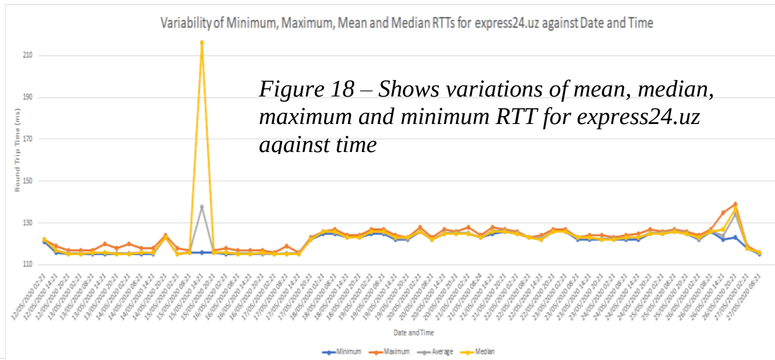


Figure 18 – Shows variations of mean, median, maximum and minimum RTT for express24.uz against time

express24.uz is a website for food delivery, like Deliveroo in the UK. It features consistent median, mean, maximum and minimum RTTs, though there is a spike present on Friday the 15th May, and a general increase after the 17th. Looking at the traceroute data did not reveal any special or unique route taken on that day. Given the small infrastructure of Uzbektelekom Joint Stock Company, this could be from high server load. This seems likely as this spike was on a Friday - a prime time for takeout. The small infrastructure could also explain the intermittent packet loss. It is also to be noted that on this very day the COVID 19 lockdown policies were eased for businesses in Uzbekistan according to (Seyfaddini, 2020). Given that the RTTs also consistently increase specifically after the weekend (the time at which take out restaurants could be closed or demand lowered), the RTT data seems to directly reflect on the operations of restaurants before and after lockdown measures have been eased, with the spike being the reopening of restaurants.

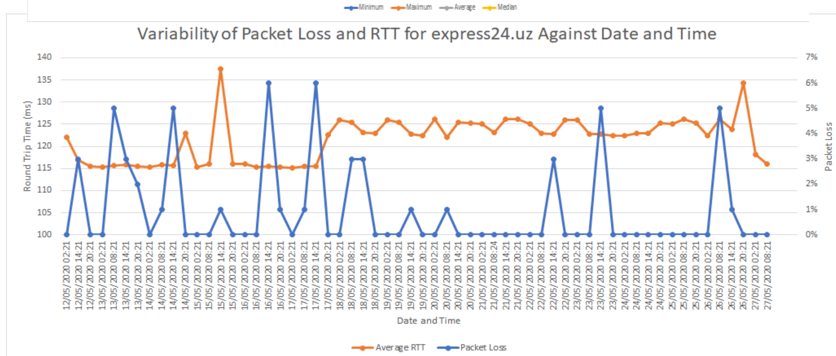


Figure 19 – Variation in average RTT and packet loss for express24.uz against time

2.3 Lossy Target Packet Loss analysis

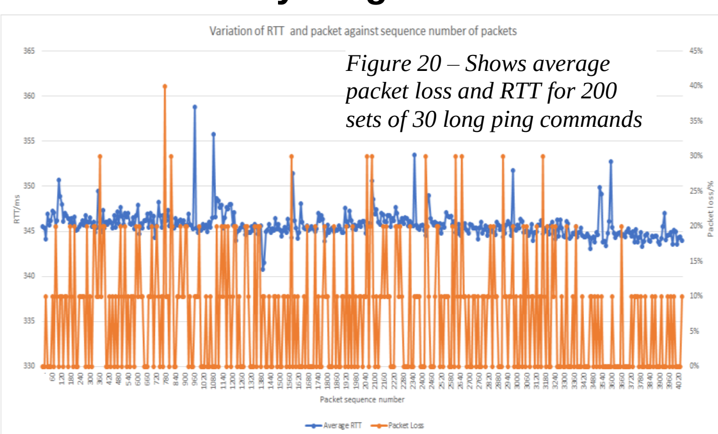


Figure 20 – Shows average packet loss and RTT for 200 sets of 30 long ping commands

This data was gained by a shell script that performed several but short ping commands (20 pings per command, 200 times). This gave us packet loss and RTT in 20 ping segments.

From figure 20, generally, a packet loss event will follow a sudden spike in average RTT. This would indicate high traffic, as the network is congested, leading to low priority ICMP packets either taking longer to be processed, or dropped out of the queue entirely. Aside from these spikes, the RTTs and packet loss seem to stay at consistent levels, where RTTs stay around 345ms, and packet loss bounces between 0, 10 and 20 percent loss. The RTT seems to also vary in a diurnal fashion at an average of 400 packet intervals. This raises a question about

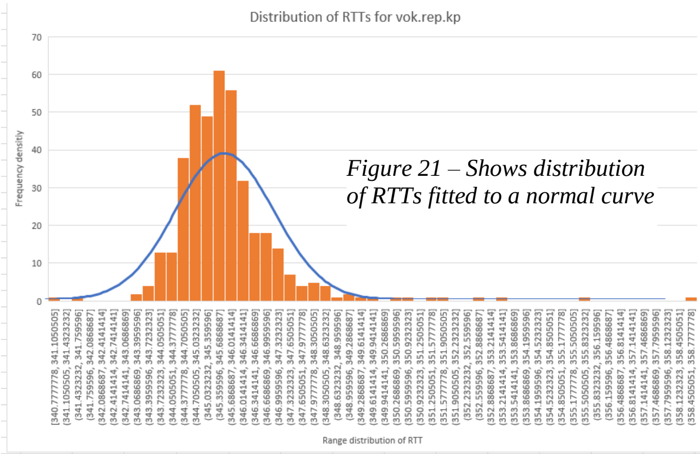


Figure 21 – Shows distribution of RTTs fitted to a normal curve

whether a slowdown is being intentionally imposed. Alternatively, the return packets may be taking a different egress route.

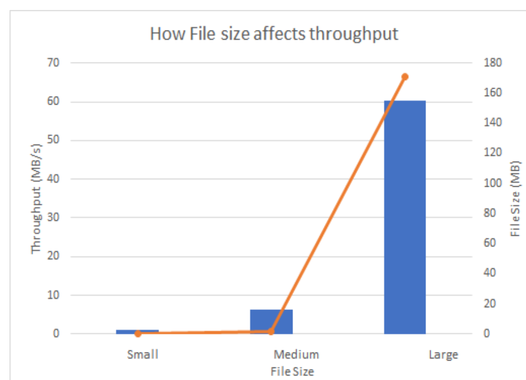
According to the Shapiro-Wilk test for normality, the data in figure 21 does not follow a normal distribution. Nonetheless, the diagram describes a best-fit scenario where a normal distribution is made to roughly fit the data using the maximum-likelihood method. We can thus infer some information from the distribution. For example, it is to be noted that the distribution curve is reasonably flat and wide indicating some degree of deviation from the mean. Again, this fluctuation in RTT might be indicative of congestion. Looking at the histogram, the data is also positively skewed meaning low RTT values are more likely to occur than very high anomalous ones. Unfortunately, while a normal distribution is a good measure of central tendency, it underestimates regions of abnormally high or low RTTs. In the case of network infrastructures, it is more important to understand why RTTs may be peaking or dipping rather than to have a general view of the central tendency. As such, we can use generalized extreme value distributions, which place importance on extreme values.

3 Throughput measurements

3.1 Throughput Affected by File Size

3.a. A simple bash shell script was written to perform the downloads of differently sized files. A small, medium and large sized file was chosen, with respective sizes of 69KB (txt), 1.4MB (exe), and 171MB (iso). Downloads were performed on a Debian FTP server hosted in Norway, that had no route changes seen in traceroute. An average of ten runs was taken.

Figure 22 – File size and throughput



With regards to a network, file size affects throughput in a few ways. Firstly, it affects the efficiency of the network. For larger files, a larger ratio of data sent over the network is the actual file, instead of packet header data. For example, if a TCP header was a kilobyte, sending a 1KB file is only 50% efficient, as half of the throughput is for the header data. Compared to a 60KB packet, over 98% of the throughput is dedicated for the file. A second reason is packet efficiency. A network can only send packets at a fixed rate, so the more data a packet stores, the higher the network's throughput.

3.2 Throughput Over the Course of Two Weeks

For running a constant download over the period of two weeks, a shell script was written to perform the download, and write the results and dates in a parser friendly format. This script was then run for two weeks automatically by crontab.

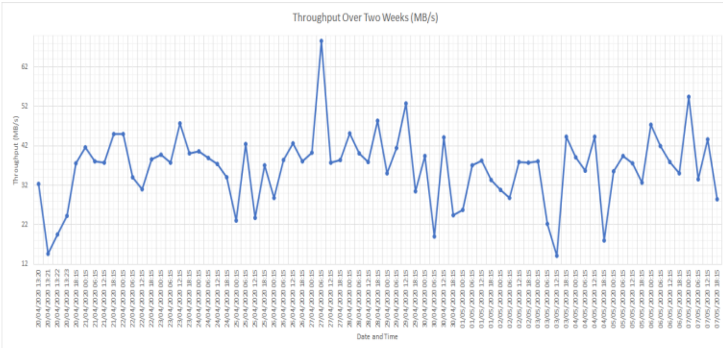


Figure 23 – Throughput of the large file over two weeks

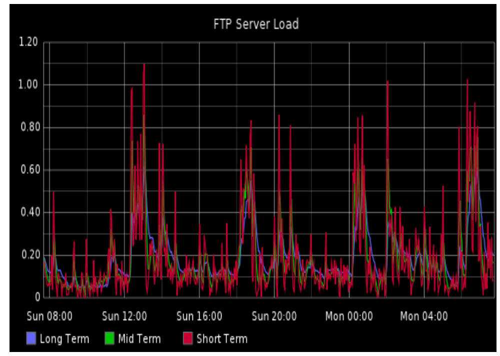


Figure 24 – A short history of the server's load

Given that this FTP server is quite obscure (a mirror FTP server for a Norway university), and in past experiences throughput can be quite temperamental, the results of this data can vary greatly. For example, when conducting other tests and downloading the same file consecutively, throughput was usually 60 MB/s, but occasionally would yield a greatly lower result. This could be attributed to server use, with the seemingly random times being attributed to the small size of the server being more easily affected by load. A short history of the server load can attest to/confirm these seemingly random times.

3.3 Throughput Affected by Parallel Instances

As for parallel downloads, xargs was used to run the wget command in parallel. The file and its size were kept constant (171MB).

```
cat url10.txt | xargs -n 1 -P 10 wget
```

Figure 25 – The command ran

This was done with values of K ranging from 1 to 10, and each K value was done 10 times and an average was taken. The data shows that throughput for each file download decreases exponentially when K increases. With downloads where K is greater than 1, the downloads do not always have evenly distributed throughput. This is because one thread/instance of wget gets more priority/execution time/allowed to execute before the others. This then takes place again with the other instances/threads, leaving the one with lowest priority the lowest throughput. Since this thread gets more execution time, it completes faster at the expense of the other threads. Note that when averaged the throughput result is similar to one with more parallel execution.

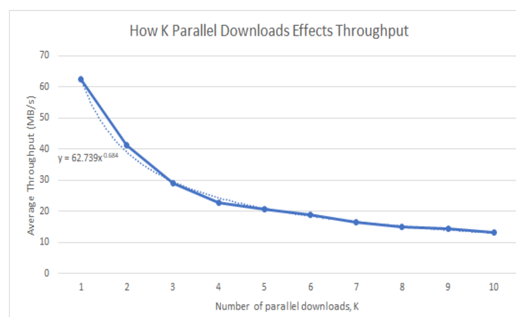


Figure 26 – How K parallel downloads effects throughput

Table 1 – 3 Simultaneous downloads with two forms of distribution between instances

Even distribution	36.5	36.1	33.4
Staggered distribution	42.5	26.8	16

As argued by (Love, 2002), total throughput can be limited by several factors. The receiving device (in this case the Raspberry Pi) could have limited receiving bandwidth, misrepresenting the network speed. Another factor could be the acknowledgement mechanism when downloading. This involves waiting for acknowledgement to return before sending another chunk of data. Accounting for the RTTs of acknowledgement packets decreases throughput for parallel downloads. Potential downsides to this testing method include network congestion, as seen with throughput over time as above, as well as packet loss, as that requires resending packets. The first factor was mitigated by taking the data all at one time, and the second did not happen with this server.

3.4 Saturating Throughput

To find out how RTTs are affected by a saturated network, a shell file was written to perform several sequential downloads. With a 25 second gap in between, a file was downloaded 10 times with 10 parallel instances, to completely saturate the channel. Another shell file was written to perform the ping command, with 600 consecutive pings. These two scripts were then run in parallel.

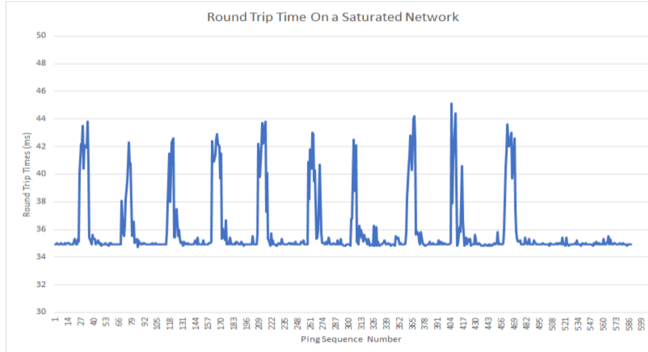


Figure 27 – RTTs when downloading

The data gained showed that RTTs increase when the channel is saturated. This could be from packet priority, as there are more regular packets at a given time, reducing the likelihood that a low priority ICMP packet will be dealt with promptly. The network, or slowest part of the network can also be congested with packets, increasing utilisation and therefore queueing times, thus increasing RTT according to (Steenbergen, n.d., pp 28-33).

4 Conclusion

Conclusively, the internet is very stable in the Western region with stable RTT and almost no packet loss. A surprising number of websites from the developing regions are being hosted by giants such as Amazon Web Services or Google. The latter provide very good service, which limits packet loss and fluctuating RTTs. If a website is planning to scale and expects to have high international traffic, it is very sensical to go with established hosts. While websites in Nepal and India do incur some loss and fluctuating RTTs, it is important to note that they most likely have small to medium locally based website traffic. Still, many of them utilize Cloudflare's DDoS protection service.

We noticed artificially high packet loss and RTTs in China. This is mainly due to the "Great Firewall of China" which filters out a huge amount of incoming international packets and data. So, if one is planning to host in China and hopes to attract international traffic, this is unfortunately difficult.

Finally, many of the tools we used can be fallacious. For example, ICMP packets have very low priority (or can often be discarded) by routers. This can result in high packet loss or RTT that is not necessarily indicative of bad network performance.

All in all, we realise that these analyses are largely best effort but strongly believe in pointing readers in the right direction for further experimentation.

5 Credits

5.1 Group Participation

Hemal Munbodh (50%) and Robin Rai (50%)

We both did work perfectly evenly. All work was done together, in person then via video calling and screen sharing. Gaining data, researching, plotting, brainstorming and report writing was all done together and distributed evenly.

5.2 Acknowledgements

Many thanks to Dr Edwin Ren, Artjoms Gorpincenko, Martino Gonzales and Brandon Hobley for their insightful technical assistance.

5.3 References

2020. *Visualisation Of Server Load With Time*. [image] Available at: <http://ftp.no.debian.org/> [Accessed 10 June 2020].

Asia-america-gateway.com. 2020. [online] Available at: <https://www.asia-america-gateway.com/> [Accessed 1 June 2020].

Love, S., 2002. *TCP Throughput Limit And Maximum Download Speed*. [online] Vad1.com. Available at: <http://www.vad1.com/tcp-throughput-limit.html> [Accessed 5 June 2020].

Linux.die.net. 2020. *Ping - Linux Man Page*. [online] Available at: <https://linux.die.net/man/8/ping>

Linux.die.net. 2020. *traceroute - Linux Man Page*. [online] Available at: <https://linux.die.net/man/8/traceroute>

Linux.die.net. 2020. *wget - Linux Man Page*. [online] Available at: <https://linux.die.net/man/1/wget>

Linux.die.net. 2020. *crontab - Linux Man Page*. [online] Available at: <https://linux.die.net/man/5/crontab>

Rexford, J., Wang, J., Xiao, Z. and Zhang, Y., 2002. BGP routing stability of popular destinations. *Proceedings of the second ACM SIGCOMM Workshop on Internet measurement - IMW '02*,.

Seyffadini, 2020. [online] Available at: <https://menafn.com/1100167306/Uzbekistan-reveals-its-COVID-19-statistics-for-May-15> [Accessed 11 June 2020].

Steenbergen, R., n.d. *A Practical Guide To (Correctly) Troubleshooting With Traceroute*. [ebook] pp.25-30. Available at: <https://archive.nanog.org/meetings/nanog47/presentations/Sunday/RAS_Traceroute_N47_Sun.pdf> [Accessed 28 May 2020].

Xu, Y., 2016. *Deconstructing The Great Firewall Of China*. [online] Internet and Cloud Intelligence Blog | ThousandEyes. Available at:

https://blog.thousandeyes.com/deconstructing-great-firewall-china/> [Accessed 8 June 2020].