ENSC 427: Communication Networks

Spring 2017

Final Project Report

Netflix Over LTE Content Distribution Network Optimization

Group 2
https://www.sfu.ca/~kbohlen/
Kurtis Bohlen - 301197502 (kbohlen@sfu.ca)
Dejan Jovasevic - 301142027(djovasev@sfu.ca)
Rohan Thomas - 301195077 (rohant@sfu.ca)

Abstract

In the United States and Canada alone, Netflix has north of 25 million users, accounting for over 30% of all downstream traffic in the US. With so much traffic Netflix employs a system of servers that form a Content Distribution Network (CDN) from which the video “chunks” are cached and streamed to the user. This is in an attempt to reduce the load on the network and improve the customer experience. With LTE providing throughput speeds similar to high speed internet access - the demand for mobile live streamed HD video has increased. We will investigate the performance of the LTE network for streaming Netflix movies to mobile devices and analyze how varying the proximity and design of Netflix’s CDN, called Open Connect, impacts this performance. In our analysis we will look at the bit rates of the transfers, loss of packets, and delays. We will be using Riverbed Modeler to create testing scenarios, in which we will change the number and location of the CDNs video servers servicing both stationary and moving mobile devices.

Table of Contents

List of Figures

1.0 Introduction

The video entertainment industry is constantly evolving and constantly incorporates the latest technologies in the process. Components of this process are providing and improving video streaming services to millions of viewers.

Among the video platforms is an on-demand service: Netflix. With over 90 million worldwide users [1], Netflix is the home of on-demand video streaming for multiple platforms both wired and mobile. 4K quality makes it the ultimate viewing experience on devices connected to high speed networks.

Recently an emphasis has been made on the streaming high-definition Netflix videos over LTE networks. Long-Term Evolution (LTE) is the latest technology deployed in high speed cellular networks allowing for speeds similar to cable high speed internet access [4]. Over 60 countries have adopted the standard globally and the total number continues to grow.

Netflix users interact with web content based on their location. This is all the beauty of Content Distribution Networks (CDN) - where networks are distributed to reduce bandwidth needs, improve response times, and deliver a high quality of experience [3].

To recognize the operation of Netflix over LTE CDN, this paper intends to run specific simulations on Riverbed Modeler (OPNET 18.5) a development environment. This tool contains numerous protocols and technologies - allowing for the modelling of various network types and the analysis of their performance. The goal is to remodel real world scenarios and extract statistics.

1.1 Related Work

There are a few related projects that did dive into the topic of streaming video content over LTE or streaming video content in a CDN. They are as follows:

1. *Video Streaming Over LTE Using Riverbed Modeler* [6]. This is a ENSC 427 group from the course offering of Spring 2016. This group undertook the task of looking at video content that was streamed over LTE. The results that they obtained were relatively different that the ones outlined in this report.
2. *Unreeling Netflix: understanding and Improving Multi-CDN Movie Delivery* [2].This paper focused on analyzing Netflix's early CDN implementations using third party providers (Akamai, Level 3, Limelight). They looked at the network performance and switching between the different CDN servers.
3. *Open Connect Everywhere: A Glimpse at the Internet Ecosystem Through the Lens of the Netflix CDN* [5].This research involved looking at present day Netflix CDN implementation (OCA appliances). Network performance and effects of CDN distribution

were given a deeper look.

These related projects were informative when narrowing down our topic and comparing results as well.

2.0 LTE Network

Long Term Evolution (LTE) is the latest technology deployed in cellular networks and is defined by the 3rd Generation Partnership Project (3GPP). It was invented to be the successor to the 3G standard as a provision for faster user speeds that were not achievable before. This advantage from LTE is due to the emphasis on being a data focused network as opposed to prior circuit switched networks. The fastest speed that users can experience is a peak of 335 Mbps in downlink. But on average it ranges from 12 to 25 Mbps in downlink. During 3 independent tests that were performed, the downlink speeds measured were 121,114, and 96 Mbps respectively.

2.1 Radio Access Network

The air interface, towers, and mobile devices of the LTE protocol make up the Radio Access Network which is called the Evolved Universal Terrestrial Radio Access Network (EUTRAN). Some of the technologies that the EUTRAN uses to achieve such high speeds are Orthogonal Frequency Division Multiplexing in both the uplink and downlink. Additionally there are advances in Multiple Input Multiple Output technologies to allow for more spacial data streams to each device. Beamforming is also used to provide stronger signal to UEs so that higher data rate modulation techniques can be used such as QPSK, 16QAM, and up to 64QAM. The general EUTRAN architecture is pictured in Figure 1 below.

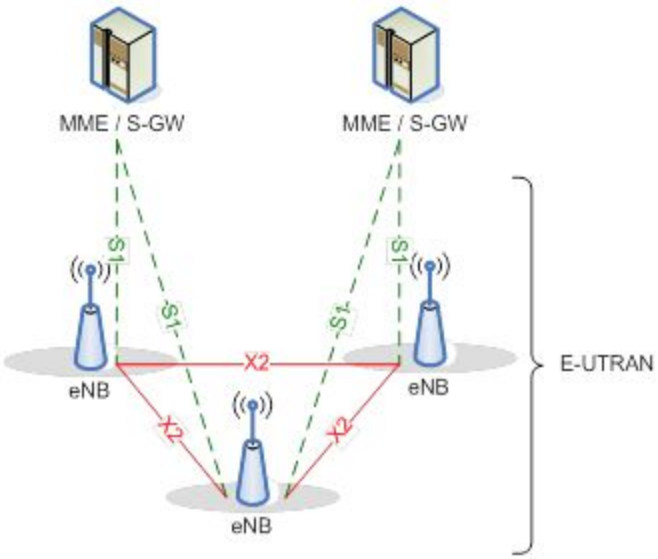


Figure 1 - EUTRAN Architecture

The towers of the EUTRAN are called Evolved NodeBs and are abbreviated as eNodeB or eNB. The eNodeB communicate with the phones or other mobile devices which are called User Equipment (UE). The towers communicate with each other using the X2 interface, and with the Evolved Packet Core, which will be discussed in detail in the next section, by means of the S1 interface. The eNodeBs main responsibility is radio resource management which includes radio bearer, admission, and mobility control, and dynamically allocates resources to the UEs for the uplink, downlink, and scheduling. Other responsibilities include IP header encryption and compression, and routing of these IP packets to the Evolved Packet Core [7].

2.2 Evolved Packet Core

The Evolved Packet Core (EPC) is the latest core network architecture again standardized by the 3GPP. It was designed to support high throughput, low latency support for the LTE EUTRAN, but is also designed to support legacy 3GPP technologies such as the 3G standard as well as non-3GPP technologies such as Wi-Fi. The major difference between the EPC and other core network architectures before it, is that it is a pure IP packet switched network. Whereas Global System for Mobile communication (GSM) was a completely circuit switched architecture and Universal Mobile Telecommunication Service (UMTS) or 3G had circuit switched voice and SMS with the addition of packet switched data network; the EPC is an overhauled evolution of the 3G data network that services data, voice, and SMS for LTE. An illustration displaying the circuit switched (CS) and packet switched (PS) portions of these core network architectures can be found below in Figure 2.

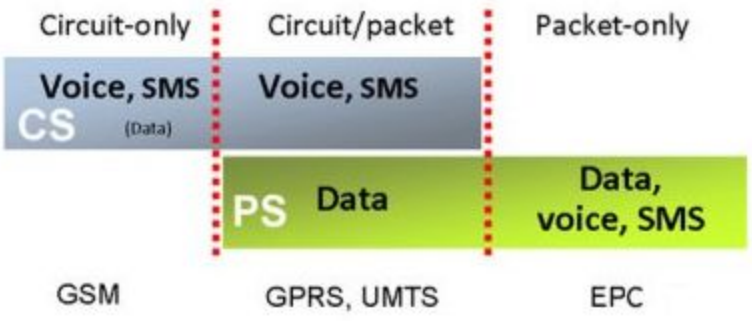


Figure 2 - Core Network Switching Domains

The EPC was designed to have a ‘flat’ architecture, with the idea that it would be able to efficiently handle the data packets in regards to cost and performance with as little nodes and protocol conversions as possible. A graphical representation of this architecture can be found in Figure 3 and this report will go into detail about the EPC nodes immediately following.

Figure 3 - EPC Architecture

2.2.1 EPC Functional Split

The EPC is divided into two parts, the control pane and the user pane. This functional split between the user data and the control signalling is to allow for adaptive network scaling. In this way network operators can easily dimension and scale their network to meet their own specifications.

2.2.2 Serving Gateway

The first node in the EPC that the users data packets pass through is the Serving Gateway or SGW. As such it is one of the nodes in the User Pane. The SGW interfaces the UEs in the radio access network with the EPC. It is primarily responsible for routing of the UEs inbound and outbound IP packets. It is in this node that LTE connects with other older cellular technologies.

2.2.3 Packet Data Network Gateway

The only other node in the EPC in the User Pane that data traffic packets are sent through is aptly named the Packet Data Network Gateway (PDN GW or PGW). This gateway interfaces between the EPC and all external packet data networks ranging from the internet to an operator's IP Multimedia Core Network Subsystem (IMS).

2.2.4 Mobility Management Entity

We now look at the nodes involved in the Control Pane. The Mobility Management Entity abbreviated MME is responsible for all control signaling regarding UE mobility and EUTRAN security. In this it controls paging and tracking of the UEs within the radio access network.

2.2.5 Home Subscriber Server

The final node in the EPC is the Home Subscriber Server (HSS) which is used in the Control Pane. It is essentially a database containing subscriber information that is used for authentication, call setup, and roaming. [8]

3.0 Content Distribution Networks

A Content Distribution Network (CDN) consists of proxy servers set up in specific locations to cache content that is in demand by users. By having content spread throughout various locations users are able to receive higher quality service with less delay and latency [3]. The users can view content directly from servers which are geographically close to them, instead of streaming from one central server [5]. One key contributor to the increase in CDN popularity is the vast increase in numbers of internet users. Another is the demand for video traffic. Video accounts for almost 70% of all downstream traffic today. With video data the threshold for delays and lost packets greatly decreases, so reducing the number of hops a packet has to traverse goes a long way in increasing user experience.

3.1 Netflix Content Distribution Network

Originally Netflix would use third party CDN providers such as; LimeLight, Level 3 and Akamai [1]. These providers would store Netflix movie content in their servers and then stream to users. Of course this would be a recurring cost for Netflix, as no one is willing to stream someone else's data for free. Hence as Netflix grew in popularity it only made sense to start developing Netflix's own Content Distribution Network [5]. The concept became known as Open Connect and Netflix started the project in 2011 [5], and in the following years they had their own hardware unit called an Open Connect Appliance, it is shown below in Figure 4.

Figure 4 - Netflix's Open Connect Appliance [9]

This is netflix hardware unit, these servers are distributed to various Internet Service Providers. They are embedded within the ISPs network so that all of the netflix streaming done by users using that ISP stays within the network. This not only provides these users with high quality content but offloads the backbone of the internet [4]. Now in order to keep the movie data current the Open Connect Appliances are updated with video content in off-peak hours. Thus the number of times video data has to traverse the internet is reduced to one. Figure 5 below shows how Netflix pushes movie content to its OCA's at the edge of the network.

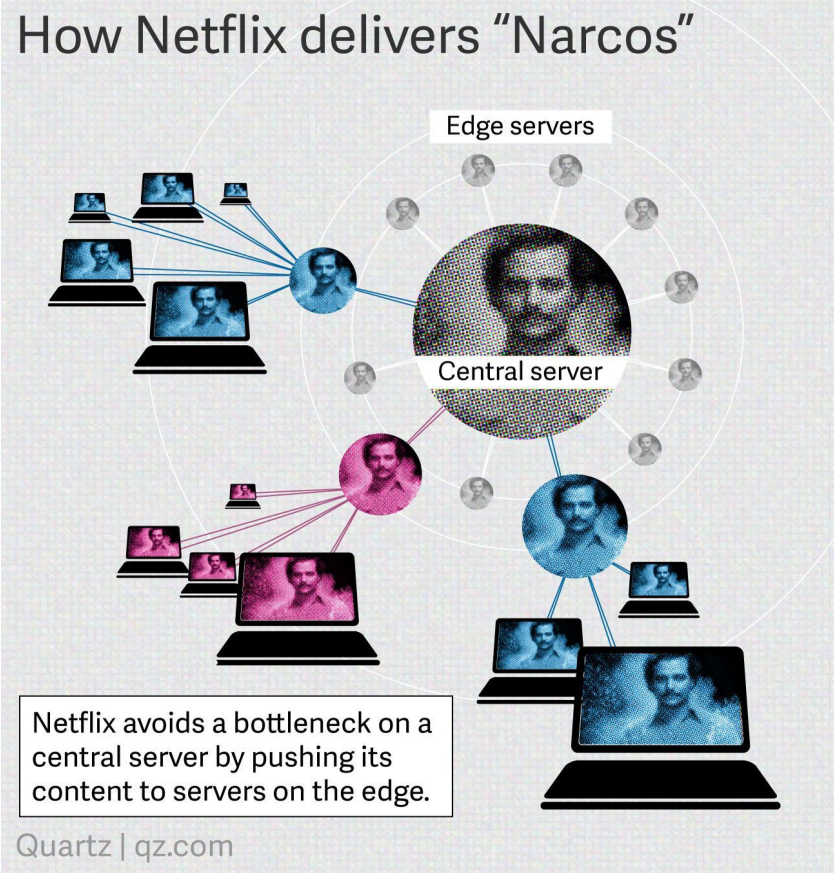


Figure 5 - Netflix pushing content (Narcos) to the end of the network using CDN servers [10]

Now when a user actually logs onto the Netflix website they are interacting with the Netflix service hosted in the Amazon Web Services cloud. The OCAs are in constant communication with Netflix service, giving updates about their "health, routability, and content availability"[8]. When a user picks a movie title, the service determines the OCA most optimal for streaming and provides the URL of this OCA to the user. The user then creates a TCP connection with this specific OCA and the streaming starts. This way all streaming is done within the network. This process is highlighted below in Figure 6.

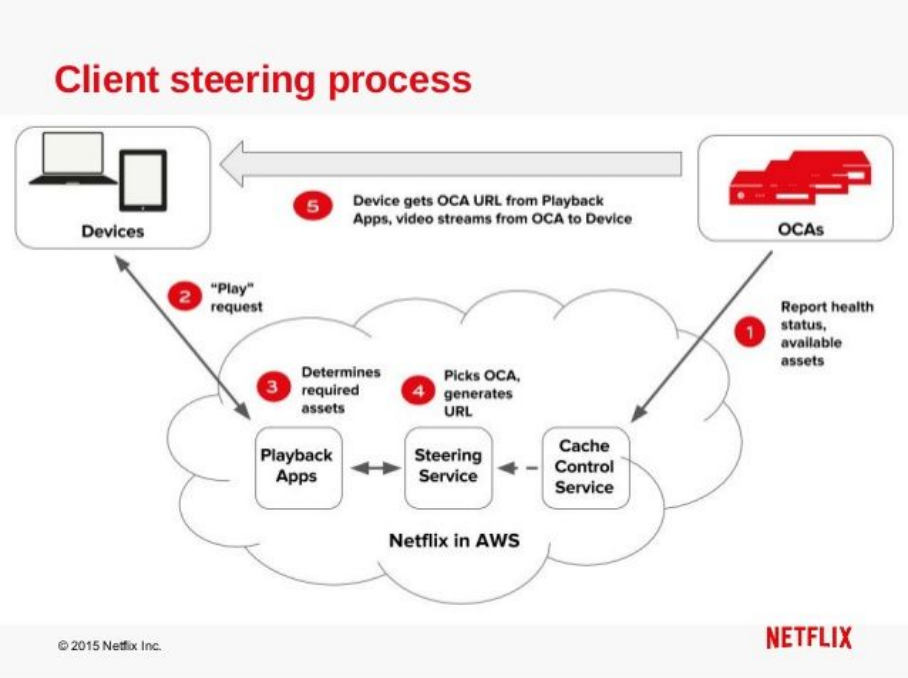


Figure 6 - The Netflix Playback Process [8]

4.0 The Problem Description

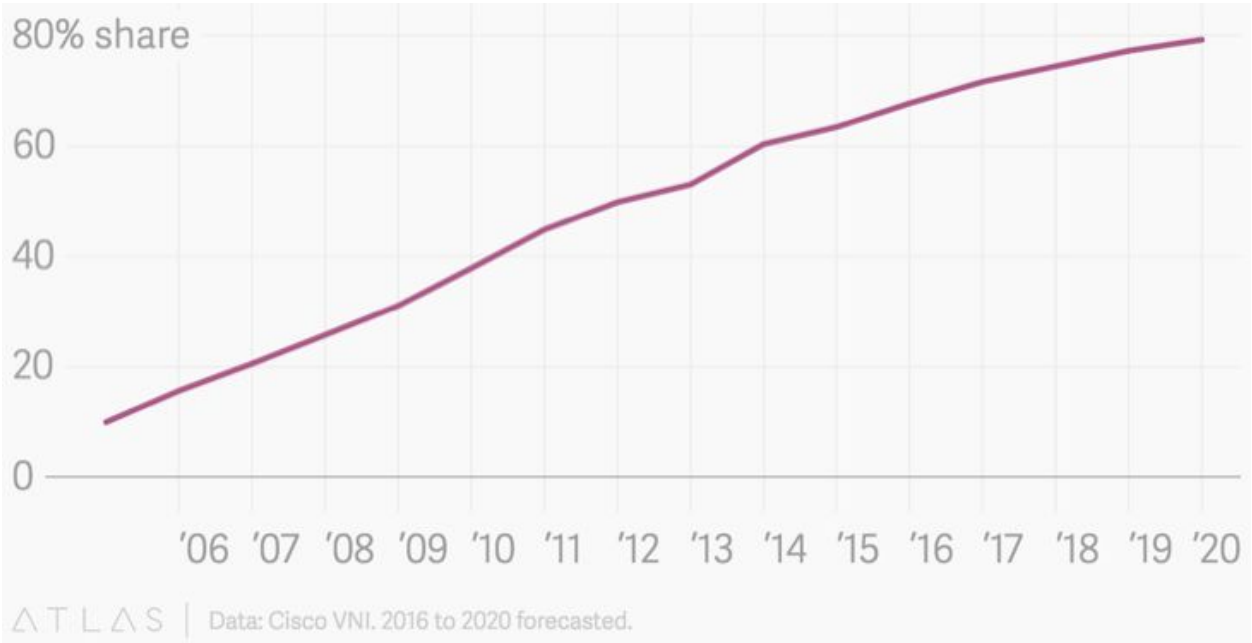


Figure 7 - Video share of Global Internet Traffic [10]

If we analyze the past decade alone, video streaming has become the main source of traffic on the internet. Figure 7 above illustrates the increasing video traffic since 2006 and the estimated

increase in the future years. A significant portion of this content originates from the main content providers – being Facebook, Google (YouTube) and Netflix. Video streaming contributes to ~70% of all traffic today. This is a significant increase from the 12% contribution in 2006. In a recent report, Cisco estimated that this percentage will continue to trend upwards and will eventually reach 90% by 2020 [10].

To supply and satisfy the increasing demands of users, content providers pushing video content to the end of the network. What this means is that they are alleviating the need for video to traverse the entire backbone of the internet to reach users. The major advantage of this method is that it reduces the total number of hops that it takes content to travel from server to client To reaffirm this concept we used the Wireshark analysis tool to analyze Netflix streaming over a local TELUS network. This is seen in Figure 8 below:

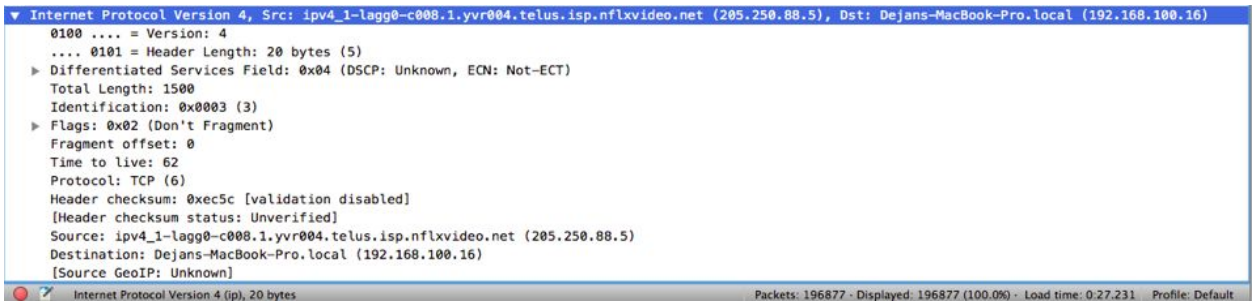


Figure 8 - The Wireshark capture of streaming High-definition video content

High definition Netflix video content was streamed on a laptop that was connected to the TELUS DSL network. While streaming a video it was observed in the simulation that it goes to a local TELUS CDN server in Vancouver. The actual name of this server is the ipv4_1-lagg0-c000.1.yvr004.telus.isp.nflxvideo.net, thus we can also deduce from this name that it is the 4th CDN server in Vancouver. During the simulation, the observed round trip time was ~6ms. The Vancouver CDN server that it went through was one of many CDNs distributed globally – this one had the lowest calculated travelling cost by the Netflix service.

In 2007 Netflix began its video streaming service. Over the years as its user base grew, the corporation needed to push its video content closer to its users to enhance the overall user streaming experience. Netflix slowly moved away from third party providers and created their own CDN network (OpenConnect Concept 2012 onwards). During the initial stages of the OpenConnect deployment a clash arose between Netflix and the major Internet Service Providers (ISPs). The ISPs refused to incorporate Open Connect Appliances since the content providers were taking up most of the bandwidth with video content. Subsequently the big ISPs refused to incorporate CDNs and this gave room for the smaller ISPs to take advantage in the market and they started installing OpenConnect hardware in their networks. As Netflix continued to expand and demand increased, the smaller ISPs began to gain market traction in the process. The large ISPs later realized this trend and were forced into using Netflix OpenConnect to offer the same quality of service as the smaller ISPs. Ever since Netflix has

been deploying CDN Servers around the world with the focus of providing faster and better quality video service to millions of users. The other content providers such as Facebook and Google have been contributing to this growing trend of expanding CDNs by installing CDN servers all around the world. This CDN expansion is causing a phenomenon referred to as the “Flattening of the Internet”[10]. The internet is being flattened as the most demanded content no longer needs to traverse the backbone of the internet to reach users. High demand for their content in turn means high profits for the provider's which allows them to buy up hardware to stream their content to users. Now although this is a positive aspect in the eyes of customers who want high quality content delivered quickly, it is decreasing the open concept of the internet. Content provider's are trying to take over “the last mile” and provide content directly to users without the need for ISPs [10]. Of course this is only possible since we as users demand content from a select few provider's; Facebook, Youtube, Netflix.

Over the years as video content has become the staple for most internet users – it has also influenced their decision process when it comes to choosing ISPs. For instance, Netflix now releases ISP rankings via their ISP Index (Figure 9).

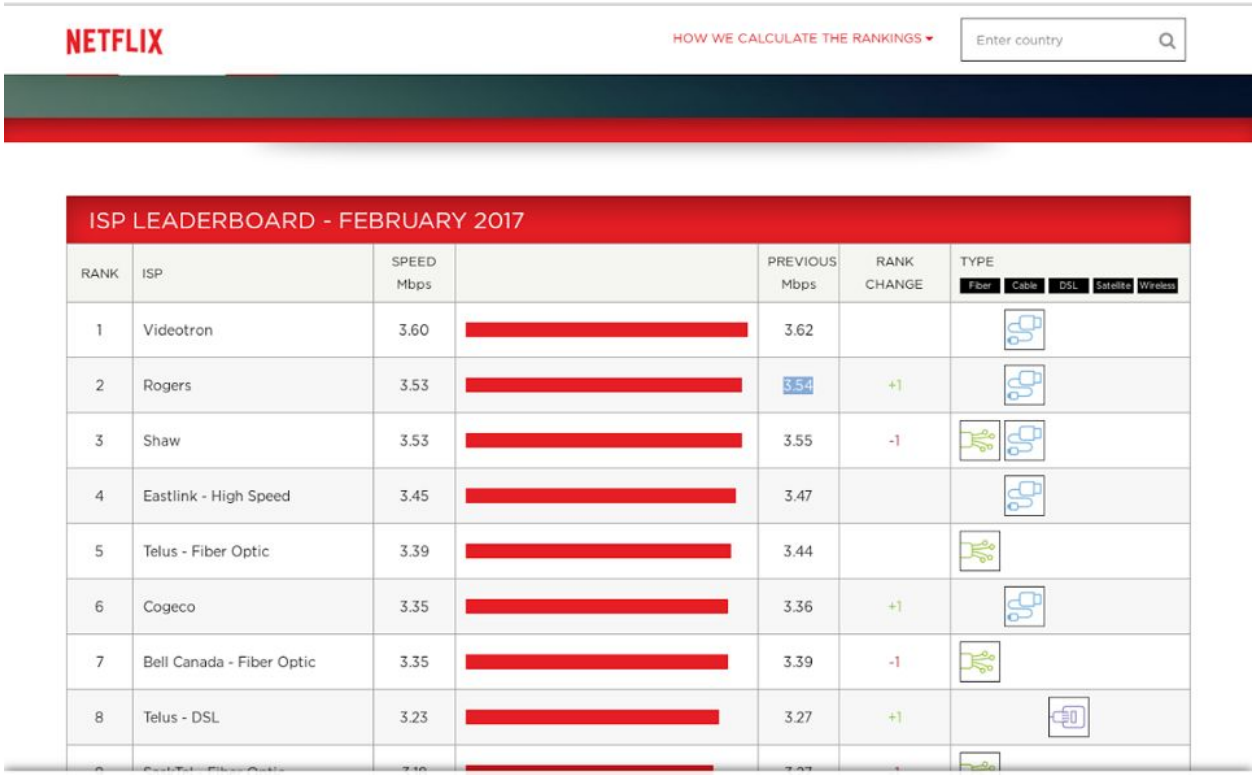


Figure 9 - Netflix ISP Ranking for Canada (February 2017) [11]

This leaderboard rates ISPs based on their streaming ability at that given point of time. This influences users to pick a service provider based on their streaming ability unlike before. Again this is showcasing the power that content providers like Netflix have.

5.0 Design

5.1 LTE Network

Our overall was to measure the performance improvements when a CDN was used to stream video content to mobile users on an LTE network. To do so we first had to create an LTE network that would be able to provide internet packets to mobile users. To achieve this we used the Riverbed Modeler option of deploying a wireless network topology. This allowed us to configure the number of eNodeB towers in a network, the radius reached by each tower cell, the number of mobile phones per cell and the number of sectors a cell was divided into.

In order to have simulations that completed in a timely manner (~4-5hrs) per scenario we scaled back our implementation to the following:

- 1 ENodeB tower
- Cell divided into 3 sectors
- 1 LTE Iphone in each cell
- Radius of cell 1km
- 1 Evolved Packet Core per Network

The results of the configuration are shown in figures 10(a) and 10(b).

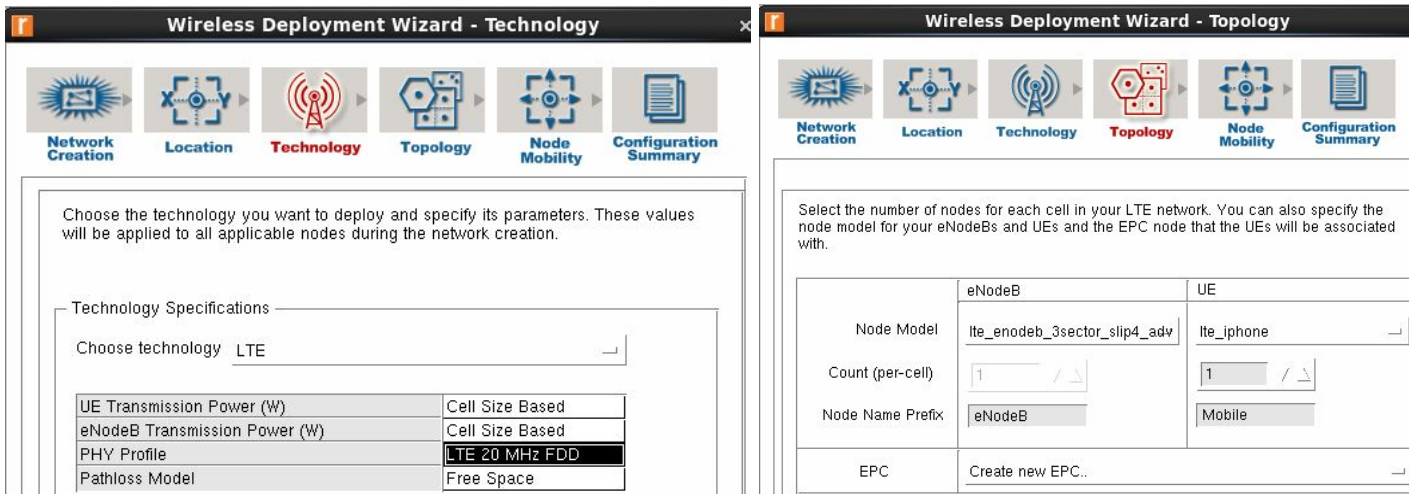


Figure 10(a) and 10(b) - Wireless Deployment Wizard configurations

We also set 20MHz Frequency Division Duplex for the physical profile, which means that the transmitter and receiver will use different radio frequencies for communication. Figures 11 and 12 show the final configuration and the overall layout of one LTE network which we used.

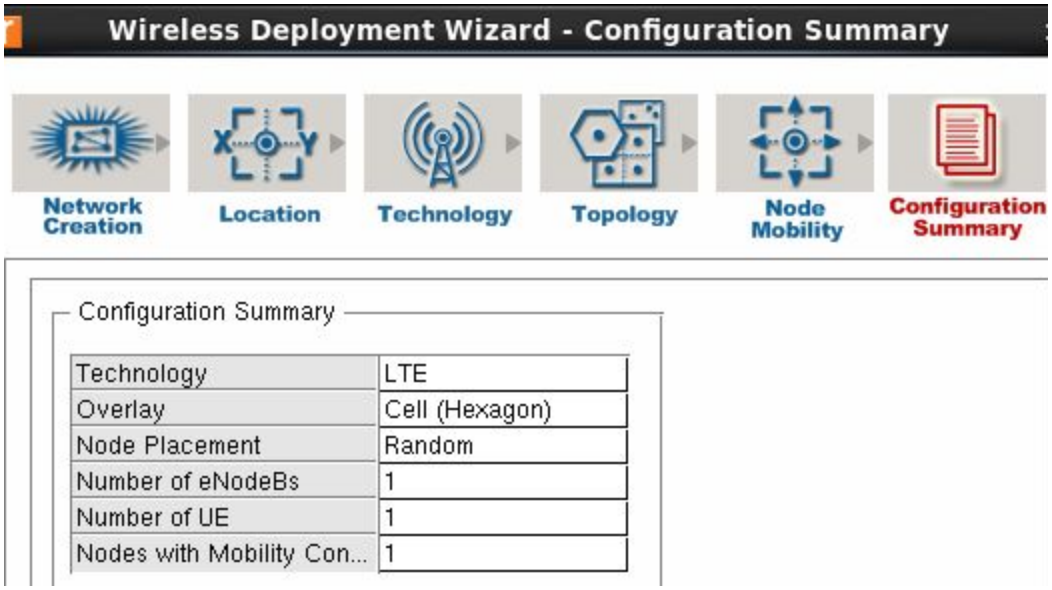


Figure 11 - Wireless Deployment Wizard configuration summary

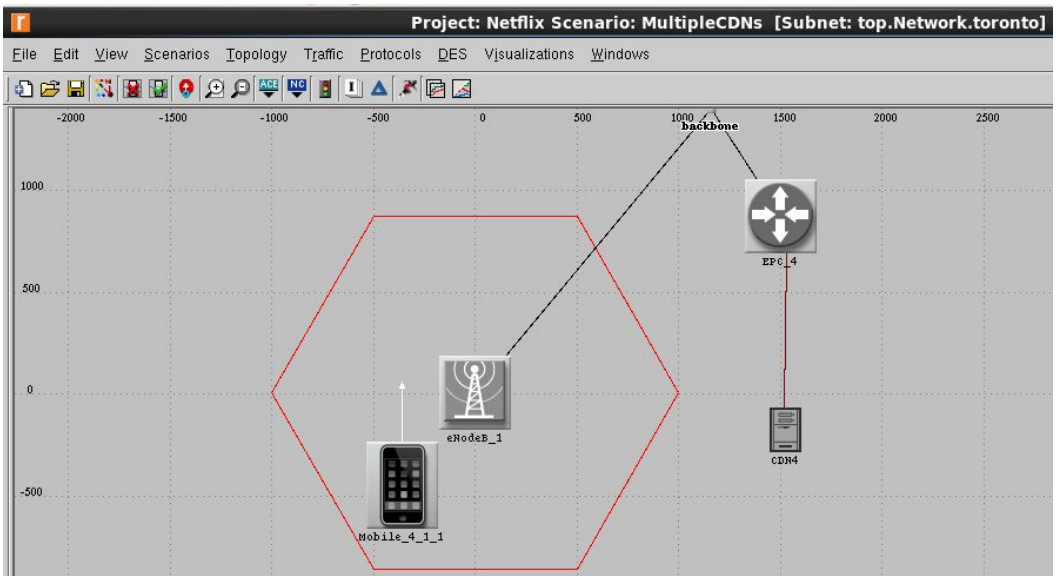


Figure 12 - Overall layout of one LTE network

As can be seen in Figure 12, we have one EPC within each network. The EPC is the connection between the LTE network and the internet.

5.2 Multiple Scenario Configuration

Now that we had the LTE network configured we could continue to develop different scenarios to test the effects CDNs had on performance. First off we created 6 subnets, each one containing the elements shown in Figure 12. The subnets where located in each of the following cities: Vancouver, Calgary, Saskatoon, Thunder Bay, Toronto and Ottawa in Canada. For our purposes we needed 3 configurations.

- Configuration 1: One Central Server providing data to all six subnets. Server was located in Denver, Colorado.
- Configuration 2: Two Servers each providing data to closest 3 subnets. East server located in Louisville, Kentucky. West server located in Salt Lake City, Utah
- Configuration 3: Each subnet has its own server

Configuration 1 represents the video data from the netflix server traversing the whole internet to reach the mobile users who are streaming video. This is the exact thing that CDNs are trying to get rid of, so this was considered our base case. Configuration 2 can be seen as a small netflix Content Distribution Network, for example when they were still using third party CDN providers. The movie content doesn't have to travel from one source to reach the users within a wireless provider's network but the situation is still not optimal. Finally configuration 3 represents a wide reaching CDN, each Internet Service Provider has a Netflix server within their network from which the users can directly stream data. The server located within the LTE network shown in Figure 12 is meant to represent the Open Connect Appliance embedded within the ISPs network. Now not shown in the Figure is the connection between the OCA and the central Netflix servers or the Amazon Web Services cloud. Figures 13, 14, and 15 showcase the three different configurations we used in this project.

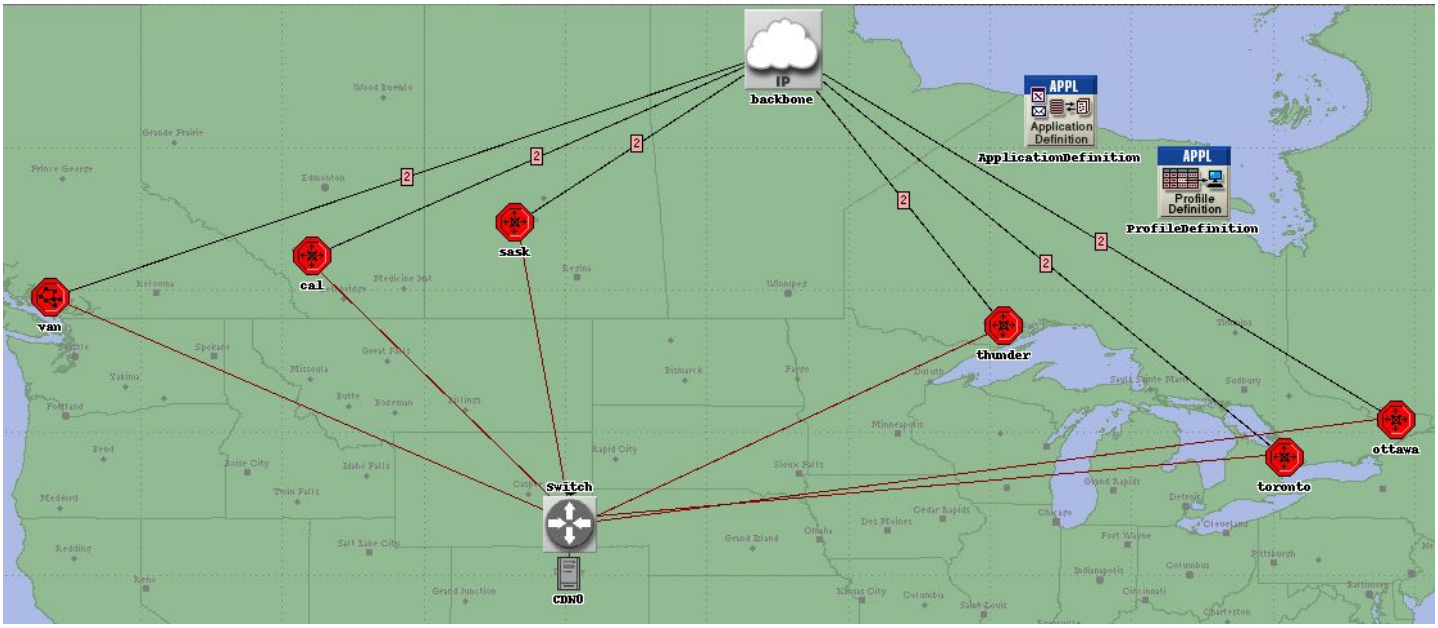


Figure 13 - Scenario 1: One Central Server in Denver, Colorado providing data to all six subnets.

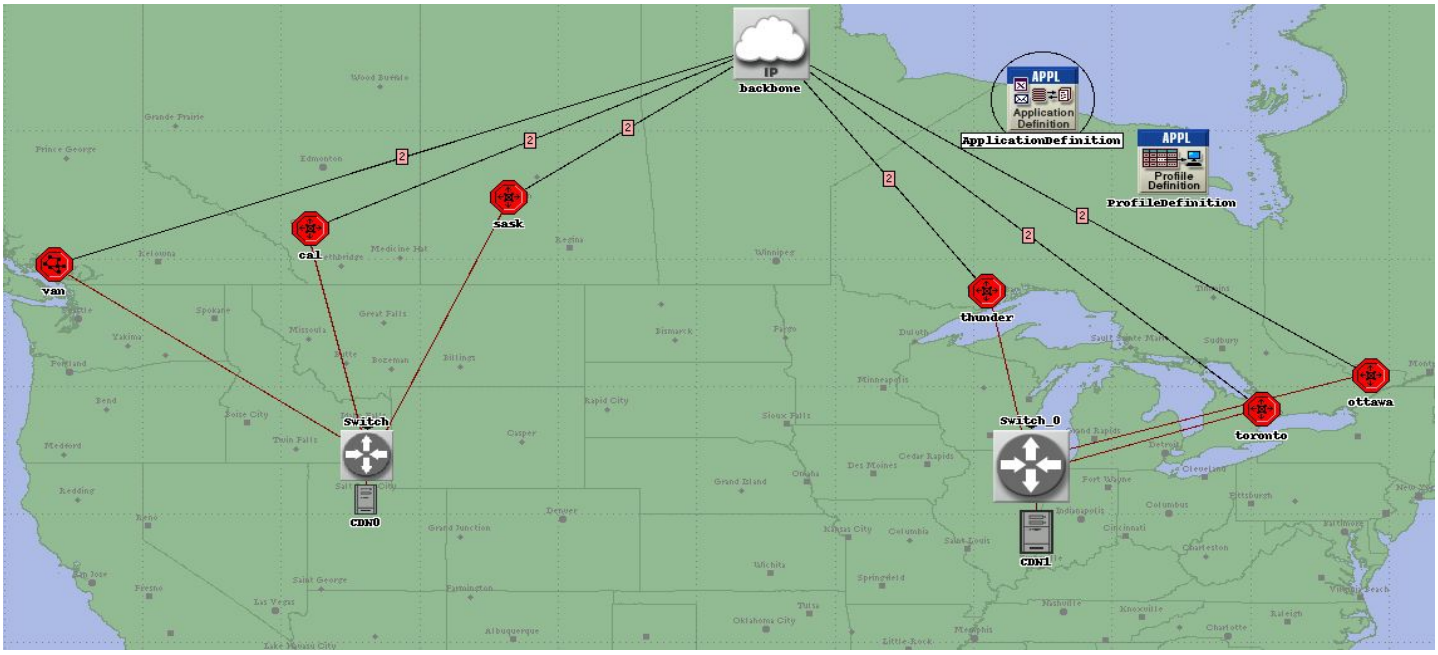


Figure 14- Scenario 2: Two Servers (Louisville, Kentucky and Salt Lake City, Utah) each providing data to closest 3 subnets.

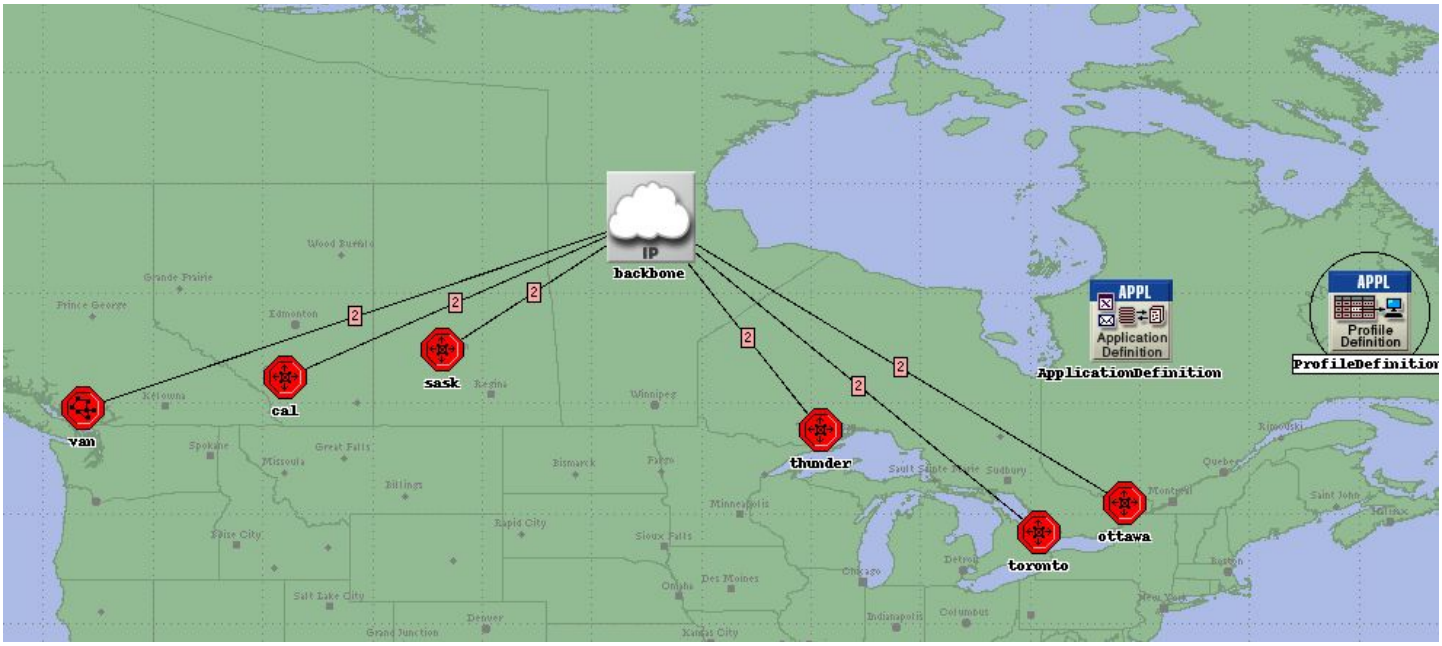


Figure 15- Scenario 3: Each subnet has its own server

With the scenarios created, the next step involved defining an application for our servers to run that would mimic that of an Open Connect Appliance. Thus the first steps entailed finding out what kind of protocols were being used by Netflix.

5.3 Dynamic Adaptive Streaming Over HTTP Application Configuration

Netflix uses a special version of Hyper Text Transfer Protocol for its application layer. This protocol is called Dynamic Adaptive Streaming over HTTP, referred to as DASH. DASH is built on HTTP but differs from it on both the client and server ends [12]. When a movie is stored on the server broken up into several smaller segments. The movie itself is encoded at various qualities, thus variable bit rates. Higher bit rates translates to higher quality segments [12]. Thus on the server one could imagine a scenario as depicted in the diagram shown below.

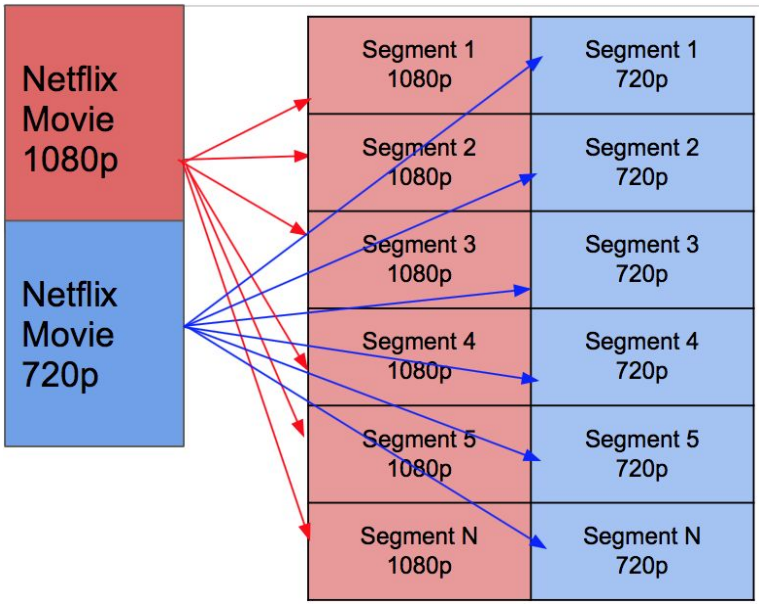


Figure 16 - Netflix video content being translated based on the respective encoded bit rates of the video

Now the movie is sent to the user in segments, where the user makes a request, analogous to an HTTP request and then one or two segments are sent before the next request is made. However, there is another interesting aspect of DASH which is taking place on the client's machine [13]. While the client is performing playback of video content, DASH is measuring the strength of the network connection and thus calculating the highest possible bit rate of the next segments to be downloaded so playback can occur without the need for buffering[12]. For example if one's connection suddenly becomes very poor, DASH will decide to download lower segments encoded at lower bit rates, which will mean lower quality picture but time spent buffering will be reduced. Now since DASH delivers data just like an HTTP web server would we know that it uses Transmission Control Protocol in the transport layer. Therefore to successfully model a Netflix CDN server we needed to find the correct frame interarrival times as well as frame sizes for the movie segments being sent to the client. Upon finding these parameters we could configure an HTTP server to transmit data to the client's at that rate.

In order to find these parameters we decided to stream Netflix over an LTE Iphone 6s and then use Wireshark to measure the size of the frames coming in and the rate at which they were arriving. Thus we created an LTE hotspot via the Iphone, connected to this hotspot on our

laptop, turned on wireshark and began to stream a Netflix movie. The Wireshark capture is shown in Figure 17 below.

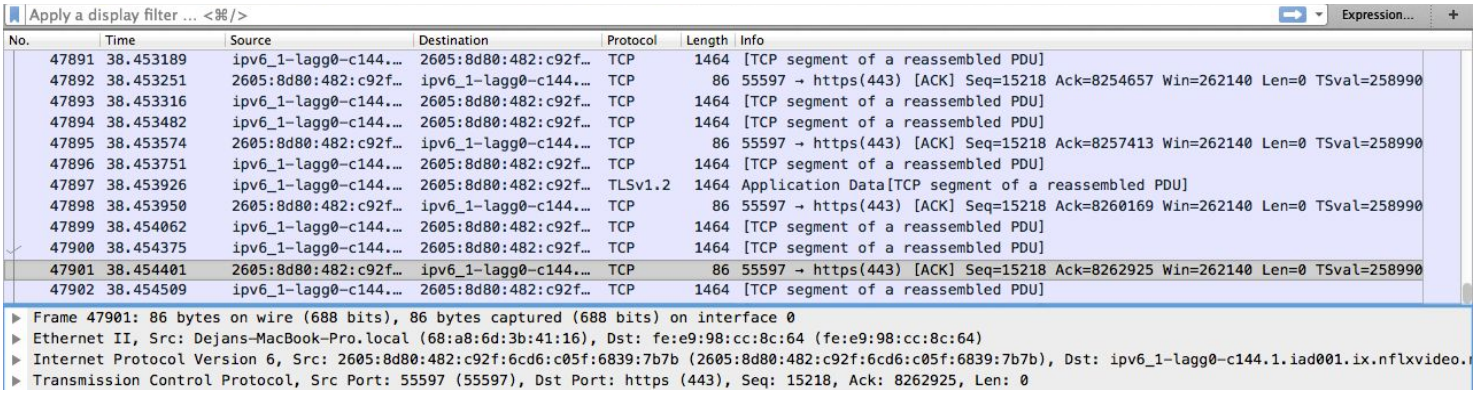


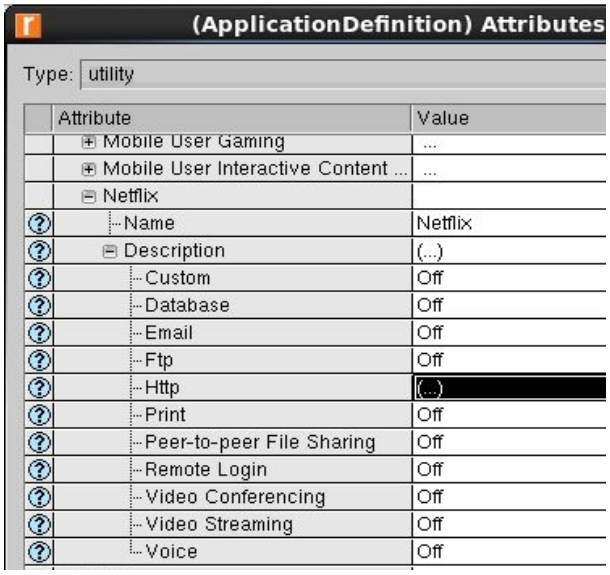
Figure 17 - The Wireshark capture of streaming Netflix video content over a Iphone 6s LTE connection

As can be seen from the screencap, the Iphone is using IPV6, TCP and the destination port of the Netflix server is port 443 which is the HTTP Secure port. The source port is the ephemeral port on our laptop which was port 55597. Now using this Wireshark screen capture we were able to come up with the following parameters.

- Frame Interarrival Time: 0.58 MilliSeconds
- Frame Size: 1464 Bytes
- Average Download Rate: 2.62 MegaBytes/Second

It was not necessary for us to model the exact behaviour of a DASH client and server, configuring an HTTP server with these parameters is what was necessary for our performance measurements.

We first configured an application, which we called Netflix to be an HTTP application. The application was provided with a Frame Interarrival time of 0.58 milliseconds and the average frame size was a constant 1464 bytes. The configuration is shown below in figures 18(a) and 18(b).



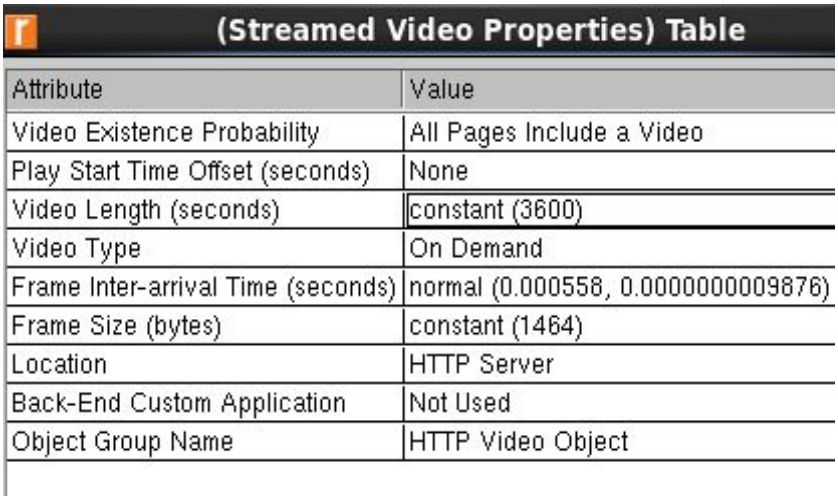


Figure 18(a) and 18(b) - Netflix application definition configurations

The frame Interarrival time was given a normal distribution with the standard deviation being 0.9876 nanoseconds, which was also calculated from the Wireshark capture. Our simulation was one hour long so we configured the video length to be 3600 seconds. Afte configuring this application we had to make our servers support this application. This was done by configuring the attributes of the server as shown in Figure 19.

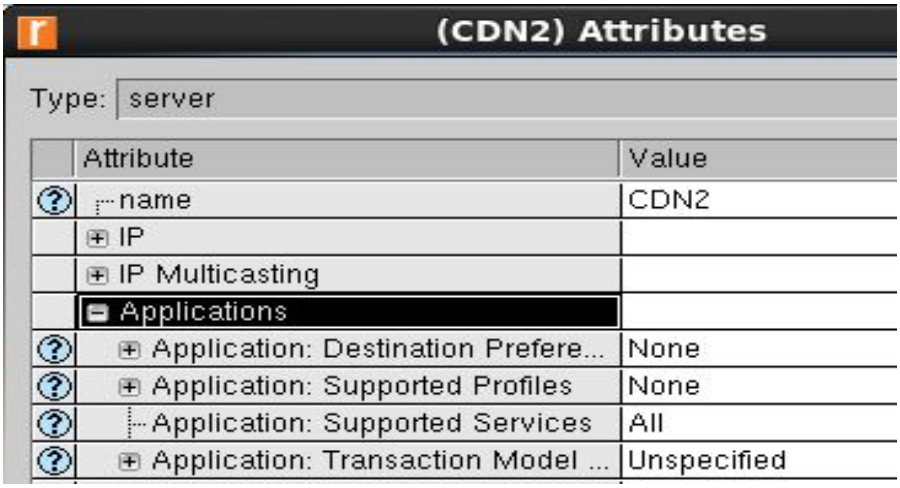


Figure 19 - Server attribute configuration

5.4 Mobile Profile Configuration

Here the server is configured to support all application services, one of which is the Netflix application service we had just defined. Now the CDN servers were configured so the next step involved configuring the profiles for the mobile phones to use. A profile can be used to model

the network usage behaviour of users, where each profile can be using multiple application services. For our purposes we needed to configure only one profile which would be using the Netflix service we had defined. After we had defined a profile we set each of our mobile phones to support this profile. Figures 20 and 21 show these steps.

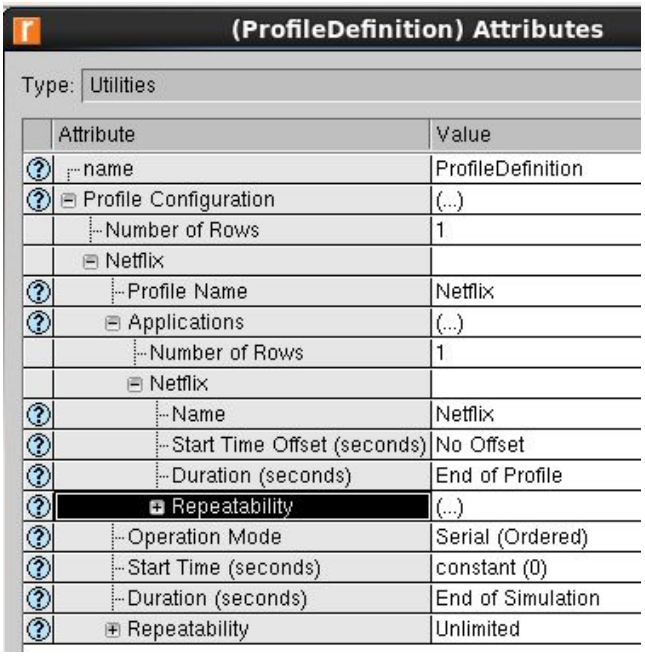


Figure 20 - Profile definition configuration

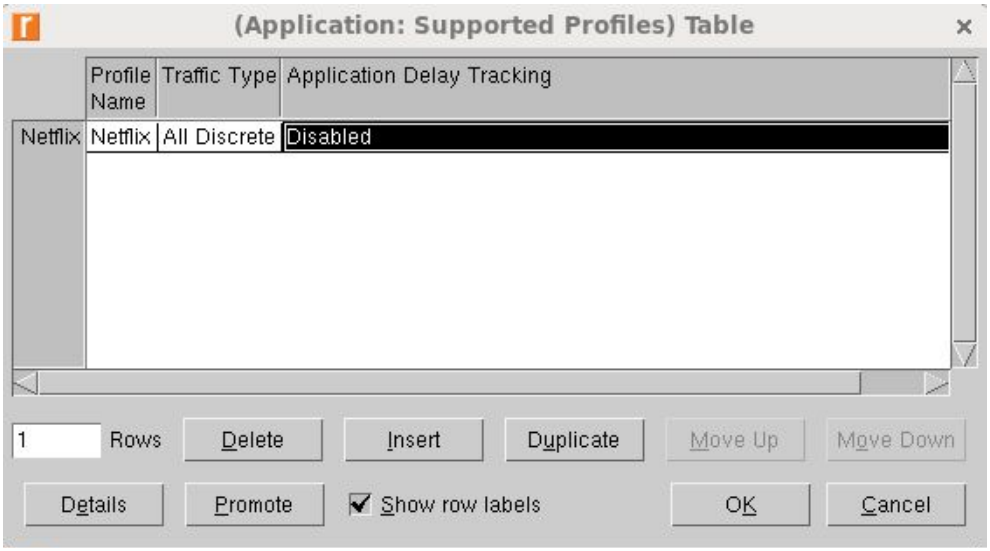


Figure 21 - Mobile phone configuration to support the new profile

The profile we defined as also called Netflix and was engineered to run for the duration of the simulation with no offset. This meant that the user would be streaming movie content for the entire hour the simulation was run. Each mobile phone was then set to support the previously defined Netflix profile. We had to ensure to turn of Application Delay Tracking as this would

collect massive amounts of data that would in turn cause our user space to run out of memory and the computer to crash.

After configuring all the aforementioned entities we were ready to actually run our simulation. We wanted to measure the throughput on the links leaving a CDN server, the queuing delay from the EPC to the respective CDN server and finally the bit error rate.

6.0 Simulation

We felt that a one hour simulation would be enough to gather sufficient results about the performance improvements with expanding CDNs. Thus we configured our DES settings to simulate for one hour for each of the three aforementioned scenarios. The 3 scenario simulation collected over 9 billion events of data. Figure 22 shown below shows the traffic received by the mobile phone in one of the subnets, Ottawa in our case. In the previous section it was shown that a LTE Iphone should be receiving roughly 2.6 MBps of data when streaming Netflix. Looking at the graphs displayed in Figure 22 we can see that we are getting around that value in the traffic received in bytes. The green line represents the received traffic by the Iphone in the Ottawa subnet for Scenario 1. The blue line represents scenario 2, and the red line represents scenario 3. As is expected the graphs for the three scenarios are almost identical, which is as expected.

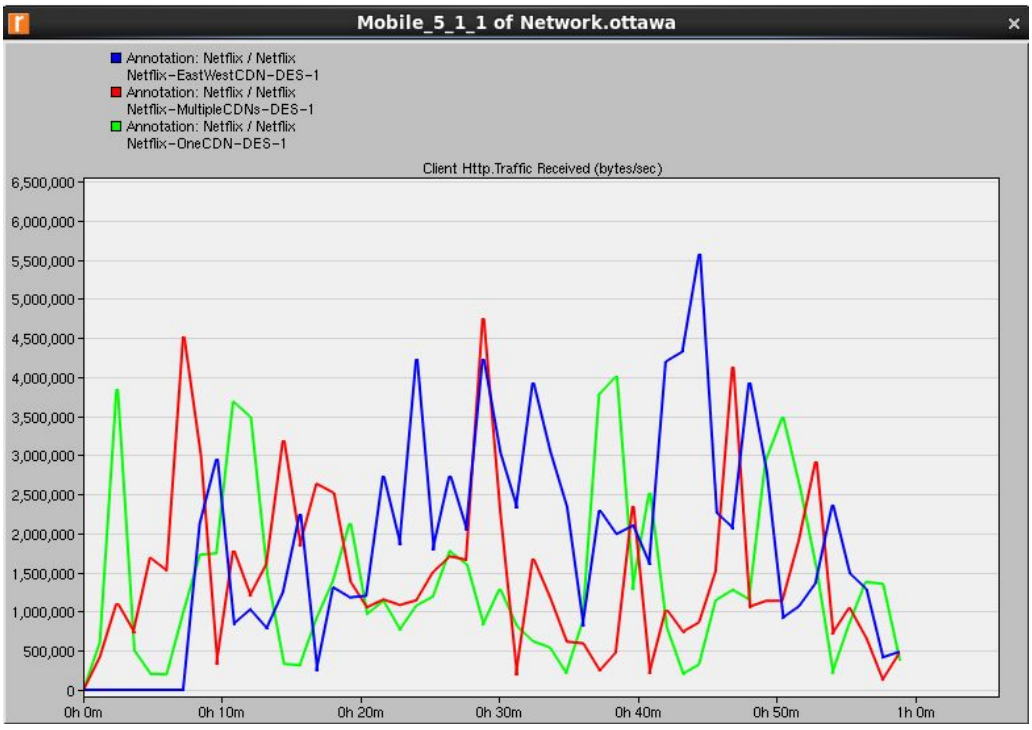


Figure 22 - Traffic received by the mobile phones in Ottawa, Canada

Having confirmed that the traffic received was relatively equal for each phone, the next step was to look at the throughput on the links. We gathered the throughput on all the links connecting either servers directly to EPCs or connecting a server to a switch which then spanned out to various EPCs. The graph shown in Figure 23 shows the data for scenario 3 in blue, scenario 2 is red and scenario 1 is green. The graph for scenario 3 is displaying the data for the connection between the server to the EPC in the calgary subnet. For scenario 2 it is displaying the throughput for the connection between the west CDN server, located in Salt Lake City and the respective switch.

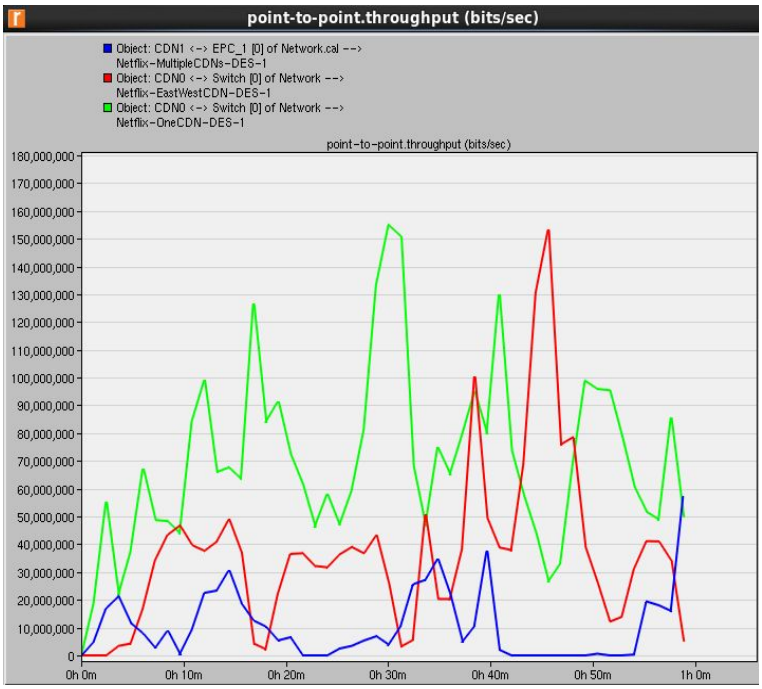


Figure 23 - Throughput in bit/sec for each scenario

Figure 24 below shows the time average values of the throughput for each link with the same color distribution as the above graph.

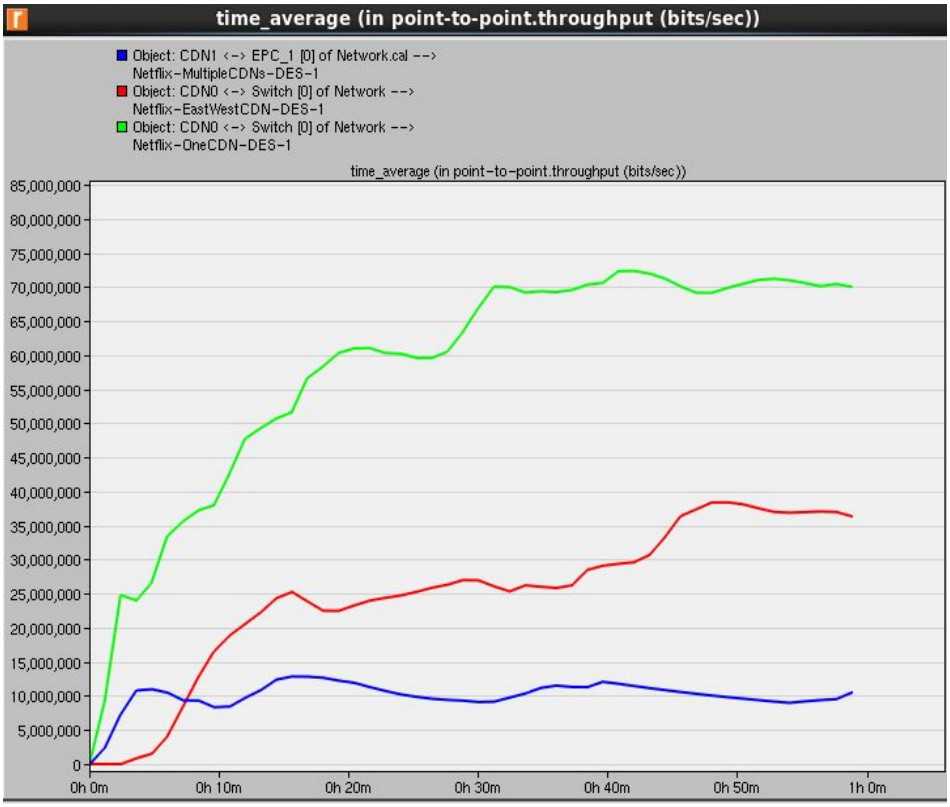


Figure 24 - Time average values of the throughput for each scenario

The time averaged graphs nicely show the correlation between expansion of the content distribution network and the decrease in throughput on the links. Scenario 1 has the highest throughput with the averaged values reaching a max of 72 million bits/second. This makes sense as here there is only one server providing content for each of the mobile phones located in Canada spread subnets. Scenario 2 reaches a peak time average of 39 million bits/second. Here we have an East and West server providing content, thus the throughput is almost halved. Finally for scenario 3 where there is a server located within each subnet, representing the distributed OCAs to different ISPs, we have a time average throughput maximum at roughly 13 million bits/second. These results are exactly what we would expect to have happen, and directly support Netflix's push to expand their Content Distribution Network. They are offloading the amount of traffic passing through the backbone of the internet. So not only are they providing better quality service to their client's, they were able to "reduce the overall demand on upstream network capacity by several orders of magnitude" [8].

We see similar results in terms when measuring delay. Figure 25 below shows the time average of the delay in seconds for the link between the EPC and the CDN server for each scenario.

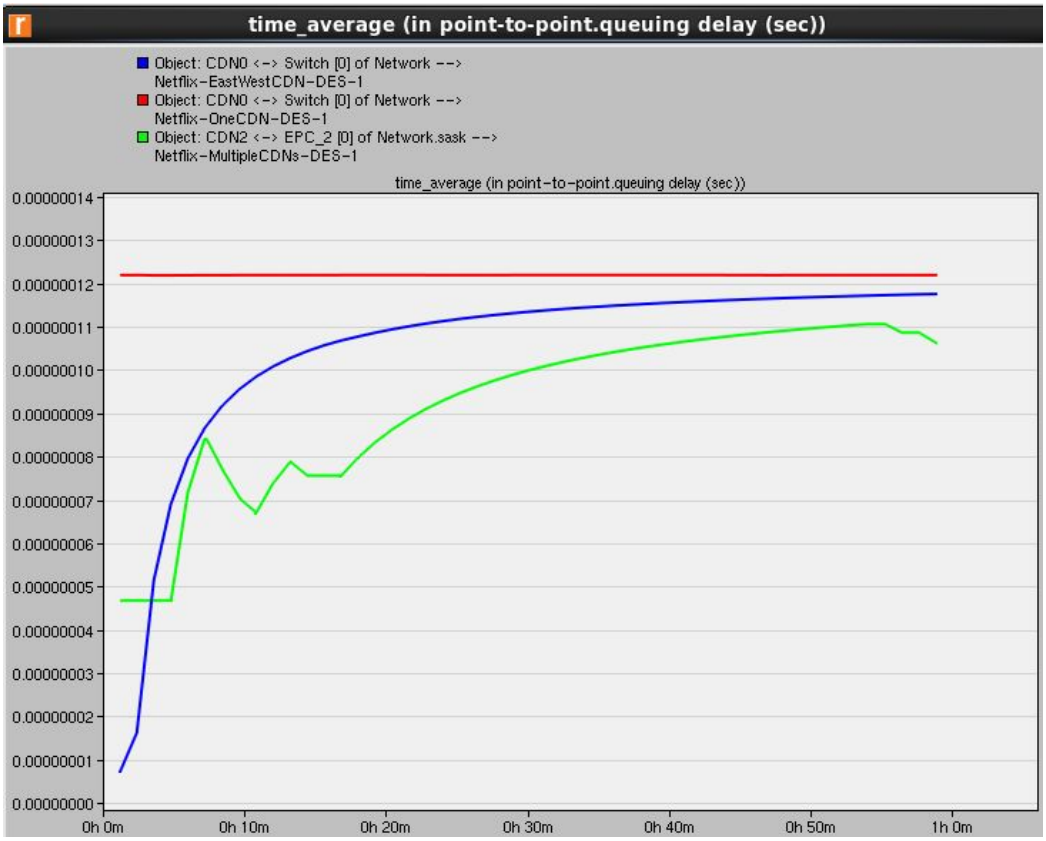


Figure 25 - Time average values of the delay

Scenario 1 has the highest and worst performing delay out of all 3 scenarios as the red line. This is the delay when we only had one CDN server. We can clearly see a decrease in the delay as we add more servers in our CDN. The blue line is the delay from Scenario 2. The green line, which had the best delay performance of all 3 scenarios is from Scenario 3 when we had a well distributed CDN with the highest number of CDN servers. From this we conclude that as you add more servers to your CDN and the more distributed it becomes the delay decreases.

Lastly we will look at bit error rate performance. We collected bit error rate results for the same links as the delay, the link between the EPC and the CDN servers for each scenario. Unfortunately the bit error rate could not be realistically modeled due to how scaled back our LTE network had to become for our simulations to run in a timely manner. Since we are not coming anywhere close to the full utilization of the links we are using in our simulation we would expect to see an extremely low bit error rate. A graph of our bit error rate results can be seen below in Figure 26.

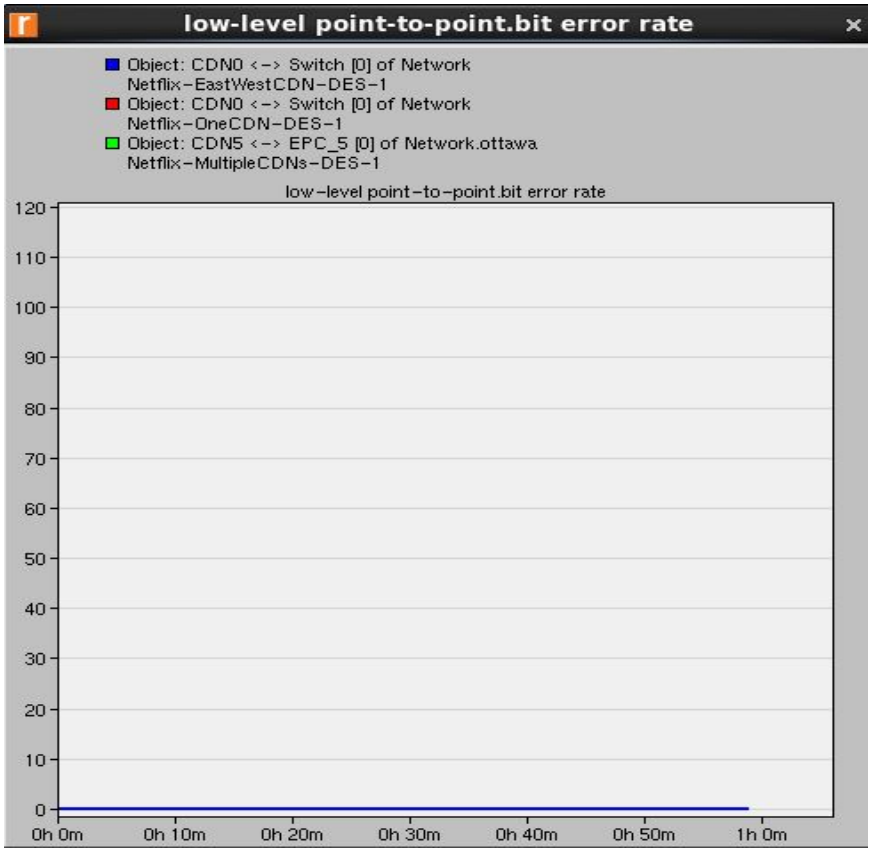


Figure 26 - Bit error rate

7.0 Future Improvements

A few recommendations for improvement are as follows:

1. Generate background traffic and calls on LTE radio access networks
2. Generate background internet traffic on EPC
3. Simulate larger LTE networks with more phones and more towers
4. Have different radio access network configurations for urban and rural areas by varying these parameters:
    a. intersite distance
    b. cell radius
    c. pathloss model
    d. number of users
5. Analyze the different stream speeds – HD compared to non-HD or even UltraHD
6. Be able to distinguish between popular cached content and less popular content that takes longer to buffer and stream
7. Create more accurate models of the Netflix CDN servers and the links

8.0 Conclusion

This project explores the streaming of Netflix video content over LTE and CDNs. This was done using Riverbed Modeler 18.6 to simulate On Demand video streaming over a Content Distributed Network. The three scenarios emphasized the importance of using CDN servers to distributed content based on location. Having more CDN servers lowers the load on the CDN. The ever-increasing demand for video content is accelerating the deployment of CDNs globally and is resulting in the flattening of the internet. This mass deployment of CDNs is critical to lower bandwidth and delay. This in turn improves the user's overall video experience. The more distributed a CDN is – the better it performs. Wireless providers will have to embed the Open Connect Appliances in order to meet the demand for streaming Netflix over mobile devices.

9.0 References

[1] V. K. Adhikari, Y. Guo, F. Hao, M. Varvello, V. Hilt, M. Steiner, and Z. Zhang, "Unreeling netflix: Understanding and improving multi-CDN movie delivery," in Proc. IEEE INFOCOM, Orlando, FL, 2012, pp. 1620-1628. doi: 10.1109/INFCOM.2012.6195531
[2] V. K. Adhikari, Y. Guo, F. Hao, V. Hilt, Z. Zhang, M. Varvello, and M. Steiner, "Measurement Study of Netflix, Hulu, and a Tale of Three CDNs," IEEE/ACM Transactions on Networking, vol. 23, no. 6, pp. 1984-1997, Dec. 2015. doi: 10.1109/TNET.2014.2354262
[3] J. Summers, T. Brecht, D. Eager, and A. Gutarin, "Characterizing the workload of a netflix streaming video server," in Proc. IEEE International Symposium on Workload Characterization (IISWC), Providence, RI, 2016, pp. 1-12. doi: 10.1109/IISWC.2016.7581265
[4] C.D. Cranor, M. Green, C. Kalmanek, D. Shur, S. Sibal, J.E. Van der Merwe, and C.J. Sreenan, "Enhanced streaming services in a content distribution network," IEEE Internet Computing, vol. 5, no. 4, pp. 66-75, Jul/Aug 2001. doi: 10.1109/4236.939452
[5] T.Böttger, F. Cuadrado, G. Tyson, I. Castro, and S. Uhlig, Open Connect Everywhere: A Glimpse at the Internet Ecosystem through the Lens of the Netflix CDN, eprint arXiv:1606.05519 [cs.NI], 2016, pp. 1-14.
[6] Z. Amer, R. Kieu, and L. Xiao, “Performance Analysis of Video Streaming over LTE using Riverbed Modeler.”
[7] Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN). 36.300. Release 8. Juha Korhonen. April 2017.
[8] Network architecture. 23.002. Release 1999. 3GPP. January 2015.
[9]Netflix, “Open Connect Overview,” Netflix Open Connect, 13-Jan-2017. [Online]. Available: https://openconnect.netflix.com/Open-Connect-Overview.pdf. [Accessed: 03-Mar-2017]
[10] J. I. Wong, “The Internet is Flat- The internet has been quietly rewired, and video is the reason why,” Quartz, Quartz, 05-Oct-2016.
[11] “Netflix ISP ranking,” Netflix ISP Speed Index. Netflix, 28-Feb-2017.
[12]“MPEG-DASH,” Encoding.com, 23-Mar-2017. [Online]. Available: https://www.encoding.com/mpeg-dash/. [Accessed: 01-Apr-2017].

[13]“Mpeg-dash vs Apple hls vs Microsoft smooth streaming vs Adobe hds,” Bitmovin, 05-Mar-2017. [Online]. Available: https://bitmovin.com/mpeg-dash-vs-apple-hls-vs-microsoft-smooth-streaming-vs-adobe-hds/. [Accessed: 01-Apr-2017].