A Framework for Generating HTTP Adaptive Streaming Traffic in ns-3

William David Diego Maza

A Framework for Generating HTTP Adaptive Streaming Traffic in *ns-3*

William Diego
Orange Labs
44 Avenue de la République
92320 Châtillon, France
william.diego@orange.com

ABSTRACT

Video streaming is today one of the most relevant service in mobile Internet, which represents around 50% of total mobile data traffic. Consequently, researchers perform huge efforts to design and propose new mechanisms and architectures to improve video streaming in mobile networks. In this regard, simulation is a relevant step to test and validate those new mechanisms and models. ns-3 is one models of the most widely used network simulator due to its rich library of network and its vast user community. Despite of its relevance, ns-3 lacks of realistic traffic source. This paper presents a HTTP Adaptive Streaming traffic generator framework for mobile networks in ns-3. Its design and validation are presented, as well as some possible evolution directions and future works.

Categories and Subject Descriptors

I.6 [**Simulation and Modeling**]: General, Model Development, Model Validation and analysis; C.2.2 [**Computer Networks**]: Applications—*YouTube traffic model, performance measures*

Keywords

Video Streaming, HAS, YouTube, ns-3 simulator, Internet traffic

1. INTRODUCTION

Video streaming is growing faster and today represents a large fraction of global Internet data traffic. Recent traffic reports as [2], [7] and [1] show that video streaming accounted for around $40-60\%$ of fixed data traffic and accounted for 50% of mobile data traffic. HTTP Adaptive Streaming (HAS) video is primarily driven by over-the-top (OTT) providers as YouTube, which is the dominant actor of video streaming in many regions and typically accounting for $50-70\%$ of total video traffic volume.

Due to the prohibitively high costs of a physical communication network (e.g. servers, routers, etc), simulation is an important stage for network traffic research as it provides the necessary tools to study new protocols and models, at least during the initial stages. Thus, there exists an obvious critical need for realistic traffic generators. In this sense, ns-3, which is an open-source simulator, proposes a large range of resources to study communication technologies, protocols and models. ns-3 provides great flexibility and today is widely employed in the academic research community. Nevertheless, ns-3 is relatively new and today it proposes some basic traffic generator models (e.g. on-off traffic model). In some cases some works propose interesting traffic models to ns-3 as those described in [3, 5, 15].

In this paper we propose a framework for generating HAS traffic in ns-3 and evaluate its performance. Our framework imitates YouTube traffic behaviour and is based on recent works related to mobile YouTube traffic characterization.

2. BACKGROUND AND RELATED WORK

2.1 YouTube Characteristics

Most popular video streaming services (i.e. YouTube, Netflix) use HAS. HAS, split up media (i.e. video) into a series of small files called chunks, which are then encoded using different video qualities as shown in Figure 1 [12]. Each chunk is transmitted individually as a single web object via plain HTTP. In the course of playout of the video, the client continuously assess available bandwidth and requests successive chunks for the data rate that can be supported. Typically, the client keeps a buffer of chunks to deal with eventual network issues (e.g. latency, packet loss, connection loss).

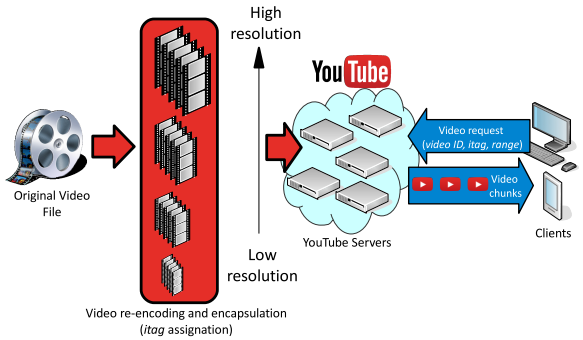

Figure 1: YouTube video service

In order to provide compatibility with all browsers, devices, bandwidth and quality requirements, a wide range of encoding for every video file is available for clients, which can be selected according needed. Besides, a numerical identifier named "*itag*" is used in order to identify different encoding schemes of a video. The *itag* information is included in the HTTP requests. Moreover each chunk is transmitted individually as a single web object via plain HTTP. During the playout of the video, the client continuously estimates available bandwidth and requests chunks for the data rate that can be supported as shown in Fig. 2. The client try to keep a buffer of video data to deal with eventual network issues (e.g. latency, packet loss, connection loss), in order to perform it, a buffering strategy is necessary.

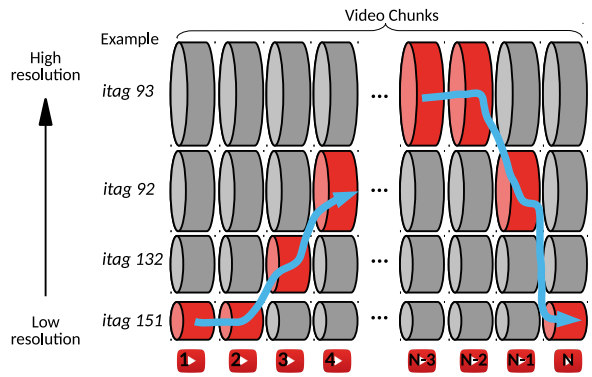

Figure 2: YouTube video streaming strategy

2.2 YouTube Buffer Management

The video buffering strategy is a key element on video streaming because it could permit the optimization of network resources (i.e. bandwidth, radio resources) or in the worst case reduce the wastage of it. Below is a brief description of most used video buffering strategy, which are described in [8, 12–14]:

- **Standard buffering:** In standard buffering strategy the device receives all video traffic (i.e. chunks) and keep data in the buffer until it is full. At this point, playback of the video starts and the video application tries to keep the buffer full along the video playback, in this regard the device will request to server the needed video data. In case the network condition are disturbed and the instant bandwidth goes below the value required for the video, the buffer data is used to fill the gap. In worst case when the buffer is empty, the video is interrupted until new video data are buffered.

- **Dual-threshold buffering:** In case of mobile networks, the standard buffering strategy is vulnerable to bandwidth drops, as well as being unable to exploit a increase of bandwidth. Dual-threshold buffering strategy is more flexible and try to fill the gaps of standard buffering strategy. It provides a resilience to data rate fluctuations or other adverse conditions associates to nature of wireless media.

  In the dual-threshold buffering strategy an initial buffering is performed before the playback of the video, which consists of filling a first threshold B_{min} (*lower threshold*) in the buffer. In this strategy, instead of trying to keep the buffer full to this min level B_{min}, it tries to fill the buffer to a second higher level B_{max} (*upper threshold*). These additional video data will be useful if the network connection encounters temporary impairments. In worst case when the buffer is empty, the video playout can start after a short pre-loading time. Also, in case of an increase of data rate, a bigger buffering of video chunks to counteract the possibility of network malfunctions can be performed. Details of dual-threshold buffering are shown in Figure 3

2.3 YouTube Traffic Models

We can find several YouTube measurement studies in literature. Much of them were focused on characterizing various aspects of YouTube videos, as well as its usage patterns and its strategy depending on supported hardware and software of terminals. But only few of these studies are focused in mobile YouTube traffic characterization.

In [10] authors analyze YouTube and Netflix video streaming using iOS and Android devices over wireless networks (WiFi, 3G and LTE). They found that when a client requests a video, the resolution is selected based on the device types (screen size), regardless of OSs on the devices or access networks. They also found that video players frequently terminated the TCP connection and open a new TCP connection to continue receiving the video content. The number of TCP connections varies depending on the playback buffer management policies of the video players running on different OSs. In [11] the above authors propose dynamic QoS-aware rules for LTE networks to select an appropriate video resolution under a fluctuating channel condition, in order to reduce the waste of the video content and enhance the QoE for end-users. They also found that YouTube uses a single TCP connection, which is opened and closed along the video streaming.

In [4] authors present an empirical study of the performance of YouTube in cellular networks. Is showed that the complex and dynamic CDN (Content Delivery Network) architecture of YouTube has better service performance in video over cellular networks in terms of improved QoE (delay and throughput) and user engagement compared to other CDNs providing HTTP video streaming.

In [9] authors analyze and compare the performance when Android and iOS devices are accessing Internet streaming services.

Is very hard to identify a valid reference of YouTube traffic characterization, because it is constantly evolving. In this paper we try to present an state-of-art of YouTube and highlight the most important evolutions that can impact its traffic characteristics. Moreover, we study the mobile ecosystem because is relative few explored field and additionally all our studies are focused on it. In this paper, we propose and evaluate a YouTube traffic generator based on models presented in [12, 14] and [16], which propose a YouTube traffic model described below.

Each application instance is composed by a video server that streams data via a TCP connection to a video client. The chunk duration and codec is 5 seconds. The client device uses a playlist information which provides several different profiles of video quality levels and is identified by the *itag* values, for example those presented in Table 1. These profiles is used in order to choose the appropriate quality, according to the network and the devise capabilities.

itag	Resolution	Encoding rate
132	426 x 240	266 kbps
92	426 x 240	395 kbps
93	640 x 360	758 kbps

Table 1: YouTube video quality information

At the beginning of the communication, the device requests a chunk with the lowest video quality ($itag = 132$, 266 kbps, 426 x 240), after that the device estimates its data rate based on the received chunk, it automatically selects the highest playable video quality and sends the YouTube server a request message. The ratio between the *Throttling Phase* throughput and the encoding rate is referred as the *throttling factor*. In [12] was found that mobiles use a throttling factor of 2.0, for encoding rates higher than 200 kbps. It is also showed that some terminals use a dual-threshold buffering strategy and others a standard buffering strategy, we implement the first one because the second one can be easily emulated using a high B_{max} and fixing $B_{min} = B_{max}$.

The server first sends an *Initial Burst*, corresponding to $35s$ seconds of video data, before to start playing the video. When the amount of data in the player buffer exceeds approximately $100s$ of video, the client aborts the TCP connection. The download is interrupted for approximately $60 - 70s$. When the amount of data in the player buffer falls below approximately $30s$, the terminal opens a new TCP connection to request to the server the next video segment. This behavior is repeated until the full video is downloaded. This threshold strategy avoids wasting data if the user aborts the video playback. The upper and lower threshold that we use are $100s$ and $30s$, respectively, but these values can be modified. We also establish a maximum timeout for chunk request in order to avoid errors during simulation, we fix this parameter at $30s$. After that the timeout is exceeded, the video session is stopped and it starts a new video session.

3. YOUTUBE TRAFFIC MODEL IN NS-3

This section describes the design of our YouTube traffic generator for ns-3. We have implemented a specific module in ns-3 in order to emulate the delivery of mobile YouTube traffic. For this purpose, we have taken advantage of the YouTube traffic model described on [12, 14] and [16]. We base our ns-3 implementation on the transactional traffic generator presented in [5].

3.1 Basic Structure

As shown in Figure 4, the `YoutubeClient` and `YoutubeServer` applications are responsible for the major functionalities, such as generating the adaptive traffic, handling HAS processes, as well as recording and process statistics. When the mobile YouTube model starts, the `YoutubeClient` and `YoutubeServer` applications are installed in client and server nodes, respectively. Both applications start a new TCP connection; then the evolution of the communication between the `YoutubeClient` and `YoutubeServer` is described in Figure 5.

We also define two main attributes, `VideoSize` and `Timeout`, which define the video duration and the maximum time to wait a video chunk before to restart the video session respectably.

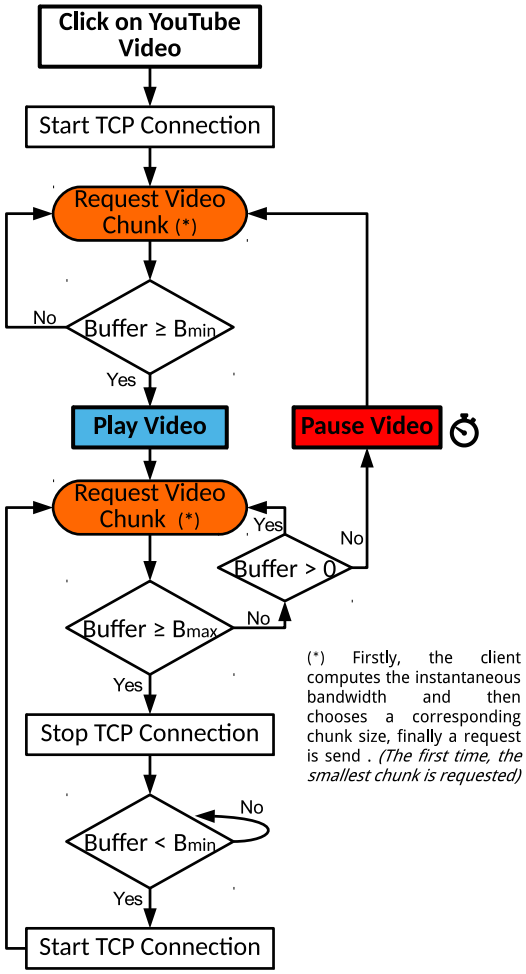

Figure 5: Flow description of our YouTube traffic generator

4. VALIDATION AND RESULTS

This section verifies and validates the YouTube traffic generator framework implementation in ns-3.

4.1 Framework Validation

In order to validate our model we perform an unitary simulation, where 2 User Equipments (UEs) are randomly placed in a LTE cell. The first UE performs a YouTube session and the second one performs a FTP download. The simulation parameters are detailed in Table 2. The bandwidth was set-up to 3Mhz (15 RBs) and the duration of the whole video is fixed at 250 seconds.

Figure 6 shows the data at player instantaneous (i.e. `Youtube Client`), which is captured using wireshark. It shows the typical behaviour of YouTube session, the first part is the *Initial Burst* where the UE buffers the initial chunks before to playout the video. In the second part the *Throttling Phase* where the UE requests needed video chunks according to dual-threshold buffering strategy in order to avoid stalling.

4.2 Simulation Example

In this section, we present simulation results and performance evaluation of our proposed framework.

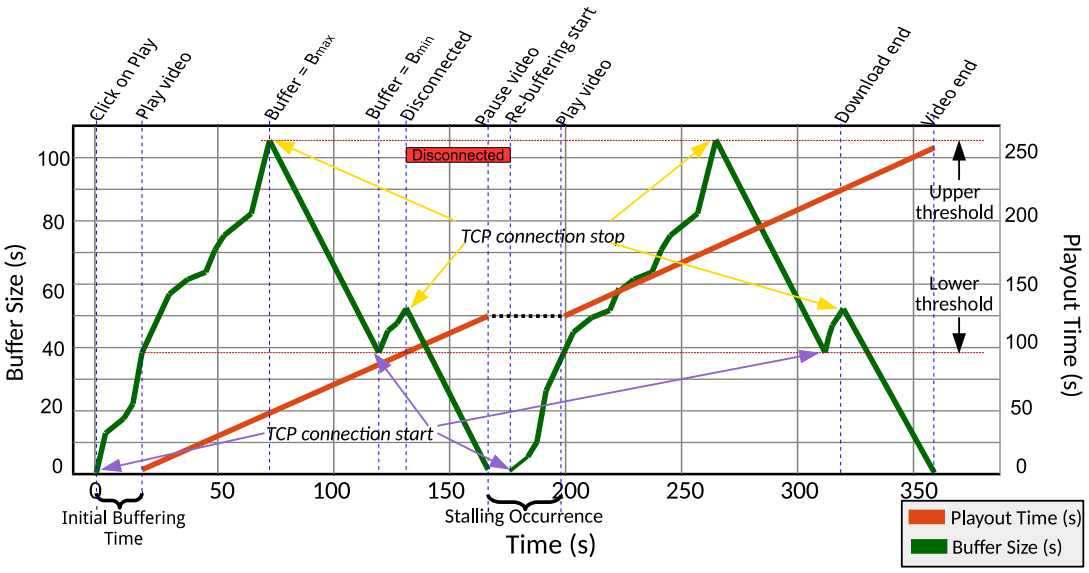

Figure 3: Dual-threshold buffering strategy Playout time and Buffer state illustration

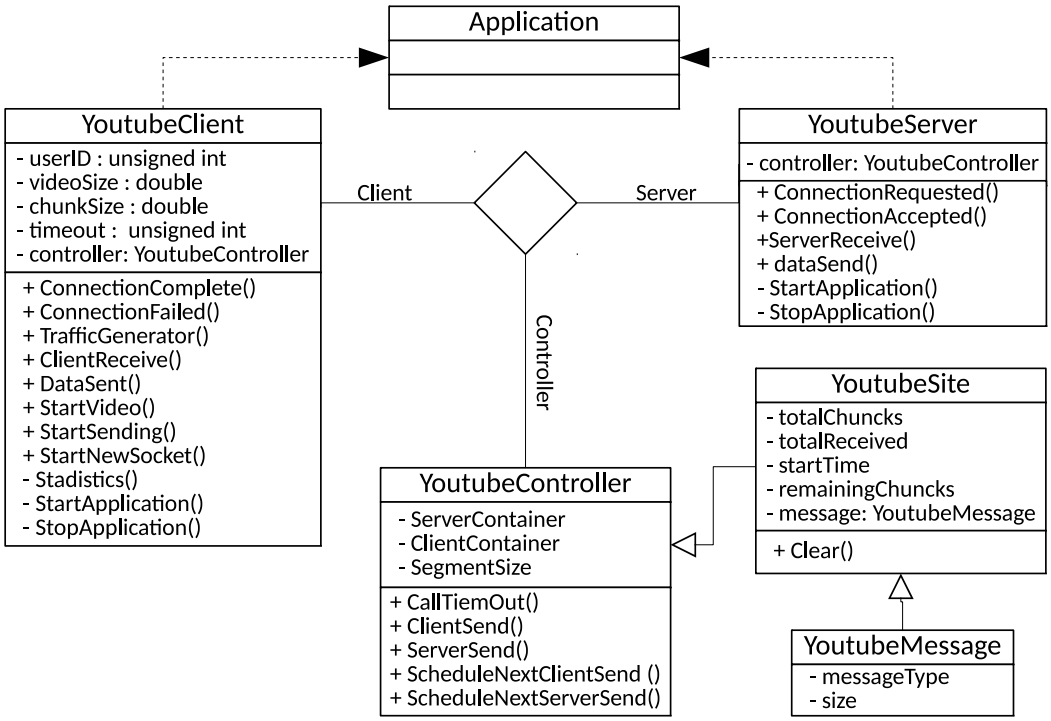

Figure 4: YouTube class diagram

4.2.1 *Scenarios and metrics*

We have evaluated the proposed YouTube traffic generator framework. We have considered a typical outdoor scenario with 10 UEs (random positions) attached to a single eNB (evolved Node B), thus inter cell interference is not taken into account. The eNB is equipped with an omnidirectional antenna and UEs experience varying channel conditions. Given the path loss model and the other network parameters, we obtained a wide range of SINR values, which provided Channel Quality Indicators (CQI) values in range of [1, 15].

At the beginning of each run, UEs are placed randomly in a disc representing the cell within a distance range of 30-500m. Then, UEs move within the disc according to a Random Walk Model, at a fixed speed of 3 km/h. The simulation parameters are shown in Table 2 and the system configuration is as follows: The cell is connected via the PDN Gateway (P-GW) to the internet. A server is implemented for FTP and Youtube. The server is connected to the P-GW via an over-provisioned point-to-point link in order to avoid congestion on this segment of the network.

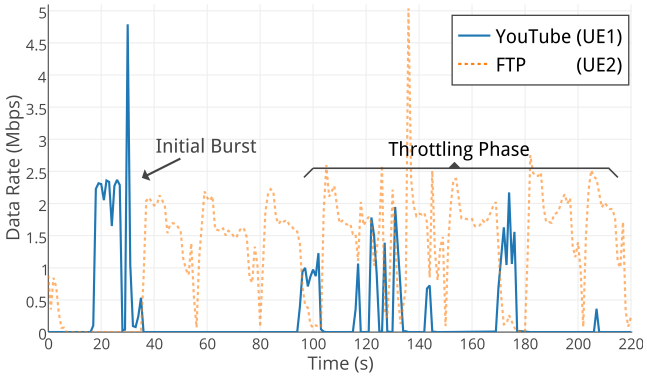

Figure 6: Time evolution of received data by a `Youtube Client` *(ns-3 YouTube framework test)*

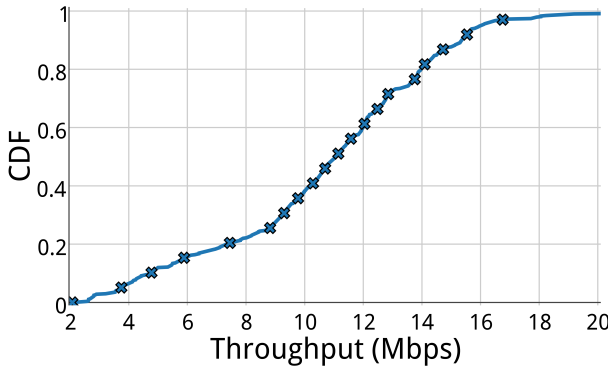

Figure 7: Cumulative Distribution Function (CDF) of the cell throughput

Users mobility model	RandomWalk (3 km/h)
Bandwidth	50 RB (10 MHz)
Cell coverage radius	500 m
Pathloss Model	Cost231
eNB TX Power / Noise Figure	46 dBm / 5 dB
UE TX Power / Noise Figure	24 dBm / 5 dB
Fading loss model	EPA 3 km/h (urban)
AMC model	PiroEW2010
DL/UL carrier frequency	2120 / 1930 MHz
RLC Transmission Mode	UM
RLC Buffer Size	100 kbytes

Table 2: Simulation parameters

4.2.2 *Traffic Description*

The 7 first UEs use YouTube and the 3 others UEs perform FTP sessions. In case of FTP, the size of downloaded files follows a uniform law between $[1, 5]$ Mbytes. The arrivals of new FTP session follow a Poisson process with an average inter-arrival time $\lambda = 10$ seconds.

We have set a maximum timeout for YouTube chunk request to avoid blocking situations during simulation, and we have fixed this parameter at 30 seconds. When the timeout is exceeded, the video session is stopped and a new video session started. The duration of the whole video is fixed at 120 seconds with an exponential inter-video interval with 10 seconds of mean.

4.2.3 *Simulation Result and Analysis*

We present here performance results related to the scenarios described previously. The simulation run lasts for 600 seconds, with a warm-up time of 5 seconds where statistics are not collected, and is replicated three times with different seeds. Applications are started at a random time uniformly distributed in $[1, 5]$ seconds.

Figure 7 depicts the cumulative distribution functions (CDF) of cell throughput in the analysed scenario.

Regarding YouTube traffic, Figure 8-a shows the mean of the initial buffering time for YouTube videos, which is around 3.8 s. It should be noted that the current Best Effort scheme is widely used today by the majority of operators. Figure 8-b shows the distribution of the quality of video chunks, almost 50% of chunks are delivered in the highest quality and only around 5% of chunks are delivered in the medium quality. When UEs are in poor radio conditions, they do not obtain enough resources to support high definition video chunks. The impacts of this low/medium quality level should be evaluated in the light of customer experience (i.e. depending on screen size). Thus, there is an urgent need of Quality of experience (QoE) metrics as a MOS to video traffic. However, this is a topic currently under study by many researchers.

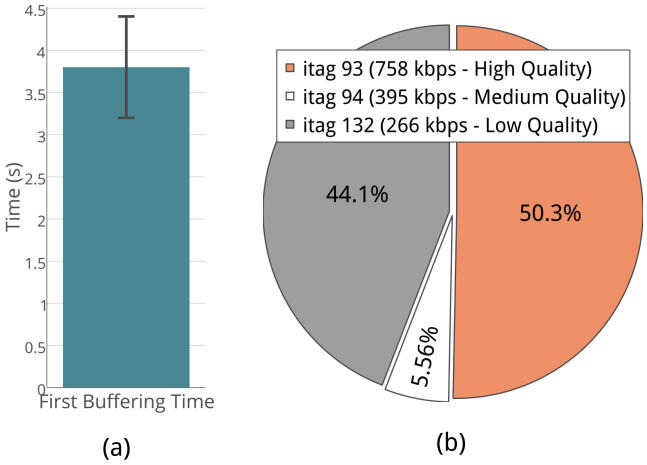

Figure 8: Performance indicators as (a) first buffering time *(90% confidence interval)*, (b) chunks quality

Simulations have highlighted the efficiency of our framework, which allows having realistic behavior of most used traffic in current mobile networks. You can also find extensive simulation results of our proposed framework in [6].

5. EXTENSION AND FUTURE WORK

This framework opens many possibilities to evolutions and can be extended to implements/evaluate following:

- Mobile Edge Computing (MEC) use case to video delivery
- Video-Aware scheduler design
- Video Mean Opinion Score (MOS) models
- Mobile video optimizers mechanisms

We plan to extend our current framework in order to implement and evaluate the RAN-aware Content Optimization to video delivery, which is a MEC use case proposed by the European Telecommunications Standards Institute (ETSI). We also work in a model for video MOS (v-MOS), which is a hot topic, where the proposed framework to generate HAS video will be a key tool to evaluate our model in realistic scenarios.

6. CONCLUSIONS

In this paper we have presented the design and the implementation of a HAS traffic generator for ns-3 based on YouTube traffic characterisation. Simulation results have confirmed the observed YouTube traffic behaviour. We analysed YouTube performance over an LTE network and we showed some performance indicators.

7. REFERENCES

[1] Sandvine. " Global Internet phenomena Report". Technical report, 2014.

[2] Cisco Systems Inc. White paper: "Cisco Visual Networking Index: Forecast and Methodology, 2014 - 2019", May 2015.

[3] D. Ammar, T. Begin, and I. Guerin-Lassous. "a new tool for generating realistic internet traffic in ns-3". In *EAI SIMUTools*, 2011.

[4] P. Casas, P. Fiadino, A. Sackl, and A. D'Alconzo. "YouTube in the move: Understanding the performance of Youtube in cellular networks". *In IEEE IFIP 2014*.

[5] Y. Cheng, E. K. Çetinkaya, and J. P. Sterbenz. "transactional traffic generator implementation in ns-3". In *In IEEE ICST 2013*.

[6] W. Diego, H. Isabelle, and X. Lagrange. "Cross-layer Design and Performance Evaluation for IP-Centric QoS Model in LTE-EPC Networks". In *IFIP WMNC*, 2015.

[7] Ericsson. White paper: "Mobility Report", February 2016.

[8] M. Haddad, E. Altman, R. El-Azouzi, T. Jiménez, S. E. Elayoubi, S. B. Jamaa, A. Legout, and A. Rao. "A survey on YouTube streaming service". In *In ICST 2011*.

[9] Y. Liu, F. Li, L. Guo, B. Shen, and S. Chen. "A comparative study of android and iOS for accessing internet streaming services". In *Passive and Active Measurement*. Springer Berlin Heidelberg, 2013.

[10] H. Nam, B. H. Kim, D. Calin, and H. G. Schulzrinne. "Mobile video is inefficient: A traffic analysis". *Columbia University, 2013*.

[11] H. Nam, K. H. Kim, B. H. Kim, D. Calin, and H. Schulzrinne. "Towards A Dynamic QoS-aware Over-The-Top Video Streaming in LTE". *In IEEE WoWMoM 2014*.

[12] J. J. Ramos-Muñoz, J. Prados-Garzon, P. Ameigeiras, J. Navarro-Ortiz, and J. M. López-Soler. "Characteristics of mobile youtube traffic". *Wireless Communications, IEEE*, 2014.

[13] F. Sonnati. "Implementing a dual-threshold buffering strategy in Flash Media Server". http://adobe.ly/RqSJnQ, 2006. [Online; accessed 22-July-2016].

[14] F. Wamser, P. Casas, M. Seufert, C. Moldovan, P. Tran-Gia, and T. Hossfeld. Modeling the youtube stack: From packets to quality of experience. *Computer Networks*, 2016.

[15] E. Weingärtner, R. Glebke, M. Lang, and K. Wehrle. "building a modular bittorrent model for ns-3". In *EAI SIMUTools*, 2012.

[16] R. Yang and H. J. Son. "Youtube's Live TV Streaming in Mobile Devices - HLS and Adaptive". Technical report, Netmanias Tech-Blog, 2013-10-30. [Online; accessed 22-July-2016].