Network Protocols and Algorithms
ISSN 1943-3581
2017, Vol.9, No.3-4

FlexiTop: a flexible and scalable network monitoring system for Over-The-Top services

Daniel Perdices[1,2], Jorge E. López de Vergara[1,2], Paula Roquero[1,2], Carlos Vega[1,2], Javier Aracil[1,2]

[1]High Performance Computing and Networking Research Group,
Universidad Autónoma de Madrid, Spain
Tel: +34 91 497 22 46 E-mail: {jorge.lopez_vergara, javier.aracil}@uam.es

[2]Naudit High Performance Computing and Networking, S.L., Spain
Tel: +34 91 269 35 40
E-mail: {daniel.perdices, paula.roquero, carlos.vega}@naudit.es

Received: Oct. 10, 2017 Accepted: Dec. 26, 2017 Published: December 31, 2017
DOI:10.5296/npa.v9i3-4.12488 URL: https://doi.org/10.5296/npa.v9i3-4.12488

Abstract

Nowadays, the demand of Over-The-Top (OTT) services such as multimedia streaming, web services or social networking is rapidly increasing. Consequently, there is a wide interest in studying the quality of these services so that Internet Service Providers (ISP) can deliver the best experience to their clients.

For this purpose, we present FlexiTop, a flexible and scalable system to actively monitor these OTT services, which allows an operator to obtain metrics with a limited resource usage. Due to the continuous evolution of OTT services, this system was designed with different approaches that can be extrapolated to future situations.

By looking at the results, the proposal meets all the expectations and requirements and therefore it proves its success. The proposed design was implemented and validated with different alternatives whenever it was possible, both in wired and wireless networks. Moreover, long-time testing was performed to both ensure its stability and analyze the obtained data.

Keywords: QoS, QoE, active monitoring, OTT services, streaming, web services.

1 Introduction

Over-The-Top (OTT) services are those services carried over the network without any network operator involved in planning, selling or provisioning them [1]. The necessities and behaviors of these services are very heterogeneous. Whereas video streaming services require a given and stable bandwidth, other services such as web services prioritize the response time over other parameters. Moreover, new metrics are interesting from the point of view of Quality of Experience (QoE). For instance, the time of a high-level operation or a transaction. Note that this may include several TCP connections or HTTP requests. Thus, the aforementioned necessities are a motivation to study these parameters on different services.

Passive network analysis arises as a convenient solution to this problem. Nevertheless, OTT services usually work with secure protocols like HTTPS. As the payload is not available, some valuable metrics cannot be obtained. For example, the requested URLs or the HTTP response codes.

Sometimes, dealing with encryption is not a viable option. In fact, current secure protocols use ephemeral keys for each session, so it is not possible to decrypt any communications, even if you have the server private key. Consequently, active network monitoring emerges as a solution to this issue, because the system can be completely monitored from the client.

Furthermore, the constant evolution of these services in terms of performance and functionality makes necessary to evolve constantly the monitoring systems. Actually, systems that are focused on a single protocol might not be useful in the near future. Additionally, both ISP and clients have a wide interest in OTT services.

With this aim, we propose FlexiTop, an active network monitoring system for these services. It has been designed to be deployed on low-end computers and with a possible massive deployment in mind, so that measurements of thousands of devices can be aggregated and analyzed. A preliminary version of this work was already presented in [2]. Now, in this paper, we provide a fully detailed explanation of the algorithms behind the tests, with a special emphasis in the measurements that were performed in a LTE network and the necessary server that aggregates all these data.

The analysis of services such as YouTube [3] or another web services has been performed previously in very controlled testbeds, paying special attention to Quality of Service (QoS) metrics. However, this proposal is completely oriented to Key Performance Indicators (KPI) and network metrics, prioritizing the ones that users think are useful and representative. In addition, some scalability and resource consumption study is added to cope with situations where there are constraints related to these aspects.

The system has been tested in different devices and networks: from powerful workstations and home PCs to embedded systems and virtualized environments, both in a large bandwidth network at Universidad Autónoma de Madrid (UAM), a land line home access, and an LTE cellular network. In every case, the results were successful and the metrics were validated with different alternatives whenever it was feasible. This made possible to compare and analyze the different issues that might be causing a deficient user experience.

To explain how FlexiTop works, this paper is structured as follows: First, different alternatives to measure quality in Internet-based services are shown in section 2, comparing them with our proposal. Second, section 3 contains a study of the technologies commonly used nowadays in OTT services. Next, section 4 enumerates the metrics that are interesting for users and service providers. After that, section 5 describes the architecture of the monitoring system. In section 6, different devices and configuration are documented to prove the portability of the system. Finally, the obtained results are discussed and some conclusions are given in section 7.

2 Related work

Nowadays, there exists a huge variety of standardized metrics for the network and transport level. These ones provides relevant information about the flows that the service use as a transport channel. This can give much information to analyze the possible cause of the observed QoE. Many of these metrics are completely described by the Internet Protocol Performance Metrics (IPPM) working group of the IETF. Concretely, the delay measurements in one way [4] and round-trip [5] are really important for this work, as these methodologies are grounded standards and good examples of how to measure the latency.

Another possible approach to the problem of monitoring OTT services is passive network monitoring. It is based on observing network patterns to characterize the traffic associated to this service. Although this is possible, it requires expensive calculations and powerful mathematical models that adapt to each service. Some characterizations for Facebook [6], YouTube [7] or other more complex VoIP (Voice over IP) services [8] can be found. In contrast to OTT services, there are also some works related to the quality of triple-play services [9].

On the other hand, applications for active network monitoring for OTT services already exists. For example, Ixchariot [10], a commercial product focused on VoIP, web browsing and video streaming, provides metrics such as jitter or packet loss, whereas we go deeper into more high-level KPIs that are more natural to the user. SamKnows [11] provides a similar but limited set of high-level KPIs, but it is mainly developed to work on modified routers, whereas we do not want to rely on custom hardware setups.

Furthermore, it has been observed that there is a high correlation between QoS and QoE [12]. This relation has been used previously in the literature [13] to be able to estimate in real time the QoE through QoS metrics. Thus, our work is based on similar principles.

Besides, some models exist for the concrete case of HTTP video streaming [14, 15]. In fact, YouTube [3] and other IPTV services [16] have already been analyzed. However, from the best of our knowledge, such study has never been done with a general approach. Consequently, it is important to measure the right QoS variables in order to ensure a good functioning of the service [17]. Thus, we have chosen KPIs that are close to the perception of the user.

For web services, there are previous studies focused on several standard QoS metrics, especially they are significant to ensure that a service meets the Service Level Agreement [18], but this is not the case of OTT services, where the metrics are completely different.

As we want to obtain more correlated metrics, application level KPIs are object of study. Some of these KPIs and other parameters such as the buffer size have been studied in other previous works [19]. Although the impact of these application parameters are not negligible, they are out of the scope of this article and a parameter-agnostic approach is provided, where they are fully configurable.

3 Technologies used in OTT

To completely understand the problem to solve, it is necessary to study the different technologies and standards that are behind the services. In this particular case, several technologies are used for OTT services. The most used ones are commented here.

On the one hand, for video streaming services, most popular technologies are:

1. Dynamic Adaptive Streaming over HTTP (DASH): binary adaptive bitrate transmission technique that works with conventional HTTP servers [20].
2. HTTP Live Streaming (HLS): similar to the aforementioned DASH, it stores in conventional HTTP servers different codifications for each video fragment [21].

Most of these technologies share an important aspect: they work over HTTP/HTTPS. This behavior allows that the monitoring of video streaming services can be based on HTTP monitoring. However, some difficulties are found when the host in the URLs is translated to an IP address. Commonly, these services work with Content Delivery Networks (CDN) that are usually correlated with the location of the client and the configured Domain Name System (DNS) servers.

Moreover, other factors and parameters that are not directly correlated to the network QoS, such as buffer size, codecs or bandwidth thresholds for each quality of the video, are important when studying the performance and QoE of the application. Nevertheless, the intention of this paper is to provide an agnostic approach in terms of configurable parameters in the user application. Other metrics such as packet loss have proved its relevant impact on the throughput when using TCP [22] and therefore we choose to observe directly the throughput, as a combination of all artifacts that affect the transmission of data. In fact, OTT service providers such as Netflix or Youtube implement throughput measurement for their users, stating that this is the main KPI they measure to provide their service.

On the other hand, for web services, some technologies are dominant:

1. Representational State Transfer (REST): communication architecture based on HTTP resources [23].
2. Simple Object Access Protocol (SOAP): standard message protocol to exchange data and call remote procedures [24].

Currently, both applications on top of web pages and applications from mobile devices use communication protocols based on REST for the communication between client and servers. Therefore, a natural interest exists in knowing the response time of these Application Programming Interfaces (APIs), usually called RESTful APIs.

4 Relevant metrics

As previously motivated, metrics might be different depending the service. Thus, their characterization and uniformity is needed to ensure that they are both valid and comparable.

4.1 DNS

First, DNS queries are a key factor of modern services. Therefore, the interest in knowing both how domain names are translated and the response time of each query is growing. The principal metric of DNS level is the response time. In addition, the list of response IP addresses is stored to allow a later deep analysis if it is needed.

Figure 1 shows an example of time series of the median of response time of DNS service for the 20 most visited domains in a country. This list of domains was retrieved from a public list. In this case, Alexa [25] was used for Spain. These metrics are easily validated using standard tools such as `dig` [26]. Daily trends are observable but their variation is small and it might not be noticeable for the clients.

4.2 HTTP

As many services work over HTTP(S), a specific monitoring system for this protocol is required. This system must be able to cope with different requirements to work with both only one connection and multiple concurrent connections.

The approach is the following: measuring the request spread, this is the time between when the first and last byte of the request is sent and estimating the response spread, this is, the time between the first and last byte of the response. Moreover, some metrics from the network level (*e.g.* fragmentation), transport level (*e.g.* retransmissions) and application level (*e.g.* HTTP codes) can be obtained. This approach is similar to the one proposed in TRANSUM [27], a dissector for Wireshark that calculates these metrics for HTTP and other protocols such as FTP or SMB2.

These metrics extends easily to web browsing. In this case, these metrics are obtained for each document and for the whole dependencies as a block. Dependencies are downloaded as if it was a web browser: the base document is downloaded and parsed, and after that, the

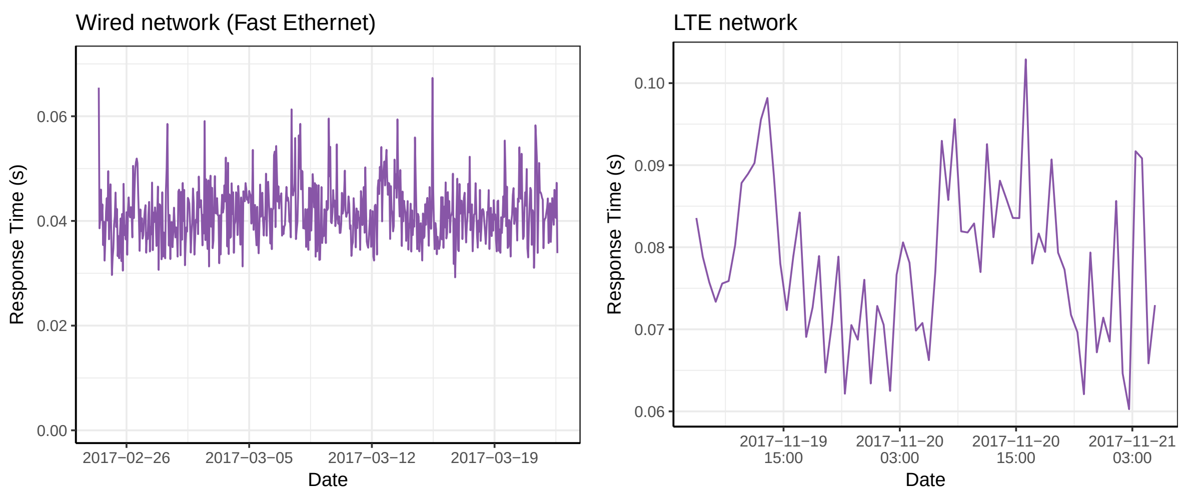

Figure 1: Time series of DNS response times. Left time series shows a couple of weeks of measurements taken on an Ethernet network at UAM. Right time series shows some days of measurements from an LTE network where variations of the network performance are observable. Fewer measurements were taken at the LTE network due to data consumption restrictions.

dependencies are downloaded concurrently, in parallel connections, mimicking the behavior of a web browser.

4.3 Multimedia streaming

For video streaming services, throughput is a key performance indicator. It is not only a limiting factor for the quality; it is also a source of issues because of its variance. Therefore, we pay attention to the throughput and its variance.

In addition to the aforementioned HTTP metrics, throughput, download time and processing time (manifest downloading, parsing, solving domain names, and decompression) are also measured.

4.4 Web services

A wide variety of services works over HTTP. Due to their heterogeneity, the services must be studied case-by-case.

In general, reasonable parameters and KPIs can be access time, complete transaction time or throughput. As a complete transaction might accumulate also parsing and processing time, in this case, we chose to include parsing/processing times in the transaction. Bear in mind that a transaction may use more than one message or connection.

4.5 Other services

There are services that still work over raw TCP sockets. For these services, aforementioned network and transport metrics are also measured but it is more valuable to look at KPIs such as connection establishment time (it includes login when required) or transaction time. These metrics are close to the real delays that the users perceive.

Best example of these type of services is e-mail. Consequently, we focused on common and well-known protocols: SMTP [28], IMAP [29] and POP3 [30].

5 Architecture

The architecture of the system was designed with several requirements in mind. They are:

- **Flexibility**: the system must adapt to the change and evolution of the services.
- **Scalability**: The system must be deployable in different locations without many configuration issues. This allows a massive deployment where data can be aggregated in a central server.
- **Low resource consumption**: the target devices of this system are low-end computers or single-board computers, so it must have the less possible impact and a limited resource consumption.

Figure 2 shows the architecture with three deployment examples of the monitoring agent and a main server. In the following subsections, both server and the monitoring agents will be described.

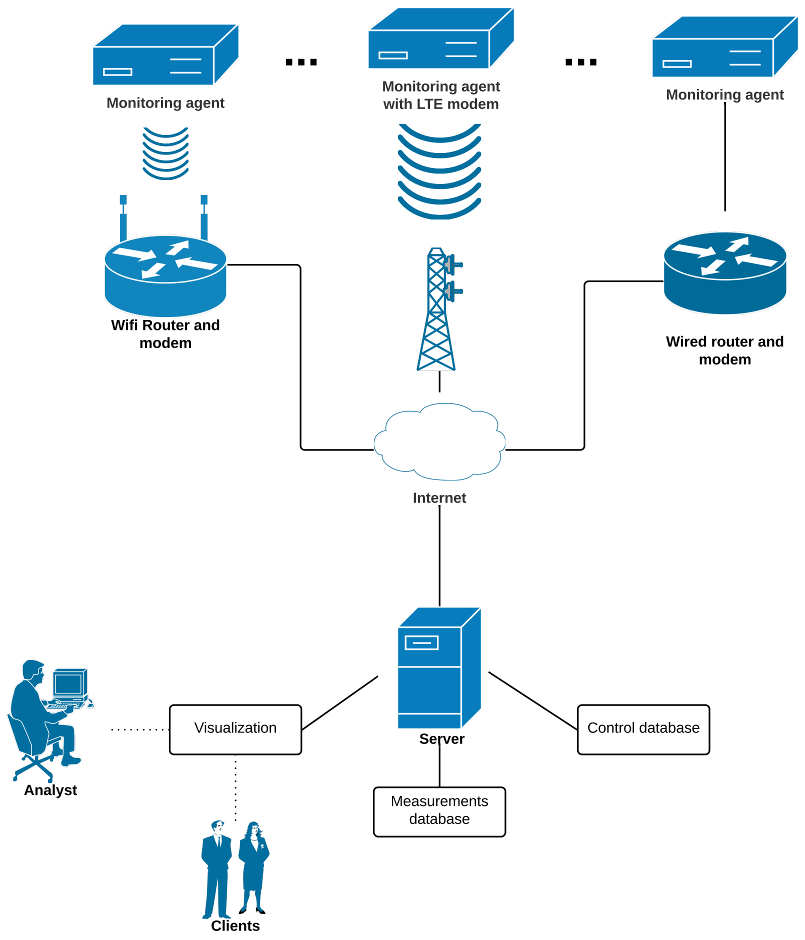

Figure 2: Architecture of the FlexiTop system.

5.1 *Architecture of the monitoring agent*

The architecture of the monitoring software agent is shown in Figure 3. It is composed of several modules:

1. **DNS**: this module is used in the whole system to test the performance of DNS servers when a domain name is resolved.

2. **HTTP**: This module consists on a configurable HTTP download engine. It supports multi-thread or single-thread downloads and also some configurations on buffers.

3. **TCP**: this module is analogous to the HTTP module, but for services that are directly based on TCP.

4. **Database Access Object (DAO)**: This module is the abstraction of collection of results and logs. It allows sending transparently the results to another computer, store them locally or keep them in a database. To send the results to the main server, a REST API is used. This API must have the common security measures to ensure that only authorized agents are able to send measurements to the main server. This is easily achieved by using

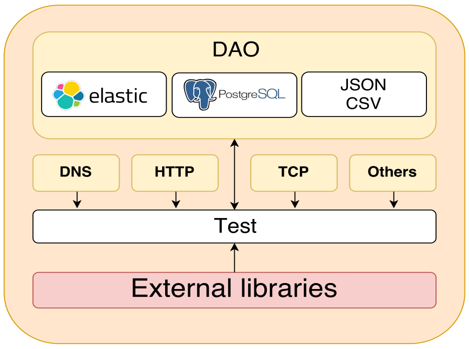

Figure 3: Architecture of the measurement agent.

Transport Layer Security (TLS) and agent authentication through certificates signed by our own Certification Authority (CA). Besides, the results are stored locally in CSV and JSON.

5. **Other libraries and utilities**: some other custom libraries are provided to help with common tasks such as parsing requests or OAuth2 [31] authentication.

With additional modules that benchmark the emerging technologies, this architecture guarantees that the requirements are met. For each test, the generic schema is the following:

1. Obtain URLs.
2. Use DNS module to check if there is a DNS issue with this domain names.
3. Use HTTP, TCP or other modules to test and obtain the desired KPIs + metrics.
4. Use DAO to store the results.

Additionally, the system also allows using external dissector such as `tshark`, so that a passive analysis of these connections is made. Note that this requires the system to record data also about identifiers (*e.g.* addresses and ports quintuple) of flows and connections.

5.2 Architecture of the central server

A central server was configured to aggregate all the results. If necessary, monitoring agents can also work without a central server. This server has the following software:

1. **Nginx** [32]: as reverse HTTP proxy to control the access to the resources below.
2. **ElasticSearch** [33]: as database for results and logs.
3. **SQLite** [34]: as database for configurations and extra data.
4. **HTTP Server (Python customised)** [35]: to deliver configuration files and updates.
5. **Grafana** [36]: as a real-time visualization environment.

This allows analysts to change remotely the configuration of each monitoring agent and to push updates to the devices.

Figure 4 shows the visualization environment where analysts can see an aggregated summary of the performance of each service and also clients can get the same summary but filtering to display only data obtained from their own agent and a comparison with the aggregated results from the deployed agents.

5.3 Special considerations for LTE deployments

In a broadband mobile network, the system has to cope with several restrictions. First, the data consumption is normally limited. Therefore, each engine of the agent has to keep information of the consumed bytes and use this information to configure the granularity of the samples. Moreover, as data consumption is not uniform across services, the granularity can be configured so that streaming services that usually need a large amount of data are tested less than web services, which only require sending and receiving an almost negligible amount of data.

In addition, a second issue is the intermittent connectivity. Connectivity problems can arise in a wireless service such as LTE, so that we had to develop an active probing subsystem that detects disconnections and performs maintenance operation in the mobile modem in order to recover the connectivity.

This intermittent connectivity also causes that some obtained registers record a high latency. Thus, outliers may appear that can affect the mean of the measures. Median was chosen in most cases as aggregation function inside temporal bins to make the registers stable to these random disconnections. These registers of outliers were kept to record also the time between disconnections and how they can affect a service.

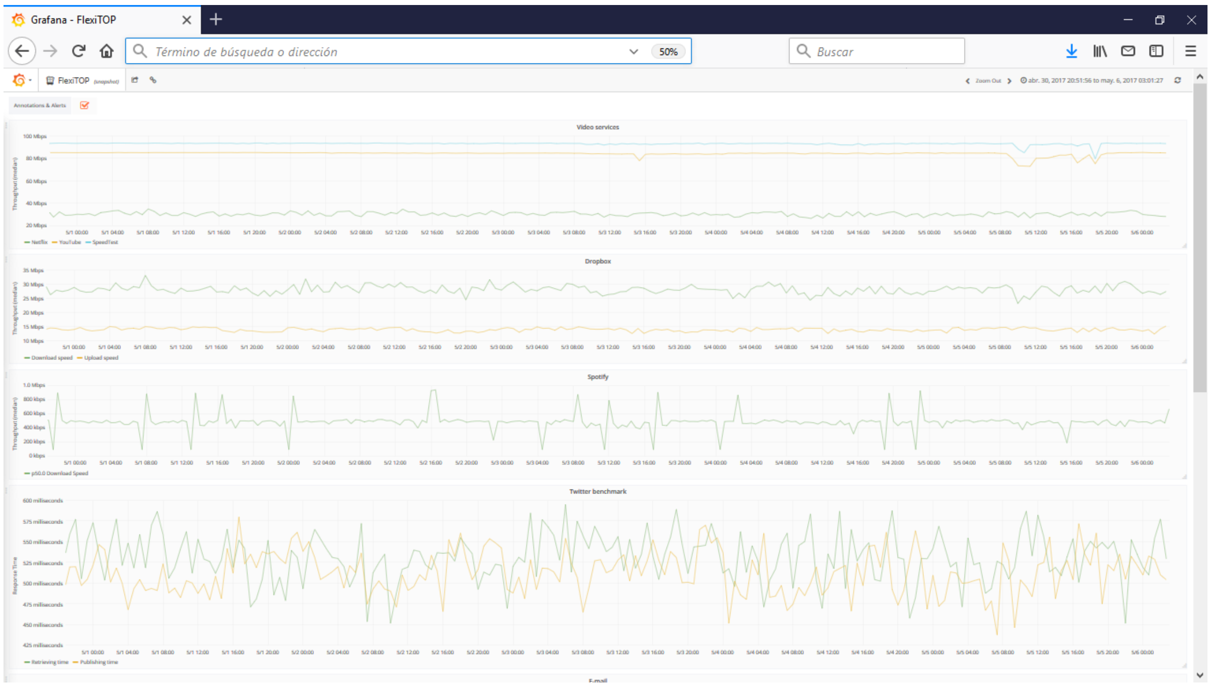

Figure 4: Screenshot of the web visualization environment.

6 Testbeds

Because the portability, versatility and high-level, a prototype of the software of the monitoring agent was developed in Python. It was tested on the following devices:

1. Orange Pi Plus 2: H3 Quad-core Cortex-A7, 1 GiByte DDR3, Gigabit Ethernet.
2. Test Computer (host): Intel Core i7 860, 8 GiByte DDR3, Gigabit Ethernet.
3. Virtual Machine (guest): dual core, 1GB DDR3, 2 GiBbyte of HDD, Gigabit Ethernet.
4. LTE Test Computer (host): Intel Core2 Duo E6550, 2 GiByte DDR2, LTE USB dongle.
5. LTE Virtual Machine (guest): single core, 1 GiByte DDR2, 10 GiByte of HDD (embedded visualization environment), using the LTE USB dongle provided by the host.

The system has been being running continuously several months. For the aforementioned devices, in the first case the device was dedicated to the system. In the other ones, the system was also dedicated to other tasks so the resource consumption was limited. Execution on both virtual environment and on host OS obtained similar results.

The virtual environment is interesting because it is a simple way of limiting the resource usage by limiting the access to the complete hardware of the devices.

The central server is installed on the first described computer, which has a public static IP address to be accessed from anywhere in Internet.

7 Tests and results

Once the architecture was implemented, several tests were done in order to validate the design. These are:

- **Multimedia streaming**: Netflix, YouTube, Yomvi, Spotify.
- **Web services**: Twitter, Facebook, Dropbox, Speedtest, Google search engine, web browsing.
- **Others**: e-mail.

In the implementation of each test, a variety of techniques was used to simulate the usage of the service. Each part of this section explains several particularities of each type of service. Some results are also shown to prove the feasibility of our solution.

7.1 Multimedia streaming

Measuring the download speed is the main goal of the system for these services, transported over HTTP. We do not ignore the effect of other metrics such as delay or jitter, but normally their effects are hidden by the reception buffer that is usually implemented in this type of services, with several seconds of video cached in the client before it is replayed. Thus, we focus on download speed. For this purpose several multimedia content URLs are extracted and downloaded simultaneously. Nevertheless, obtaining the URLs may differ depending on the measured service.

In some cases, such as YouTube, there exists a REST API [37] and some libraries to parse DASH manifest as well. Similarly, Spotify provides a REST API [38] that allows users to get 30 seconds samples. Therefore, sophisticated techniques are not needed to deal with them.

On the other hand, some other techniques were used in order to analyze the protocol and be able to obtain valid URLs. In this case, a useful method is to follow a Man-In-The-Middle (MITM) approach. This allows observing how the service works and how these URLs can be obtained. The used tool was `mitmproxy` [39], which is freely available under MIT license. With the same goal, reverse engineering techniques can also be applied to on-line multimedia players to infer the API they used.

An example of this procedure was applied to Yomvi, a Spanish video streaming platform. In this case, just by analyzing the HTTPS connections, the URLs were easily obtainable without the necessity of using a web browser.

However, this process can be hard and very intensive for the monitoring agent. In this case, the only alternative is to execute a headless web browser during a short period time. Although this procedure seems viable and there exist tools that work inside the web browser for this purpose, this is the worst case for the low resource consumption.

This is the case of Netflix streaming platform. Although it works with DASH, it uses encryption for the URLs of the video fragments. In this case, the above procedure is possible. Anyway, after the release of fast.com [40] , Netflix provides a speed test based on their video fragments. Performing reverse engineering on this web page is more viable and the current test is working by mimicking the usage of this tool.

Figure 5 shows the results of the throughput in Mbit/s for these video streaming platforms. In the left side, a long time test was performed to prove the stability of the system. In the right side, a day of monitoring using an LTE connection is shown to observe the variability that may affect mobile devices and the deficiencies on QoE in case the visualization codec is strongly dependent of the throughput. In both cases, this analysis provides valuable information on how QoS distributes over the time.

For the case of audio streaming, download speed is also the most valuable KPI. Nevertheless, the demand of throughput is not high enough to be really compared to video streaming one. Figure 6 shows the medians for the download speed of a sample track obtained through the Spotify API. For the wired network, we notice that the variability does not have a huge impact because of some pre-fetching mechanisms of the application and because the throughput is greater than a reasonable threshold. It is also remarkable the notorious change of QoS around February 2017, followed again by a decrease in the observed parameter. Given that these data are taken in the wild, these changes show that our tool can be useful to detect variations in the measured KPIs.

Whenever it was possible, these metrics were validated with the players or other tools available to measure some of these metrics.

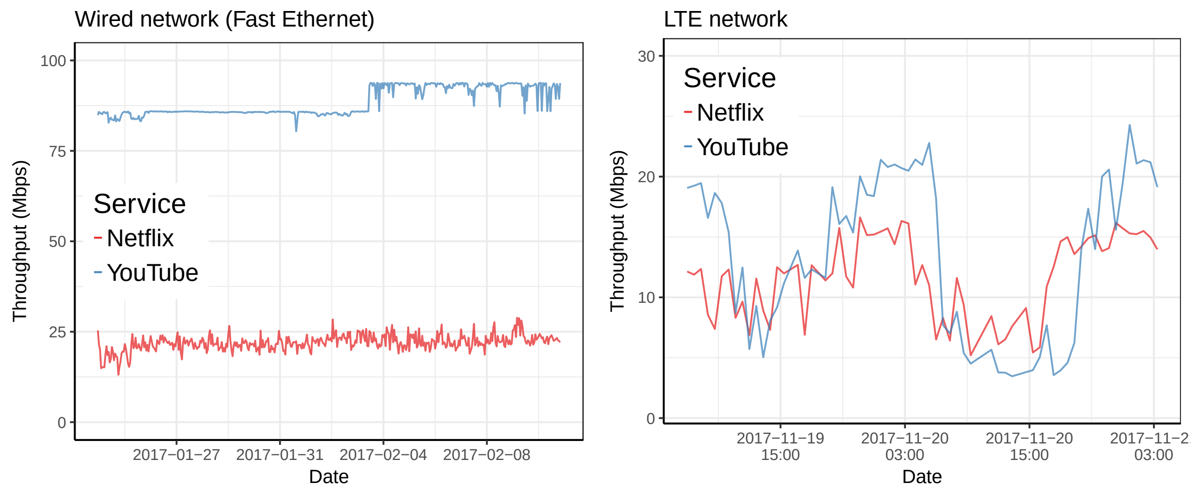

Figure 5: Time series of achieved throughput for Netflix and YouTube. Left time series shows a couple of weeks of measurements taken on an Ethernet network at UAM. Right time series shows some days of measurements from an LTE network where daily variations of the network performance are observable. Fewer measurements were taken at the LTE network due to data consumption restrictions.

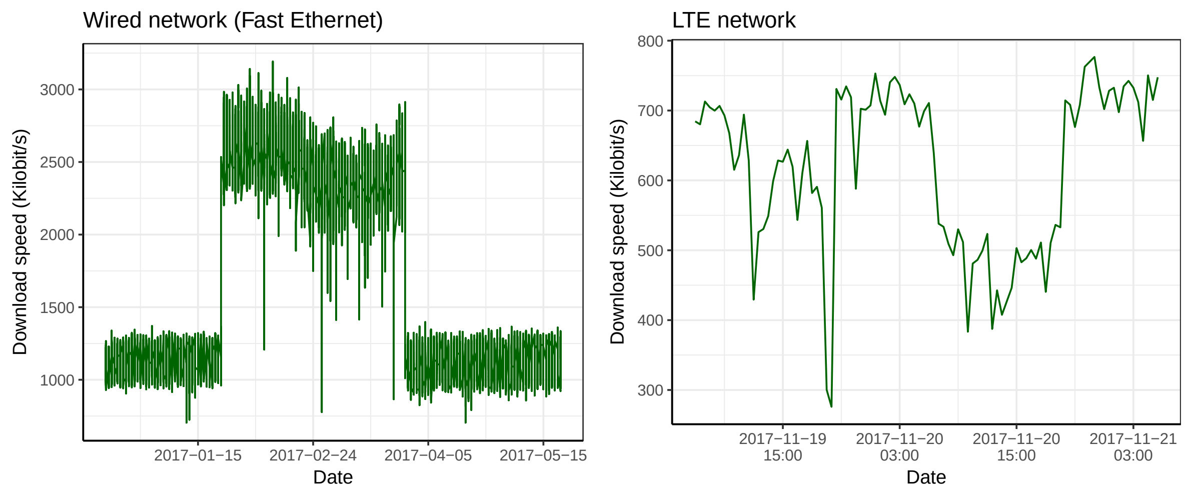

Figure 6: Time series of achieved throughput for Spotify. Left time series shows a couple of months of measurements taken on an Ethernet network at UAM. Right time series shows some days of measurements from an LTE network where daily variations of the network performance are observable. Fewer measurements were taken at the LTE network due to data consumption restrictions.

7.2 Web services

For web services, the process is analogous to the one mentioned above. For services that provide a public API, the monitoring test is built on top of it, and it mainly tries to measure response time and throughput. This is the case of Twitter [41] and Dropbox [42].

For other services, an analysis of the requests and responses is required. Again, we used MITM technique to be able to monitor HTTPS connections and to study the logic of the protocol and the API. In this case, a controversial case is WhatsApp. Although API is almost described by third-party developers, terms and conditions of usage forbid the use of any third-party software to access WhatsApp. Furthermore, each device requires a phone number (this is against the scalability of the system). In addition, another issue in this service is that any device that violates the terms of usage eventually gets banned. Because of this, the unique possible approach is to integrate the client into the system.

As an example of measuring a concrete web service, Figure 7 shows the medians of Twitter service response times when publishing a tweet and retrieving a timeline. Without taking into account internal functional aspects of the application, this metric should be much closer to the experience of the users than analyzing the connection establishment time. In both time series, a daily trend can be observed. During night hours, the service obtains the lowest response times, while the response time increases in peak hours. Given that Twitter is a global service, this is probably caused by the amount of traffic traversing the operator's network.

For services that are completely based on file handling, the most relevant metric is the throughput. In this case, we provide the example of Dropbox. Bear in mind that previous examples of speed were obtained with multiple connections. In this case, as the official application of Dropbox was using only a single connection, we show the results with a single connection approach. Figure 8 shows the medians of Dropbox throughput. In both cases, a daily trend seems to happen but this trend is not as clear as before. This results in a noisier time series and therefore in a higher variance of the performance along the day. This higher variance is likely to affect QoS.

A completely different case is the download of concrete web pages. In this case, the time to download of some pages was measured. For this task, the base document is firstly downloaded. It is parsed looking for dependencies. Then, all the dependencies are downloaded in parallel. All the download times are calculated. As motivated before, we measure the request spread as the difference between the timestamps of first byte and last byte of HTTP request. The response spread is analogously calculated. Moreover, some extra metrics such as download speed and URLs can be extracted.

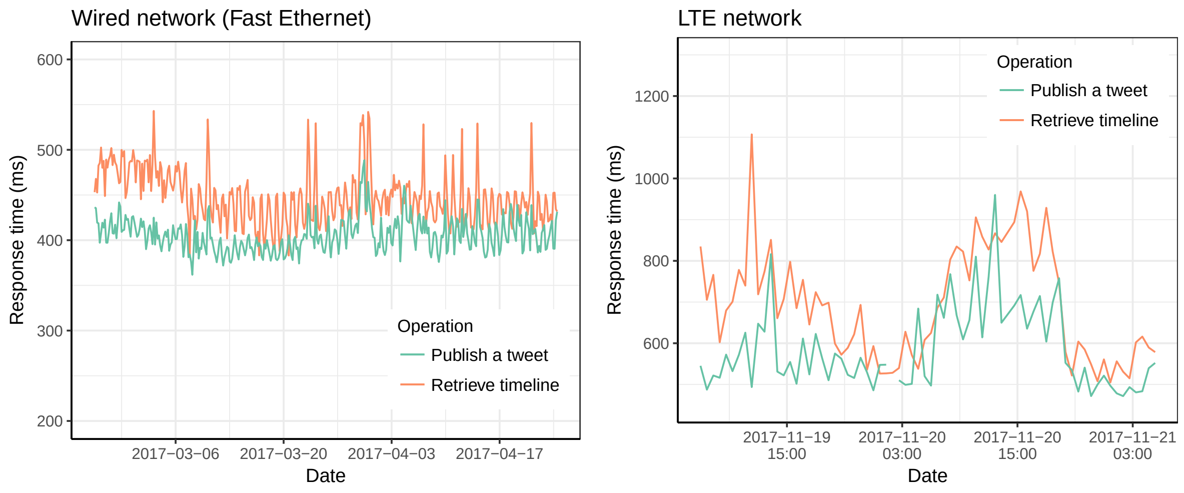

Figure 7: Time series of the time to publish a tweet and the time to retrieve a timeline at Twitter. Left time series shows a couple of weeks of measurements taken on an Ethernet network at UAM. Right time series shows some days of measurements from an LTE network where variations of the network performance are observable. Fewer measurements were taken at the LTE network due to data consumption restrictions.

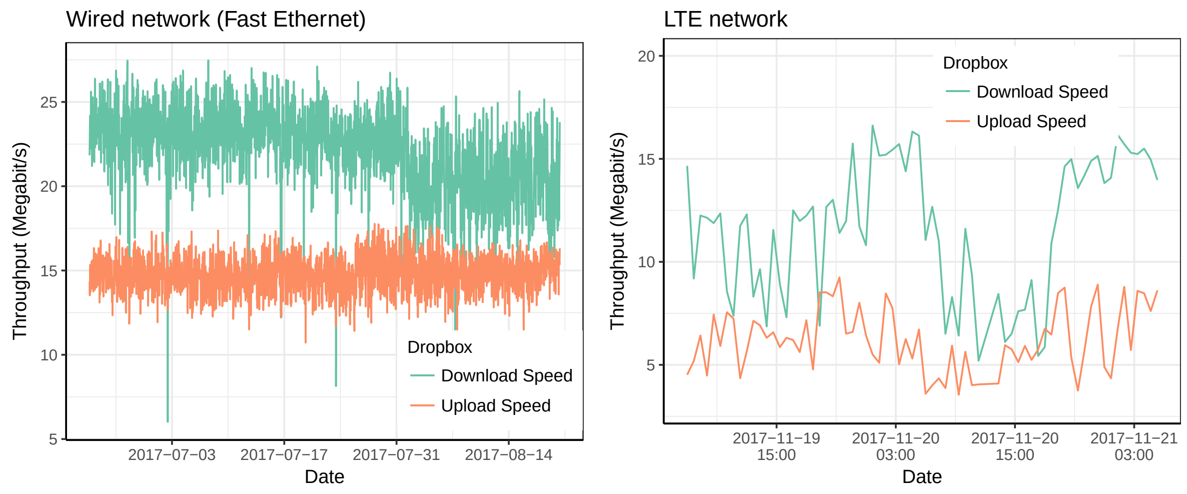

Figure 8: Time series of the upload and download speed of Dropbox using one connection. Left time series shows a couple of weeks of measurements taken on an Ethernet network at UAM. Right time series shows some days of measurements from an LTE network where variations of the network performance are observable. Fewer measurements were taken at the LTE network due to data consumption restrictions.

Up to now, JavaScript documents are downloaded but not interpreted. This avoids security issues and reduces the resource consumption. Nevertheless, this may compromise the correlation between the metrics and the QoE because of the current trend of dynamic web pages. For mostly static webs, the times were validated successfully on several web browsers.

For future versions, it is planned to add the ability to execute JavaScript code with the support of headless web browsing technologies [43]. Other possible is to integrate the monitoring agent into a web browser so that this task is easily solved.

As an example of web browsing measurement, we studied the load time of Facebook website. In this case, as we want to avoid dynamic content, we download a profile page with a high proportion of static content. Figure 9 shows the medians of load times of this profile page. Daily trends are observable, but, most interesting it is that there are some peaks, which denote that there were probably some issues.

7.3 Other services

For this case, the e-mail services using standard protocols like IMAP, POP3 and SMTP was chosen as example of OTT service that does not work over HTTP/HTTPS. As in previous cases, some KPIs and metrics are studied, such as Connection Establishment Time (CET) for IMAP, POP3 and SMTP, time to retrieve the mail list, and send time. This just shows that the developed tools can also deal with services that work over plain TCP connections.

Passive monitoring could be a strong alternative in this context. However, once again due to the encryption issue, there is not a cost-effective way of identifying the commands. Therefore, active network monitoring or decryption are necessary to obtain KPIs such as the aforementioned ones. Our proposal allows the user to obtain these KPIs and to identify the flows.

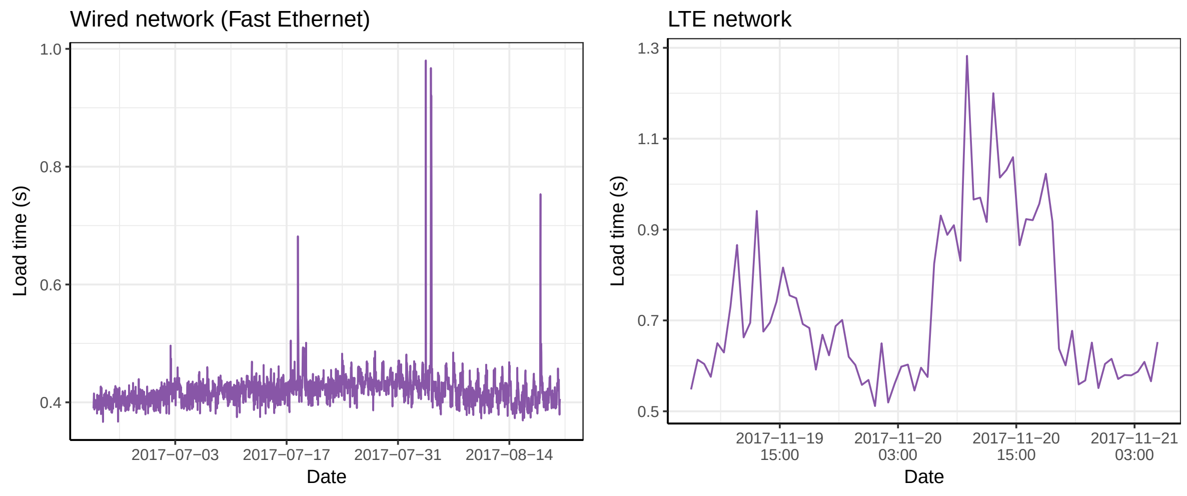

Figure 9: Time series of the time to load a Facebook profile page. Left time series shows a couple of weeks of measurements taken on an Ethernet network at UAM. Right time series shows some days of measurements from an LTE network where variations of the network performance are observable. Fewer measurements were taken at the LTE network due to data consumption restrictions.

Figure 10 shows the time series for Connection Establishment Time (CET), retrieving time and deleting time for IMAP and POP3 provided by Gmail service. In both cases, the CET represents a huge amount of time. As natural, always retrieving time is higher than deleting time as retrieving time implies some data transfer. The same figure includes below analogous metrics for SMTP. In this case, sending time is clearly greater than CET. All these metrics were performed against Gmail platform through TLS, using a random 2 Kilobyte email payload. The length of the payload is configurable and it can be increased if bandwidth tests are demanded.

8 Conclusion

Active network monitoring has proved to be an alternative to the passive analysis of OTT services. It is both a simple solution to the encryption and flow characterisation issues and allows controlling several application-level parameters such as the number of simultaneous connections or the buffer size.

As a proof of concept, an implementation of the architecture was developed. The obtained results are satisfactory and have been validated with different tools on each. In addition, the metrics and KPIs have shown its utility and relevance because they are closer to the user perception.

The proposed architecture meets all the requirements. Therefore, the development cost of new tests for incoming applications is controlled. Moreover, the proposed procedures of elaborating tests covers different possible scenarios that may occur. The next step of the system will handle services that must run inside a web browser.

In addition, special considerations were taken to be able to run the system in a LTE network so that it is able to recover from disconnections and to limit the data consumption. Thus, results were shown and commented for both wired and LTE networks.

As a future work, the analysis and characterization of the distributions of these metrics and KPIs can lead to a deeper study of the causes behind the observed network performance. For

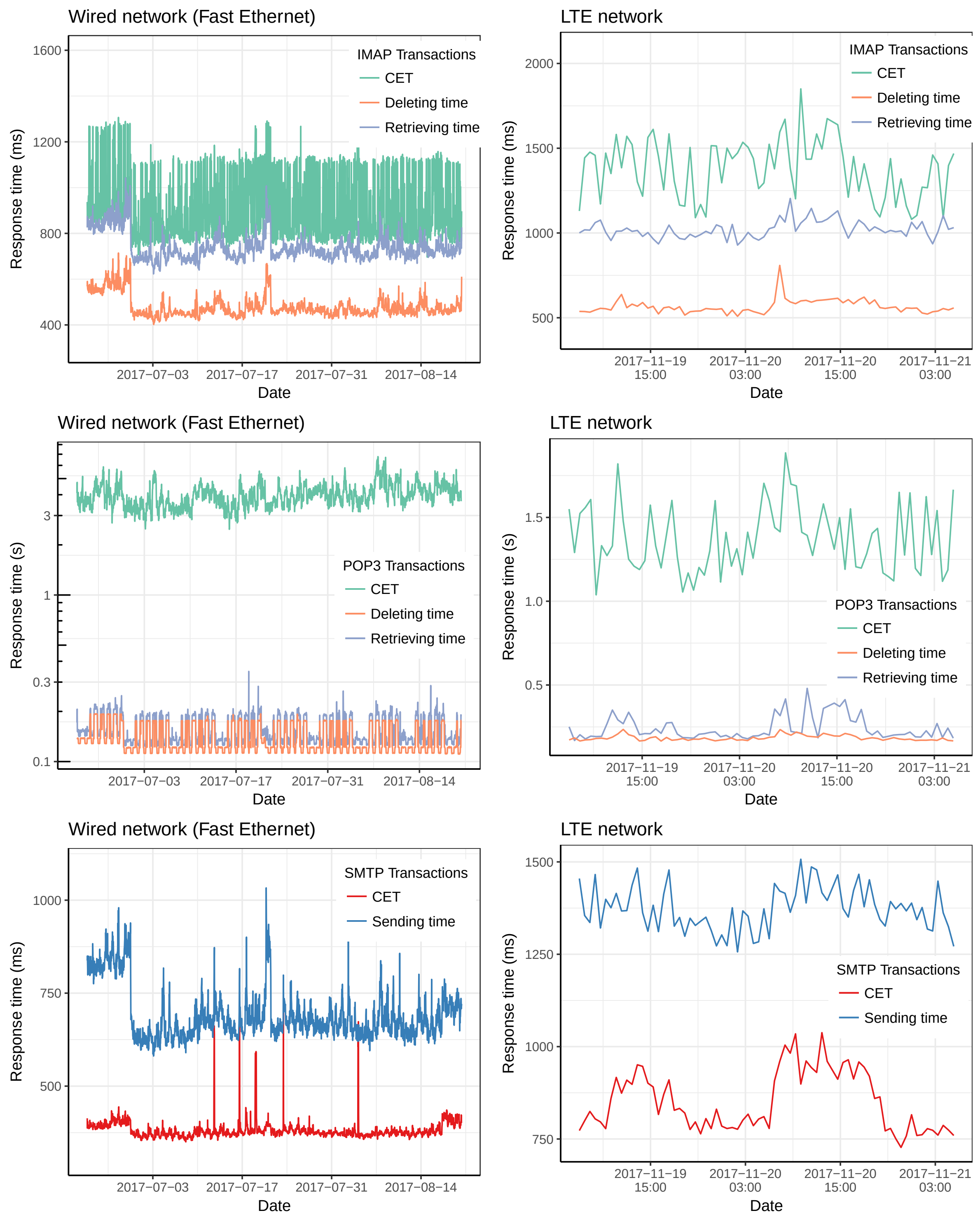

Figure 10: Time series of the KPIs for Gmail over IMAP (top), POP3 (middle) and SMTP (bottom). Left time series shows a couple of weeks of measurements taken on an Ethernet network at UAM. Right time series shows some days of measurements from an LTE network where variations of the network performance are observable. Fewer measurements were taken at the LTE network due to data consumption restrictions.

instance, detecting improvements in the infrastructure or detecting peak hours. Furthermore, some important metrics such as packet loss or jitter were not analyzed because of the desired focus on the proposed high-level KPIs, but this could provide more information about the QoS and observed KPIs.

Acknowledgement

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness and the European Regional Development Fund under the project TRÁFICA (MINECO/FEDER TEC2015-69417-C2-1-R).

References

[1] W. Green, B. Lancaster, and J. Sladek, "Over-the-Top Services," vol. 4, no. 7, 2006. [Online]. Available: http://www.pipelinepub.com/1207/pdf/Article_3.pdf

[2] D. Perdices, J. E. López de Vergara, P. Roquero, C. Vega, and J. Aracil, "FlexiTop: sistema de medidas de calidad de servicio escalable y flexible para servicios OTT," in *XIII Jornadas de Ingenieria Telematica - JITEL2017*, 09 2017, pp. 1–6. [Online]. Available: http://dx.doi.org/10.4995/JITEL2017.2017.6497

[3] G. Dimopoulos, "YouTube Traffic Monitoring and Analysis," Master's thesis, Technical University of Catalonia, 2012. [Online]. Available: http://hdl.handle.net/2099.1/15251

[4] S. Kalidindi, M. J. Zekauskas, and D. G. T. Almes, "A One-way Delay Metric for IPPM," RFC 2679, Sep. 1999. [Online]. Available: https://rfc-editor.org/rfc/rfc2679.txt

[5] G. Almes, S. Kalidindi, and M. J. Zekauskas, "A Round-trip Delay Metric for IPPM," RFC 2681, Sep. 1999. [Online]. Available: https://rfc-editor.org/rfc/rfc2681.txt

[6] M. Kihl, R. Larsson, N. Unnervik, J. Haberkamm, A. Arvidsson, and A. Aurelius, "Analysis of facebook content demand patterns," in *2014 International Conference on Smart Communications in Network Technologies (SaCoNeT)*, June 2014, pp. 1–6. [Online]. Available: http://doi.acm.org/10.1109/SaCoNeT.2014.6867760

[7] J. P. Laulajainen, A. Arvidsson, T. Ojala, J. Seppanen, and M. Du, "Study of YouTube demand patterns in mixed public and campus WiFi network," in *2014 International Wireless Communications and Mobile Computing Conference (IWCMC)*, Aug 2014, pp. 635–641. [Online]. Available: http://doi.acm.org/10.1109/IWCMC.2014.6906430

[8] A. Cuadra-Sanchez and J. Aracil, "A novel blind traffic analysis technique for detection of WhatsApp VoIP calls," *International Journal of Network Management*, vol. 27, no. 2, 2017. [Online]. Available: http://doi.acm.org/10.1002/nem.1968

[9] P. de la Cruz Ramos, R. Pérez Leal, F. González Vidal, and L. Bellido Triana, *A Model for Quality of Experience Estimation Based on Quality of Service Parameter Monitoring for Multimedia Convergent Services (3-Play)*. John Wiley & Sons, Inc., 2014, pp. 191–223. [Online]. Available: http://dx.doi.org/10.1002/9781118984352.ch10

[10] Ixia. IxChariot. [Last access: December 20, 2017]. [Online]. Available: https://www.ixiacom.com/products/ixchariot

[11] SamKnows. SamKnows Whitebox. [Last access: December 20, 2017]. [Online]. Available: https://www.samknows.com

[12] S. Khirman and P. Henriksen, "Relationship Between Quality-of-Service and Quality-of-Experience for Public Internet Service," in *PAM 2002, Passive and Active Network Measurement workshop*, Mar. 2002. [Online]. Available: http://www.pamconf.net/2002/Relationship_Between_QoS_and_QoE.pdf

[13] K. D. Singh, Y. Hadjadj-Aoul, and G. Rubino, "Quality of experience estimation for adaptive HTTP/TCP video streaming using H.264/AVC," in *2012 IEEE Consumer Communications and Networking Conference (CCNC)*, Jan 2012, pp. 127–131. [Online]. Available: http://dx.doi.org/10.1109/CCNC.2012.6181070

[14] T. Maki, M. Varela, and D. Ammar, "A Layered Model for Quality Estimation of HTTP Video from QoS Measurements," in *2015 11th International Conference on Signal-Image Technology Internet-Based Systems (SITIS)*, Nov 2015, pp. 591–598. [Online]. Available: http://doi.acm.org/10.1109/SITIS.2015.41

[15] D. Rivera, N. Kushik, C. Fuenzalida, A. Cavalli, and N. Yevtushenko, "QoE Evaluation Based on QoS and QoBiz Parameters Applied to an OTT Service," in *2015 IEEE International Conference on Web Services*, June 2015, pp. 607–614. [Online]. Available: http://doi.acm.org/10.1109/ICWS.2015.86

[16] R. K. P. Mok, E. W. W. Chan, and R. K. C. Chang, "Measuring the quality of experience of HTTP video streaming," in *Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management, IM 2011, Dublin, Ireland, 23-27 May 2011*, 2011, pp. 485–492. [Online]. Available: http://dx.doi.org/10.1109/INM.2011.5990550

[17] P. Varga, G. Kún, G. Sey, I. Moldován, and P. Gelencsér, "Correlating User Perception and Measurable Network Properties: Experimenting with QoE," in *Autonomic Principles of IP Operations and Management*, G. Parr, D. Malone, and M. Ó Foghlú, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 218–221. [Online]. Available: https://doi.org/10.1007/11908852_19

[18] A. Michlmayr, F. Rosenberg, P. Leitner, and S. Dustdar, "Comprehensive QoS Monitoring of Web Services and Event-based SLA Violation Detection," in *Proceedings of the 4th International Workshop on Middleware for Service Oriented Computing*, ser. MWSOC '09. New York, NY, USA: ACM, 2009, pp. 1–6. [Online]. Available: http://doi.acm.org/10.1145/1657755.1657756

[19] A. Mansy, B. Ver Steeg, and M. Ammar, "Sabre: A client based technique for mitigating the buffer bloat effect of adaptive video flows," in *Proceedings of the 4th ACM Multimedia Systems Conference*, ser. MMSys '13. New York, NY, USA: ACM, 2013, pp. 214–225. [Online]. Available: http://doi.acm.org/10.1145/2483977.2484004

[20] I. Sodagar, "The mpeg-dash standard for multimedia streaming over the internet," *IEEE MultiMedia*, vol. 18, no. 4, pp. 62–67, 2011. [Online]. Available: http://doi.acm.org/10.1109/MMUL.2011.71

[21] R. Pantos and W. May, "HTTP Live Streaming," Internet Engineering Task Force, Internet-Draft draft-pantos-http-live-streaming-21, Mar. 2017, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-pantos-http-live-streaming-21

[22] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Throughput: A Simple Model and Its Empirical Validation," *SIGCOMM Comput. Commun. Rev.*, vol. 28, no. 4, pp. 303–314, Oct. 1998. [Online]. Available: http://doi.acm.org/10.1145/285243.285291

[23] M. Masse, *REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces*. O'Reilly Media, Inc., 2011.

[24] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, and S. Weerawarana, "Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI," *IEEE Internet Computing*, vol. 6, no. 2, pp. 86–93, 2002. [Online]. Available: https://doi.org/10.1109/4236.991449

[25] Alexa. Top Sites in Spain. [Last access: December 20, 2017]. [Online]. Available: https://www.alexa.com/topsites/countries/ES

[26] Internet Systems Consortium. dig. [Last access: December 20, 2017]. [Online]. Available: https://ftp.isc.org/isc/bind9/cur/9.11/doc/arm/man.dig.html

[27] Tribelab. TRANSUM Wireshark Plugin: Analyzing a Website Problem. Last access: December 20, 2017. [Online]. Available: https://community.tribelab.com/mod/page/view.php?id=492

[28] D. J. C. Klensin, "Simple Mail Transfer Protocol," RFC 5321, Oct. 2008. [Online]. Available: https://rfc-editor.org/rfc/rfc5321.txt

[29] M. Crispin, "Internet Message Access Protocol - VERSION 4rev1," RFC 3501, Mar. 2003. [Online]. Available: https://rfc-editor.org/rfc/rfc3501.txt

[30] M. T. Rose and J. G. Myers, "Post Office Protocol - Version 3," RFC 1939, May 1996. [Online]. Available: https://rfc-editor.org/rfc/rfc1939.txt

[31] W. Denniss and J. Bradley, "OAuth 2.0 for Native Apps," RFC 8252, Oct. 2017. [Online]. Available: https://rfc-editor.org/rfc/rfc8252.txt

[32] nginx. [Last access: December 20, 2017]. [Online]. Available: https://nginx.org/en/

[33] Elastic. Elasticsearch: RESTful, Distributed Search & Analytics. [Last access: December 20, 2017]. [Online]. Available: https://www.elastic.co/products/elasticsearch

[34] SQLite Home Page. [Last access: December 20, 2017]. [Online]. Available: https://sqlite.org/

[35] Flask (A Python Microframework). [Last access: December 20, 2017]. [Online]. Available: http://flask.pocoo.org/

[36] Grafana Labs. Grafana - The open platform for analytics and monitoring. [Last access: December 20, 2017]. [Online]. Available: https://grafana.com/

[37] Google, Inc. YouTube Data API. [Last access: December 20, 2017]. [Online]. Available: https://developers.google.com/youtube/v3/

[38] Spotify, Inc. Spotify Web API. [Last access: December 20, 2017]. [Online]. Available: https://developer.spotify.com/

[39] Mitmproxy: An interactive TLS-capable intercepting HTTP proxy for penetration testers and software developers. [Last access: December 20, 2017]. [Online]. Available: https://mitmproxy.org/

[40] Netflix, Inc. Internet Speedtest Fast.com. [Last access: December 20, 2017]. [Online]. Available: https://fast.com/en/

[41] Twitter, Inc. Twitter REST API reference. [Last access: December 20, 2017]. [Online]. Available: https://dev.twitter.com/rest/public

[42] Dropbox, Inc. Drobpox HTTP API reference. [Last access: December 20, 2017]. [Online]. Available: https://www.dropbox.com/developers/documentation/http/overview

[43] A. Hidayat. PhantomJS: Scriptable Headless WebKit . [Last access: December 20, 2017]. [Online]. Available: http://phantomjs.org/

Copyright Disclaimer

Copyright reserved by the author(s).

This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution license (http://creativecommons.org/licenses/by/3.0/).