A Framework for QoE Analysis of Encrypted Video Streams

Steve Göring* Alexander Raake* Bernhard Feiten†
*Audiovisual Technology Group, Technische Universität Ilmenau, Germany;
Email: [steve.goering, alexander.raake]@tu-ilmenau.de
†Deutsche Telekom AG, Technology & Innovation, Germany; Email: bernhard.feiten@telekom.de

***Abstract*—Today most internet traffic is generated by video streaming. YouTube and other video streaming platforms are using encrypted streams (HTTPS) for transport of video content. Encryption will lead to more requirements on network and content providers, e.g. caching mechanisms will not work direct. Estimation of video quality for measuring users satisfaction is also harder because there is no direct access to the video bitstream. We are building up a framework for analyzing video quality that allows us to store client information, decrypted network traffic and encrypted messages. Our approach is based on a man-in-the-middle proxy for storing the decrypted video bitstream, active probing and traffic shaping. Using these data, we are able to calculate video QoE values for example using a model such as ITU-T Rec. P.1203. Our framework will be used for generating datasets for encrypted video stream analysis, analyzing internal behavior of video streaming platforms, and more. For experimental evaluation, in this paper we analyze the influence of our man-in-the-middle proxy on key-performance indicators (KPIs) for video streaming quality.**

I. Introduction

YouTube, Vimeo, Netflix and Amazon Prime are well known video portals. Traditional broadcast television is more and more complemented and replaced by such video on demand platforms. Today about 70% of the internet traffic is generated by video streaming [2, p 14]. Technologies like 4k resolution, high framerate and HDR, will increase the traffic volume rapidly. Most of all popular video streaming providers are using variants of HTTP-based adaptive streaming (HAS) to deliver their video content, for reducing overhead and increasing performance [5]. Efficiency of YouTube's HAS was already analyzed [6]. For security or privacy reasons the content is, in most cases, transported encrypted using HTTPS [7]. Automatic estimation of video quality in such an encrypted scenario is not directly possible, because a good estimation requires access to the video bitstream. In this paper we describe a framework for automated analysis of video stream quality based on client-side and network measurements combined with bitstream analysis and ITU-T Rec. P.1203 (model for HAS quality estimation) [1, 3]. For accessing the video bitstream we built a system using a man-in-the-middle proxy and client-side automated active video probing and controlling. In addition, we add network traffic shaping for simulating different network conditions in our experiments conducted in a fast university network. Furthermore, using network shaping, our results are reproducible without considering changes in the way how the streaming platform delivers the video. All required decrypted network data was stored based on our proxy, enabling to analyze what a user would have watched, after assembling the video segments into separate video streams. Our active video probing approach is able to collect several data, e.g. stalling events, stalling duration and startup delay. Combining these collected data we can apply a video quality model such as P.1203, allowing to estimate MOS values of the played video. Orsolic et al. are analyzing encrypted video streams using machine learning and statistical calculations based on the encrypted stream [4]. Compared to our approach we are able to collect decrypted (bitstream) information, encrypted streams and meta information of a video. Our collected and generated data can be used for training and building up a database to analyze encrypted video streams. In this paper we use our framework for a small example evaluation. The database used for the experiments for this paper are made public available[1]. To evaluate the approach itself, we will measure to which extent the man-in-the-middle approach influences video performance compared to measurements without a proxy. Our framework is just a starting point for more automated video quality estimation and further analysis of encrypted video streams.

II. Our Approach

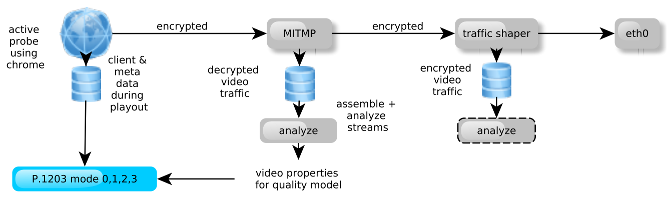

Fig. 1. Framework for measuring encrypted video streams; beside encrypted network traffic we also collect active probing results, video segment files and decrypted traffic.

Our approach can be divided into several steps, compare Fig. 1. First, we need a given video url (e.g. YouTube) and a set of network settings used during traffic shaping. For one measurement run, the man-in-the-middle proxy, a web browser session (we use google-chrome), encrypted network traffic recording, meta data extraction and our active probing script were started in parallel. The active probing step can be

[1]https://github.com/Telecommunication-Telemedia-Assessment/mitmprobe_validation_dataset

replaced by simulated user's behavior or a real user interaction transparently. After the selected video has completed playing, all data from the client side is stored (player load time, startup delay, video duration, average stalling duration, stalling events, quality events), and the analysis of the recorded man-in-the-middle dump is launched. Our analysis step assembles all segments into different video stream files and aggregates all stored and required video properties for the quality model.

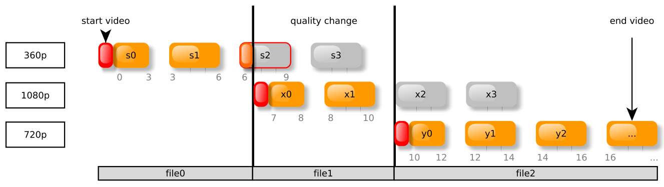

Fig. 2. Example of assembling a video stream; three different quality switches occur during video play-out and reassemble video files for our analysis.

Fig. 2 shows how different video segments are assembled. Assuming that each segment has a correct timestamp and that the current quality level is the most recent one, we assemble all segments to different video quality streams. For example, for a session starting from 360p, going to 1080p, and finally to 720p we will extract three different video stream files that have corrected starting, end times and no video overlap. We will not analyze additionally transfered segments, even though they are stored in our dump-files. A later analysis of those transferred segments can be done in a subsequent experiment, see e.g.[6].

After assembly, we are able to use all collected data as input for the P.1203 video model (each mode–0 to 3 can be used). The resulting MOS values for the overall session, per-second audio- and video MOS-scores and diagnostic KPI information can be stored in our video-quality report.

III. Experimental Evaluation

In the experiment presented here we analyze the influence of our man-in-the-middle-proxy pipeline on video play-out. We performed running measurements with and without proxy and compared based on the information collected at client side. Due to the fact that we compare with measurements without proxy, only quality-scores according to P.1203 mode 0 can be calculated. As input, P.1203 Mode 0 uses stalling, bitrate, codec and framerate. If these model input-parameters are in agreement between the cases with and without proxy, it can be assumed that there is no impact of the measurement on the results, and MOS-estimates will agree as well. Hence, in the validation presented here, we only focus on player load time, startup delay and average stalling duration. We analyzed three different YouTube videos of short (55s), medium (121s), and long duration (331s). In experiment, even longer videos are not suitable because YouTube's behavior and our emulated network conditions have distinct temporal characteristics. Our measurements were collected in the following way. First, we applied traffic shaping (simulation of dsl 2, 6, 25 Mbit/s parameters). Secondly we start measuring using our proxy. We store all results and then re-run without proxy. For each network condition and video we repeated this approach 32 times and calculated average values. As an example, player load time,

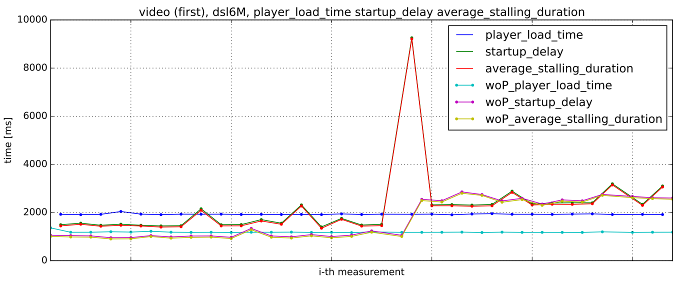

Fig. 3. 32 measurements; first video (55s); dsl 2 Mbit/s; notably are fluctuations during measurement period.

startup delay, average stalling duration of 32 measurements for our first video using dsl 2 Mbit/s parameter were presented in Fig. 3. There is an approximately constant offset in player load time between using our proxy `prx` and without proxy `wop`. The offset for all other measurements is not clearly constant and considering Fig. 3 measurement values are fluctuating in time. Hence, we also calculated mean difference (`wop` - `prx`) values for all experiments to get an better overview. Table I shows all mean differences of all experiment runs.

TABLE I
MEAN DIFFERENCES [MS] OF ALL EXPERIMENTS, WOP - PRX

video	dsl [bit/s]	avg stalling	player load time	startup delay
first (55 s)	2 M	8473	-620	8507
	6 M	-536	-742	-534
	25 M	-472	-749	-486
second (121 s)	2 M	9784	-463	8669
	6 M	-322	-637	-329
	25 M	-788	-651	-785
third (331 s)	2 M	-800	-447	-715
	6 M	-851	-595	-855
	25 M	-902	-651	-908

The player load time is nearly constant for every experiment. Startup delay and average stalling duration is different only for 2 exceptions, otherwise approximately constant. E.g, periods of additional congestion in our university network may explain this. Inter-measurement differences can be explained based on the fact that our interleaved measurement approach is not optimal because conditions can change after some seconds. The experiment represents a first approach for evaluating our man-in-the-middle-proxy pipeline. A more extensive evaluation will be performed in a future lab test where we run in an isolated environment to control all influences.

IV. Conclusion and Future Work

We introduced an automated framework for collecting encrypted video stream data to apply quality models that are based on decrypted bitstreams. Furthermore, we evaluated our approach in a small experiment, indicating that only a constant offset is created using the man-in-the-middle-proxy. Future work will be about extending our system to a distributed measurement tool, and building up a database for encrypted stream analysis. Additionally, we plan to run more in-depth analyses of the collected data.

V. References

[1] Alexander Raake, Marie-Neige Garcia, Werner Robitza, Peter List, Steve Göring and Bernhard Feiten. “Scalable Video Quality Model for ITU-T P.1203 (aka P.NATS) for Bitstream-based Monitoring of HTTP Adaptive Streaming”. In: *QoMEX 2017*. to appear. IEEE. 2017.

[2] Cisco. *Whitepaper: Cisco Visual Networking Index:Forecast and Methodology, 2015-2020*. 2015.

[3] ITU-T. *Recommendation P.1203 - Parametric bitstream-based quality assessment of progressive download and adaptive audiovisual streaming services over reliable transport*. Tech. rep. International Telecommunication Union, 2016.

[4] Irena Orsolic et al. “YouTube QoE Estimation Based on the Analysis of Encrypted Network Traffic Using Machine Learning”. In: *Globecom Works*. IEEE. 2016, pp. 1–6.

[5] Michael Seufert et al. “A survey on quality of experience of HTTP adaptive streaming”. In: *IEEE Communications Surveys & Tutorials* 17.1 (2015), pp. 469–492.

[6] Christian Sieber et al. “Sacrificing efficiency for quality of experience: YouTube’s redundant traffic behavior”. In: *IFIP Networking*. IEEE. 2016, pp. 503–511.

[7] *YouTube’s road to HTTPS*. https://youtube-eng.googleblog.com/2016/08/youtubes-road-to-https.html. Accessed: 2017-02-25.