Video Quality Representation Classification of Safari Encrypted DASH Streams

Ran Dubin, Ofer Hadar, Itai Richman, Ofir Trabelsi
Communication Systems Engineering
Ben-Gurion University of the Negev
Israel

Amit Dvir[1], Ofir Pele[1,2]
[1]Center for Cyber Technologies
[1]Department of Computer Science
[2]Department of Electrical and Electronics Engineering
[1]Ariel University
[1]Israel

***Abstract*—The increasing popularity of HTTP adaptive video streaming services has dramatically increased bandwidth requirements on operator networks, which attempt to shape their traffic through Deep Packet Inspection (DPI). However, Google and certain content providers have started to encrypt their video services. As a result, operators often encounter difficulties in shaping their encrypted video traffic via DPI. This highlights the need for new traffic classification methods for encrypted HTTP adaptive video streaming to enable smart traffic shaping. These new methods will have to effectively estimate the quality representation layer and playout buffer. We present a new method and show for the first time that video quality representation classification for (YouTube) encrypted HTTP adaptive streaming is possible. We analyze the performance of this classification method with Safari over HTTPS. Based on a large number of offline and online traffic classification experiments, we demonstrate that it can independently classify, in real time, every video segment into one of the quality representation layers with 96.13% average accuracy.**

***Index Terms*—HTTPS Video Streaming, Encrypted Traffic, Quality Representation Classification, Safari**

I. Introduction

Every day, hundreds of millions of Internet users view videos online [1]. Google encourages all website owners to switch from HTTP to HTTPS by taking into account whether sites use secure, encrypted connections as a signal in their ranking algorithms [2]. Additionally, YouTube network traffic is encrypted. The YouTube video streaming solution is based on Adaptive Streaming Over HTTP (DASH) [3]. DASH is a Multi Bit Rate (MBR) streaming method, designed to improve viewers' Quality of Experience (QoE) [4]–[17].

Network traffic classification algorithms use two main techniques: DPI packet content analysis and statistical feature classification [18]–[25]. However, their effectiveness for encrypted traffic is concentrated mainly in recognizing TLS/SSL handshake parameters that help recognize the application content types (video, chat, etc.) or the application name. They do not try to classify the video stream quality representation or provide any enrichment data on the video streams.

YouTube analysis was conducted in many aspects such as YouTube servers' location [4]–[6], [26]–[32]. Many recent works have suggested methods for encrypted traffic classification and several surveys have presented detailed description of the state of the art methods [18]–[20]. Several works have examined different statistical features [33]–[43]. Some of these features are not relevant for video streams classification. For instance, the packet size is often MTU size in video streaming, as it consumes high bandwidth and re-transmission occurs often. TCP parameters such as server sent bit rate, inter-arrival packet time, RTT and packet direction are weak features. To the best of our knowledge, this work is the first that classify quality representation of encrypted YouTube streams.

In this paper we present a novel video stream quality representation classification for DASH. We classify the video quality representation, and each feature (group of packets) is classified by itself without any dependencies on past or future samples. Our scheme was tested on various browsers with Adobe's Flash as the player over HTTPS network traffic on offline and online YouTube video traffic streams. Our method recognizes, the YouTube video traffic quality representation layer with 96.13% average accuracy.

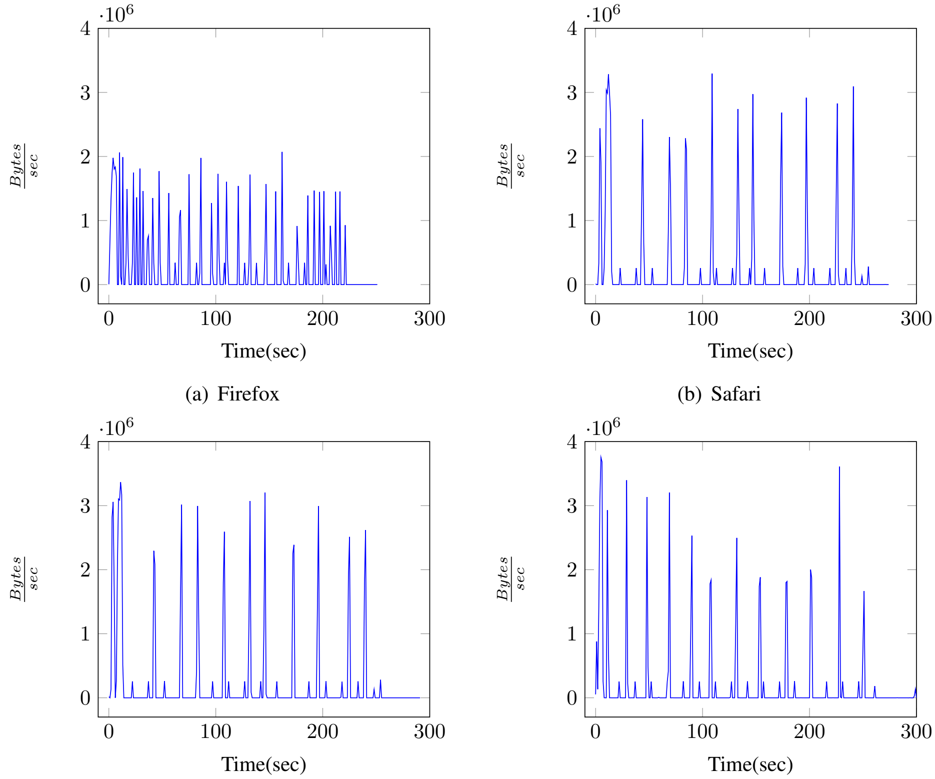
II. Video Quality Representation Classification

A DASH server stores a video which is segmented into fixed duration segments. Each segment is encoded into m representations (m can be different for different videos). The user selects whether to download a constant or adaptive representation. In the adaptive mode, the client's video player application (via adaptation logic), based on his network condition estimate and playout buffer selects a suitable representation to download each chunk which is a part of a segment.

A. Preprocessing

First, we divide the traffic into flows based on a five-tuple representation: {protocol (TCP/UDP), src IP, dst IP, src port, dst port}. Then, we decide for each flow whether it is a YouTube flow. This is done based on the Service Name Indication (SNI) field in the *Client Hello* message. If the "*googlevideos.com*" string is found in the SNI, the flow is passed to the next module. Note that, the YouTube flows identification can also be done using machine learning techniques [44], [45].

Second, we remove audio packets. The audio data and the video data can be found in the same 5-tuple flow and in some cases we cannot distinguish between them. This can result in



(a) Firefox

(b) Safari

(c) Explorer

(d) Chrome

Fig. 1. Traffic flows of auto mode downloads of the same movie from different browsers using flash player. On the time of this research, all major browsers used flash players. All flows have the same characteristics: *peaks* (of packets) with silences before and after. Note that the differences between the flows may be caused by: auto mode, network conditions, video container, video encoder, etc.

a classification error since the boundaries between the quality representations are very close.

Third, we remove the first and last peak in each flow. The first peak is large as the player fills the buffer and thus the quality is not related to the peak volume [5]. As our method is the first step for traffic shaping and the last peak is not relevant for traffic shaping, we remove it.

Finally, we remove TCP re-transmissions using a TCP stack [21] as re-transmissions are caused mostly by network conditions.

B. Feature Extraction

The feature extraction is done on the preprocessed traffic, where non-YouTube flows, most of the audio packets, the first and last peaks and TCP re-transmissions have been removed.

To better understand encrypted YouTube streaming traffic properties, we examined YouTube traffic under different browsers. Fig. 1 depicts traffic download patterns of auto quality representation using different browsers. In the figure we can see that all flows contains peaks. We decided to encode every peak of the streams to a feature. The feature is *bit per peak*. Note that, several works such as [46] done the same analysis focusing either in fixed quality or fixed network bandwidth where others such as [47], [48] showed the same characteristics (On/Off which is equivalent to peak).

C. Learning

The proposed classification solution is illustrated in Fig. 2. It has a training step and a testing step. In the training step, first, we constructed our dataset based on YouTube video streaming captures (PCAP trace files [49]). Each video was downloaded with the three following fixed qualities $\{360P, 480P, 720P\}$. Data was preprocessed and the *bit per peak* features were extracted. Afterwards, training data was clustered using k-means++ [50] (step (3) in Fig. 2). The end product of these steps is a code-book that represent the entire training dataset.

For each quality, for each time index we computed the average *bit per peak*. We now have an average *bit per peak* vector (its length is the maximum time index) for each quality. From these vectors and using the codebook from the k-means stage we computed a representative string for each quality. In the classification stage we carried out the *bit per peak* extraction for each segment and then assigned a symbol (the one with the shortest distance to the average) to it from the codebook. Finally, we assigned a label by finding which center was the closest.

III. Performance Evaluation

A. Dataset

The video titles used in this study are popular YouTube videos from different categories such as news, video action

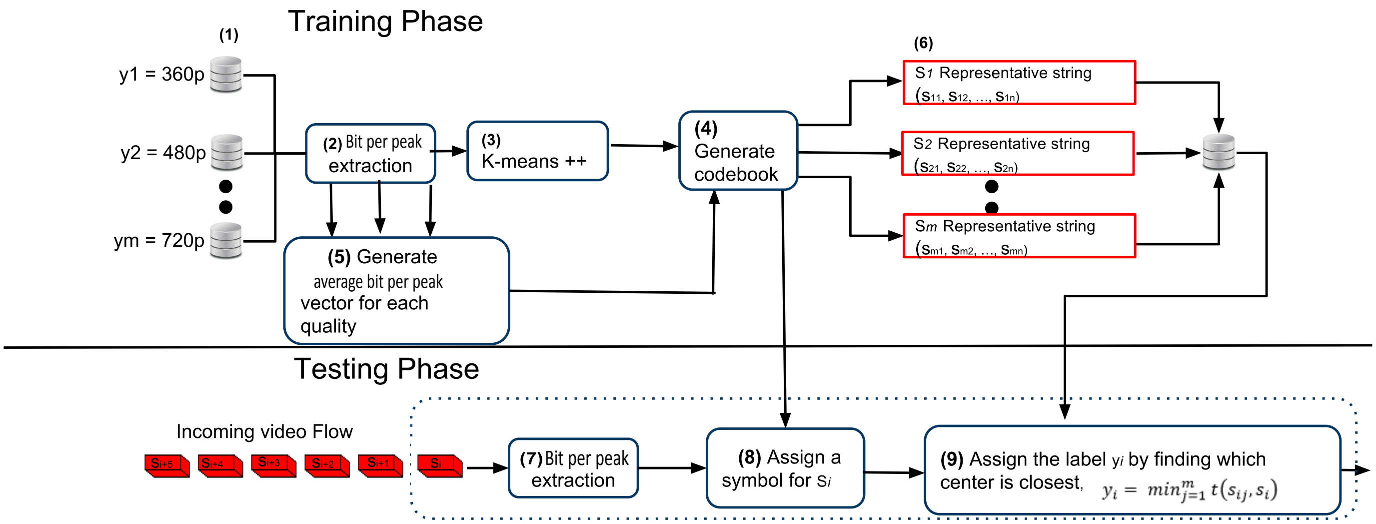
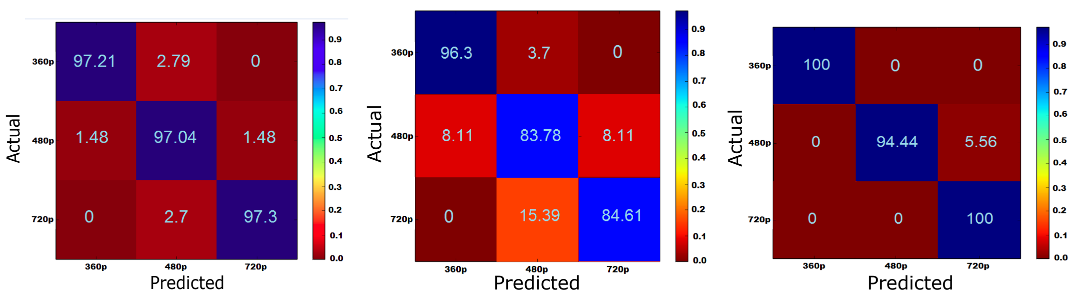


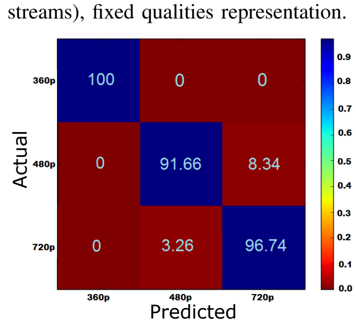
Fig. 2. Proposed algorithm diagram flow



(a) Proposed solution, 40 training video titles (different streams), fixed qualities representation.

(b) Nearest Neighbor using Average Bit Rate Feature(naïve), *test-fixed-train-titles*: 40 training video titles (different streams), fixed qualities representation.

(c) Proposed solution, *test-adaptive-train-titles*: 5 training video titles (different streams), auto quality representation.



(d) Proposed solution, *test-adaptive-test-titles*: 5 new videos titles (not seen in training), auto quality representation.

Fig. 3. Confusion matrices.

trailers and GoPro videos [49]. In this study we decided to focus on the Safari browser since the fixed quality download mode and the adaptive quality selection mode have similar characteristics. We show that for Safari, we can learn an accurate model for static or automatic quality modes simply by using a fixed training dataset. Future studies will add additional browsers. The training dataset contained 120 video streams of 40 unique video titles each of which was separately downloaded with fixed quality from the following qualities: $\{360P, 480P, 720P\}$.

We have three testing datasets: 1) *test-fixed-train-titles*: 120 video streams of 40 unique video titles (same titles as in the training phase) each of which was separately downloaded with a fixed quality from the following qualities: $\{360P, 480P, 720P\}$; 2) *test-adaptive-train-titles*: 5 video streams of 5 unique video titles (titles taken from the training phase titles) each of which was downloaded with an adaptive quality representation (auto mode); 3) *test-adaptive-test-titles*: 5 video streams of 5 unique video titles (new titles that were not in the training phase) each of which was downloaded with an adaptive quality representation (auto mode). All the test video streams were different from the ones that were used in the training phase (because of network conditions).

B. Accuracy Evaluation on the Different Test Sets

Fig. 3(a) shows that our classification errors in the fixed quality representation mode, are between close quality representations and were lower than 3%. The average classification accuracy was 2% better when we tested video titles from our training set (Fig. 3(c)) than when we tested video titles that were not in our training set (Fig. 3(d)).

We examined why the error of classifying $480P$ quality representation segments as $720P$ in adaptive streams was relatively higher than the other errors (see Figs. 3(c) and 3(d)). We found that when the quality representation switches from $360P$ to $480P$ there are high bit rate bursts. These bursts cause the erroneous classification of these segments as $720P$. In this work, we only trained the classifier based on the fixed quality switch mode. In future work, we will consider quality representation switches in our training.

C. Classifier Comparisons

Our proposed solution is the first classifier for encrypted adaptive video streaming over HTTPS. In this section, we describe and compare to two other new classification approaches: a naïve classifier and an algorithm based on a network traffic malware fingerprinting algorithm [51]. Since the malware fingerprinting is not designed for auto representation switching we used the fixed mode dataset in the tests. The naïve algorithm uses the average bit per peak for each quality. We used our entire fixed representation testing dataset and found the closest average quality bit rate for each feature. Fig. 3(b) illustrates the naïve approach and the proposed algorithm is presented in Figs. 3(a), 3(c), 3(d). Table I summarize this comparison. It shows that our proposed solution (with our bit per peak feature) achieved the highest identification results whereas all the other algorithms, especially using time differences obtained much lower identification results.

Feature	classifier	average confusion
Bit Per Peak	naïve	88.23%
Time differences	Shimoni et al. [51]	38.26%
Bit Per Peak	Shimoni et al. [51]	81.46%
Time differences	Proposed solution	62.21%
Bit Per Peak	Proposed solution	97.18%

TABLE I
COMPARISON OF THE DIFFERENT CLASSIFIERS AND FEATURE CREATION METHODS ON THE *test-fixed-train-titles* DATASET. NOTE THAT THE NAÏVE ALGORITHM IS BASED ON BIT PER PEAK FEATURES AND CANNOT BE USED WITH TIME DIFFERENCES.

IV. CONCLUSIONS

We propose a novel algorithm for YouTube HTTP adaptive video streaming quality representation classification. Our solution was tested on the Safari (Flash player) browser with offline and online network traffic over HTTPS. We achieved an average classification accuracy of 97.18% in the fixed mode, 98.14% in the adaptive mode where video titles (not streams) were from the training data, and 96.13% in the adaptive mode where video titles were not from the training data.

REFERENCES

[1] Cisco. The zettabyte era: Trends and analysis, 2015.
[2] Google. Google webmaster central blog: Https as a ranking signal, 2014.
[3] ISO/IEC. Information technology - Dynamic adaptive streaming over HTTP (DASH), May 2014.
[4] F. Wamser, M. Seufert, P. Casas, R. Irmer, P. Tran-Gia, and R. Schatz. Yomoapp: a tool for analyzing qoe of youtube http adaptive streaming in mobile networks. In *EuCNC*, 2015.
[5] Georgios Dimopoulos et al. Youtube traffic monitoring and analysis. 2012.
[6] Tobias Hoßfeld, Raimund Schatz, Ernst Biersack, and Louis Plissonneau. Internet video delivery in youtube: from traffic measurements to quality of experience. In *DTMA*. 2013.
[7] S. Lederer, C. Muller, and C. Timmerer. Towards peer-assisted dynamic adaptive streaming over http. In *PV*, 2012.
[8] István Ketykó, Katrien De Moor, Toon De Pessemier, Adrián Juan Verdejo, Kris Vanhecke, Wout Joseph, Luc Martens, and Lieven De Marez. Qoe measurement of mobile youtube video streaming. In *MoViD*, 2010.
[9] O. Oyman and S. Singh. Quality of experience for http adaptive streaming services. *CM*, 2012.
[10] Weiwei Li, Hamood-Ur Rehman, Mark H. Chignell, Alberto Leon-Garcia, Leon Zucherman, and Jie Jiang. Impact of retainability failures on video quality of experience. In *SITIS*, 2014.
[11] Weiwei Li, Hamood-Ur Rehman, Diba Kaya, Mark H. Chignell, Alberto Leon-Garcia, Leon Zucherman, and Jie Jiang. Video quality of experience in the presence of accessibility and retainability failures. In *ICHNQRSR*, 2014.
[12] P. Spachos, W. Li, M. Chignell, A. Leon-Garcia, J. Jiang, and L. Zucherman. Acceptability and quality of experience in over the top video. In *ICCW*, 2015.
[13] C. Müller, S. Lederer, and C. Timmerer. An evaluation of dynamic adaptive streaming over http in vehicular environments. In *MV*, 2012.
[14] C. Sieber. Implementation and user-centric comparison of a novel adaptation logic for dash with svc. In *IM*, 2013.
[15] C. Mr and C. Timmerer. A vlc media player plugin enabling dynamic adaptive streaming over http. In *ACM Multimedia*, 2011.
[16] C. Sieber, T. Hoßfeld, T. Zinner, P. Tran-Gia, and C. Timmerer. Implementation and User-centric Comparison of a Novel Adaptation Logic for DASH with SVC. In *IM*, 2013.
[17] I. Ben Mustafa and T. Nadeem. Dynamic traffic shaping technique for http adaptive video streaming using software defined networks. In *SECON*, 2015.

[18] A. Dainotti, A. Pescape, and KC. Claffy. Issues and future directions in traffic classification. *Network, IEEE*, 2012.

[19] S. Valenti, D. Rossi, A. Dainotti, A. Pescapè, A. Finamore, and M. Mellia. Reviewing traffic classification. In *DTMA*, 2013.

[20] Z. Cao, G. Xiong, Y. Zhao, Z. Li, and L. Guo. A survey on encrypted traffic classification. In *ATIS*, 2014.

[21] R. Dubin, O. Hadar, A. Noam, and R. Ohayon. Progressive download video rate traffic shaping using tcp window and deep packet inspection. In *WORLDCOMP*, 2012.

[22] B. Niemczyk and P.Rao. Identification over encrypted channels. In *BlackHat USA*, 2014.

[23] P. Fu, L. Guo, G. Xiong, and J. Meng. Classification research on ssl encrypted application. In *TCS*. 2013.

[24] P. Fu, G. Xiong, Y. Zhao, M. Song, and P. Zhang. An identification method based on ssl extension. In *SRAID*, 2013.

[25] M. Korczynski and A. Duda. Classifying service flows in the encrypted skype traffic. In *ICC*, 2012.

[26] Vijay Kumar Adhikari, Sourabh Jain, and Zhi-Li Zhang. Youtube traffic dynamics and its interplay with a tier-1 isp: An isp perspective. In *SIGCOMM*, 2010.

[27] Ruben Torres, Alessandro Finamore, Jin Ryong Kim, Marco Mellia, Maurizio M. Munafo, and Sanjay Rao. Dissecting video server selection strategies in the youtube cdn. ICDCS, 2011.

[28] Alessandro Finamore, Marco Mellia, Maurizio M. Munafò, Ruben Torres, and Sanjay G. Rao. Youtube everywhere: Impact of device and infrastructure synergies on user experience. IMC, 2011.

[29] Michael Zink, Kyoungwon Suh, Yu Gu, and Jim Kurose. Characteristics of youtube network traffic at a campus network - measurements, models, and implications. *CN*, 2009.

[30] Meeyoung Cha, Haewoon Kwak, Pablo Rodriguez, Yong yeol Ahn, and Sue Moon. I tube, you tube, everybody tubes: Analyzing the worlds largest user generated content video system. In *IMC*, 2007.

[31] Xianhui Che, B. Ip, and Ling Lin. A survey of current youtube video characteristics. *MultiMedia, IEEE*, 2015.

[32] Shane Alcock and Richard Nelson. Application flow control in youtube video streams. *CCR*, 2011.

[33] Vern Paxson. Empirically derived analytic models of wide-area tcp connections. *TON*, 1994.

[34] R. Alshammari and AN. Zincir-Heywood. Unveiling skype encrypted tunnels using gp. In *CES*, 2010.

[35] S. Zander, T. Nguyen, and G. Armitage. Self-learning ip traffic classification based on statistical flow characteristics. In *PANM*, 2005.

[36] D. Zhang, C. Zheng, H. Zhang, and H. Yu. Identification and analysis of skype peer-to-peer traffic. In *ICIW*, 2010.

[37] I. Paredes-Oliva, I. Castell-Uroz, P. Barlet-Ros, X. Dimitropoulos, and J. Sole-Pareta. Practical anomaly detection based on classifying frequent traffic patterns. In *INFOCOM WKSHPS*, 2012.

[38] D. Bonfiglio, M. Mellia, M. Meo, and D. Rossi. Detailed analysis of skype traffic. *Multimedia, IEEE Transactions on*, 2009.

[39] KT. Chen, CY. Huang, P. Huang, and CL. Lei. Quantifying skype user satisfaction. In *CCR*, 2006.

[40] E. Hjelmvik and W. John. Statistical protocol identification with spid: Preliminary results. In *SNCNW*, 2009.

[41] R. Bar-Yanai, M. Langberg, D. Peleg, and L. Roditty. Realtime classification for encrypted traffic. In *EA*, 2010.

[42] AM. White, AR. Matthews, KZ. Snow, and F. Monrose. Phonotactic reconstruction of encrypted voip conversations: Hookt on fon-iks. In *SP*, 2011.

[43] CV. Wright, L. Ballard, F. Monrose, and GM. Masson. Language identification of encrypted voip traffic: Alejandra y roberto or alice and bob? In *USENIX Security*, 2007.

[44] Peipei Fu, Li Guo, Gang Xiong, and Jiao Meng. Classification research on ssl encrypted application. In *TCS*. 2013.

[45] Guang-Lu Sun, Yibo Xue, Yingfei Dong, Dongsheng Wang, and Chenglong Li. An novel hybrid method for effectively classifying encrypted traffic. In *GLOBECOM*, 2010.

[46] Javier Aorga, Saioa Arrizabalaga, Beatriz Sedano, Maykel Alonso-Arce, and Jaizki Mendizabal. Youtubes dash implementation analysis. CSCC, 2015.

[47] Pablo Ameigeiras, Juan J. Ramos-Muoz, Jorge Navarro-Ortiz, and Juan M. Lpez-Soler. Analysis and modelling of youtube traffic. *TETT*, 2012.

[48] Ashwin Rao, Arnaud Legout, Yeon-sup Lim, Don Towsley, Chadi Barakat, and Walid Dabbous. Network characteristics of video streaming traffic. CoNEXT, 2011.

[49] Dataset of the Paper. `https://drive.google.com/folderview?id=0B_NMAPuEyaa6flNRcUY2QnVVWU1FczdZWEJRbDMzT09zSkd6T3FReHhRVndmNmVyaDcyQjA&usp=sharing`.

[50] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *ACM-SIAM*, 2007.

[51] A. Shimoni and S. Barhom. Malicious traffic detection using traffic fingerprint. https://github.com/arnons1/trafficfingerprint, 2014.