Novel HTTPS classifier driven by packet bursts, flows, and machine learning

Zdena Tropková
CTU in Prague
Prague, Czech Republic
tropkzde@fit.cvut.cz

Karel Hynek
CTU in Prague & CESNET
Prague, Czech Republic
hynekkar@fit.cvut.cz

Tomáš Čejka
CESNET
Prague, Czech Republic
cejkat@cesnet.cz

***Abstract*—Encryption of network traffic recently starts to cover remaining readable information, which is heavily used by current monitoring systems; thus, it is time to focus on novel methods of encrypted traffic analysis and classification. The aim of this paper is to define a new network traffic characteristic called Sequence of packet Burst Length and Time (SBLT), which was inspired by existing approaches and definitions. Contrary to other works, SBLT is feasible even for high-speed backbone networks as a part of IP flow data. The advantage of SBLT features is shown using a machine learning classification model for HTTPS traffic types as an example. This paper presents the definition of SBLT, proposes a new annotated public dataset of HTTPS traffic with 5 categories, and evaluates the developed classifier reaching accuracy over 99 %. This classifier can help analysts to deal with a huge amount of encrypted traffic and maintain situational awareness.**

***Index Terms*—classification, encrypted traffic, packet burst, IP flow, Machine Learning, HTTPS, TLS**

I. Introduction

Encrypted traffic is definitely beneficial for privacy reasons; however, it is also a great challenge to monitoring systems and network security tools. The rise of encrypted traffic on the internet is enormous in recent years. According to Google's transparency report[1], 95 % of Chrome traffic is encrypted. Unfortunately, the encrypted traffic can be easily leveraged for malicious purposes by threat actors, and the state-of-the-art monitoring solutions are still not capable of its processing. Therefore, the report [1] published by the European Union Agency for Cybersecurity recognizes encrypted traffic as a possible serious security threat. Also, this is the main motivation of the research of this paper — to develop a method to classify types of encrypted traffic.

Common network monitoring tools leverage all available readable information from the packets, mainly from Transport Layer Security protocol (TLS) handshake. One of the most important parameters usually sent in the first Client Hello packet is Server Name Indication, i.e., a hostname, which exposes domains visited by a user. The hostnames are a valuable source of information in encrypted traffic analysis for malware detection, parental control systems, or network statistics measurement. However, encrypted Client Hello (RFC draft [2]) extension is already supported in Firefox 85[2], and it hides even the hostnames from monitoring infrastructure.

The IP flow-based network monitoring systems started to deal with encrypted traffic monitoring by enriching the bidirectional flows for features that are not based on packet content. Cisco Joy[3] by Cisco Systems can export a Sequence of Packet Length and Time (SPLT) of up to the first 200 packets. It is possible to calculate multiple features from the SPLT, offering flexibility in designing the detection mechanisms. Nevertheless, the long SPLT comes with higher processing at the flow exporters and collectors and larger flow records, which also consume a significant portion of the monitored network's bandwidth, especially when dealing with backbone lines with more than 200 Gbps throughput. Therefore, there is a need for finding features that are similarly flexible as SPLT while significantly reducing the flow record size.

Based on the explained motivation, we studied the most common traffic on the internet — the HTTPS traffic. According to the literature and our observations and experiments, we defined five HTTPS traffic categories and created a large dataset of HTTPS traffic which came from environments of large ISP's backbone lines, and made it publicly available. By analysis of the problem with export limitation in mind, we proposed extending traditional IP flow records with a new feature called Sequence of packet Burst Length and Time (SBLT), which can aggregate large portions of information while maintaining a small flow records size. To show usability of SBLT in this paper, we created a machine learning model capable of HTTPS traffic category classification with high accuracy. Our classifier is helpful for network visibility, helping to raise a situational awareness during diagnosis of operational errors, and incident handling/response, where a large amount of benign traffic needs to be filtered out. Additionally, it can help with network optimization and overall statistics regardless of the Client Hello headers' content. However, SBLT is universal enough to be applied to many other network classification challenges.

II. Related Work

TLS is one of the most popular protocols for secure communication. Therefore, multiple research works challenged its security. Chen et al. [3] analyzed three web pages that use

[1]https://transparencyreport.google.com/https/overview

[2]https://blog.mozilla.org/security/2021/01/07/encrypted-client-hello-the-future-of-esni-in-firefox/

[3]https://github.com/cisco/joy

978-3-903176-36-2 ©2021 IFIP

TLS encryption. By observing SPLT features and utilizing the knowledge about the website, they could infer actions performed by the user in great detail. However, their approach is not based on IP flows and cannot work with SPLT of limited size, and thus it is not feasible for large-scale high-speed networks.

The recent studies, which focus mainly on content-less traffic classification, usually use deep learning methods [4]–[6]. The Deep learning approach has an indisputable advantage of self-finding the feature-set from the SPLT. However, they also usually require longer sequences, which cannot be exported on high-speed networks. Thus the classical method of manual feature engineering and statistical modeling is still common.

Luxemburk et al. [7] studied HTTPS brute-force attacks and solved the short SPLT by concatenating flows with the same IP addresses and destination port. Even though the presented approach works well and the models achieved the F1 score of 0.962, it is not universal and cannot be applied for every use case.

Hofstede et al. [8] also focused on HTTPS brute-force attack detection; however, instead of SPLT, they enriched IP flow for the histogram of packet sizes. The histogram aggregates the SPLT vector; thus, they are much more suitable for high-speed monitoring. Their algorithm outperformed previous works with a recall of 59 %.

Similarly as histograms, packet burst statistics are also a form of aggregation of SPLT. The usefulness of packet bursts in traffic analysis was mentioned by Dyer et al. [9], and Shi et al. [10] in terms of website fingerprinting. Leroux et al. [11] used these studies as a basis for classification. Their proposed classifier uses burst-like features for traffic classification into four categories. However, these studies do not work with flow-based data, and their burst definition vastly differs from ours since they do not assume HTTP/2 and SPDY protocol.

We are not aware of any research work aiming to classify TLS or specifically HTTPS traffic from IP flow-based data enriched for bursts' statistics.

III. Packet Burst

SPLT is too detailed for some cases. Typically, the SPLT record of even small multimedia transfers (such as pictures, smalls video parts) have the majority of packets with the maximal message transfer unit size (usually 1500 B) in one direction and little inter-packet times. The SPLT, with a small number of packets, covers only the beginning of the communication, unable to reveal the overall traffic shape. Therefore, we propose a "Sequence of packet Burst Length and Time" (SBLT) as an additional traffic property that can extend traditional IP flow records to represent information about later behavior of connections after the establishment.

A. Packet Burst Definition

Sarvotham et al. [12] define packet burst based on the average inter-packet times in the connection. However, this definition cannot be applied in our case since we aim to evaluate bursts on a running sequence of the packets without storing large packet sizes and timestamps sequences.

Research studies [9]–[11] define a burst as a sequence of packets sent in one direction that lies between two packets sent oppositely. The advantage of such a definition is its independence from connection parameters since it does not rely on any time constant. However, it requires a request/response type of traffic, such as HTTP/1. This definition does not fit the protocols that are capable of transmitting multiple unrelated communication within one connection, such as HTTP/2. A more suitable burst definition is formulated by Taylor et al. [13], which defines a burst by the maximal size of inter-packet space. However, such definition allows creating packet bursts with a single packet, which confuses the classifier. Additionally, Taylor et al. [13] computed burst across all traffic regardless of IP addresses and ports, which is unusable in flow-based detection systems. Nevertheless, none of the previous definitions are suitable. Therefore, we formulated a new packet burst definition, inspired by Max-Interval Method [14] used in neural biology.

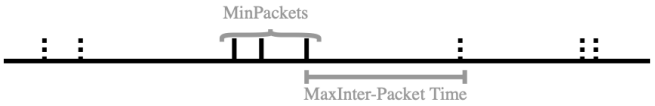

Fig. 1. Illustration of proposed burst definition and its parameters. Lines represent packets arrival in time. Dotted lines represent packets that do not belong to any packet-burst with *MinPackets = 3*

Proposed definition: The novel definition splits packet stream into two sequences based on the direction and calculates burst for each of them separately. The burst is defined based on two constants depicted in Fig. 1. The *MinPackets* constant specifies the minimal required number of concurrent packets that can form a burst. The purpose of this requirement is to filter out small multi-packet communications that are usually created by SETTINGS frames in HTTP/2 or by keepalive packets in other protocols. These small one-packet transfers are already included in SPLT sequence. Since HTTP/2 might split header and data into two packets, we opted to choose the minimal burst size of three packets, to select larger transfers and filter small and insignificant ones.

The *MaxInter-Packet time* constant specifies the requirement on the maximal time between two in-burst packets. Such value strongly depends on multiple unknown parameters such as connection quality, server's current load, and network jitter. Unfortunately, since the constant does not adapt to the connection parameters of each flow, it is impossible to set a threshold that would fit all situations.

We have experimented with various values of the Maximal inter-packet constant on one-hour browsing traffic, and the 1 s interval showed the best results. Even though it might seem like an immense value, smaller interval leads to burst fragmentation due to connection errors. Additionally, our aim is to pick up larger bursts during classical website browsing (such

as regular AJAX requests and responses) or video buffering.

To explore the usability of 1 s interval on a larger portion of traffic, we turned our faculty office into a laboratory for traffic measurement by deploying ipfixprobe[4] monitoring probe and tcpdump[5] on a SOHO router&Wi-Fi access point with OpenWrt connected via 1 Gbps symmetrical ethernet line to the internet. The laboratory consisted of 10 computers, five smartphones, and even a smartwatch (Apple Watch in our case). Computers and smartphones cover the most popular operating systems (Windows, Linux, macOS, iOS, and Android), and during the capture, all of them were used by university employees and students. In this laboratory, we run a virtual machine with Ubuntu 20.4 LTS with Firefox 86.0.1. By decrypting communication with exported TLS keys, we verified that 1 s interval is a good choice for burst separation since it identifies real communication bursts. The same experiment was also performed in our home networks connected with 20 and 50 Mbps asymmetrical VDSL lines with similar satisfactory results. Therefore, we opted to use it. Additionaly, the 1 s interval is also supported by the paper by Taylor et al. [13].

Additionally, we used our laboratory setup to analyze a large amount of traffic for network errors, resulting in unexpected behavior, including large interpacket spaces. As expected, we found in one-day traffic analysis traffic with larger interpacket spaces caused by higher retransmission timeout, representing a possible limitation of our burst definition. However, retransmissions can always be filtered out during the export process.

B. Burst Extension of IP Flow Data

Based on the definition from Sec. III-A and its parameters, we implemented plugin to IP flow exporter ipfixprobe. The plugin computes burst statistics and represents them in the form of IPFIX *basicList* records, which allow exporting arrays with variable length [15]. The calculated statistics for each burst are written in the Tab. I. The implementation created in the ipfixprobe plugin exports burst statistics as two sequences (one for each direction) of maximal length 10, which occupy similar bandwidth as SPLT of length 30. We did not filter out retransmitted packets due to performance reasons — retransmission filtering is computationally intensive on high-speed traffic.

For completeness, we have performed a stress test of the flow exporter with enabled SPLT and SBLT plugins. We used a server with 2x Intel® Xeon Gold 5218 CPU (2.30 GHz), 96 GB of RAM and 200 Gb/s network card. The exporter was capable of processing 170 Gb/s traffic with only negligible packet drops.

IV. Our Approach

We applied SBLT characteristics to HTTPS traffic classification problem. HTTPS is one of the most prevalent protocols that can be used for transferring various types of data. Thus this section describes the identified traffic types and features used for their classification.

[4]https://github.com/CESNET/ipfixprobe

[5]www.tcpdump.org

TABLE I
Calculated Burst statistics

Characteristics	Description
TimeStart	Timestamp of first packet in burst
TimeEnd	Timestamp of last packet in burst
Packets	Number of transferred packets in burst
Bytes	Number of transferred bytes in burst

A. HTTPS Traffic Categories

The Global Internet Phenomena Report [16] from Sandvine claims that most of the global traffic (in terms of volume of transferred data) is generated by Video (55 %), Web Pages (8 %), and File Sharing (3 %). However, the data in the report are not limited only to HTTPS traffic; therefore, we analyzed our one-day traffic from our laboratory, previously mentioned in Sec. III-A. These data were also used for the analysis of HTTPS traffic types and shapes. By labeling data by Server Name Indication from TLS Client Hello packet, we conclude that most of the laboratory traffic falls into five categories: *(i) Live Video Streaming, (ii) Video Player, (iii) Music Player, (iv) File Upload/Download, (v) Website and Other Traffic*

Our observations are consistent with the Global Internet Phenomena Report [16] since the identified categories fall into the 10 most prevalent traffic types categories; thus, we decided to use them.

B. Feature Engineering

From the thorough analysis of the identified traffic categories, we have created an initial feature vector of 75 characteristics. We used connection statistics such as the number of transferred bytes and packets in each direction, packet sizes statistics such as minimum, maximum, standard definition, and quantiles. The same statistics were calculated from burst bytes, burst packets, inter-burst spaces, and burst durations. Additionally, we used PCA dimension reduction on packet lengths and burst sequences. Last but not least, we included the first 10 individual burst sizes and the first 10 individual packet sizes into our feature vector. When flow does not contain at least 10 packets or 10 bursts, we filled the rest of the values with zeros.

V. Dataset Creation

Even though some previous studies published their TLS traffic datasets, their scope was mainly focused on one particular type of TLS traffic. Therefore, we decided to create our own data set and made it publicly available [17].

For each identified category in Sec. IV we have chosen the service representatives known for particular traffic type based on two website popularity lists[6,7]. We also used several popular websites that primarily focus on the audience in our country. The identified traffic classes and their representatives are provided below:

[6]https://s3.amazonaws.com/alexa-static/top-1m.csv.zip

[7]https://moz.com/top500

Live Video Stream Twitch, Czech TV[8], YouTube Live
Video Player DailyMotion, Stream.cz, Vimeo, YouTube
Music Player AppleMusic, Spotify, SoundCloud
File Upload/Download FileSender[9], OwnCloud, OneDrive, Google Drive
Website and Other Traffic Websites from Alexa Top 1M list[10]

However, since the dataset is the most important part of Machine Learning algorithms, the local data only would not be representative enough. The number of communication errors, connection speed, and unexpected user interactions are the corner cases that should be represented in the dataset. However, the corner cases are complicated to simulate realistically. Therefore, we used data from CESNET2 academic backbone infrastructure that connects more than half a million users to the internet.

Due to the limitation of the capture infrastructure of CESNET2 network, we could filter the wanted traffic only by IP addresses and ports. However, capturing *Website and Other* category traffic only by IP address would be very challenging since many requests are created dynamically by JavaScript to multiple servers. Thus we split the dataset into two parts. Computers in our laboratory generated the *Website and Other* traffic category, and other categories were captured in the backbone environment.

a) Generated capture: The website traffic was generated by an automated script in our laboratory previously mentioned in Sec. III-A. Our 1 Gbps symmetrical connection represents a typical end-user setup in the CESNET2 network. Users can also typically connect via Wi-Fi; therefore, we generated traffic via classical Ethernet and also Wi-Fi connections with the Wi-Fi access point located in our lab.

We used the script that commanded the browser to visit the first 200 (the first 100 were accessed by ethernet and the rest by Wi-Fi) most popular websites from the Alexa TOP 1 million list in random order random wait time (between 10–60 s) on each web-page. We used two Major browsers, Google Chrome and Mozilla Firefox in their default settings.

We are aware of possible artifact creation since we merge two data samples from different environments. Therefore, we used typical end-user connection setups used in the CESNET2 network to minimize this possibility.

b) Backbone traffic capture: The traffic categories other than *Website and Other* were captured directly on backbone lines, and the creation of this part of the dataset was done in multiple steps.

As a first step, we obtained address spaces that are used by representatives of each category. Then, we created traffic filters distributed on the ISP's network, measuring and capturing points located at its infrastructure's perimeter. As a result, there were PCAP files of raw backbone traffic that were automatically and immediately converted into enriched IP flow data with ipfixprobe exporter. The process also included automatic anonymization and filtering based on the SNI from the first step. The flow data contain the previously defined burst characteristics and SPLT of length 30.

[8]www.ceskatelevize.cz/ivysilani/
[9]filesender.cesnet.cz
[10]Includes social network, chat, and other various types of traffic

TABLE II
THE NUMBER OF IP FLOWS PER EACH GROUP

Traffic Category	Number of unique IP Flows
Live Video Stream	10,373
Video Player	12,553
Music Player	10,701
File Upload	10,862
File Download	20,393
Web Browsing	80,789

c) Dataset statistics: The dataset statistics are shown in Tab. II. The significant imbalance in the *Web Browsing* category is selected on purpose for more realistic evaluation because Web Browsing also outnumbers other types of HTTPS traffic in the natural network environment.

VI. EVALUATION

We used the annotated dataset from real backbone traffic described in Sec. V. The dataset was split with stratified sampling into the *Design part* and *Validation part* in ratio 7:3. The *Design part* was used with cross-validation for feature elimination and hyper-parameter selection during classifier creation. The *Validation part* was used for the testing of classifier performance.

We also applied the method for imbalanced learning since we do not have equally distributed classes in the dataset. We applied *random under-sampling* for the the *File Download* and *Web Browsing* category as it is one of the most common approach for dealing with imbalanced datasets (e.g., according to [18]). The dataset balancing method is applied only on the data given to the training phase of the algorithms since it is usually not recommended to apply it to the testing data.

A. Feature reduction

We have evaluated the contribution of each feature by using Random Forest Classifier on the *Design part* of the dataset, and we eliminated redundant features with zero or negative impact. The final set of selected features and their model importances (calculated with Gini index) can be seen in Fig. 2.

B. Performance of the multiple classifiers

In order to classify HTTPS traffic, we experimented with five ML algorithms: K-Nearest Neighbours (We use 5-NN in our study), Extremely Randomized Trees, Random Forest, and Gradient boosting. The input parameters (also called hyperparameters) of each algorithm were set experimentally by evaluating the combination of preselected values. The hyperparameter tuning was performed with 5-fold cross-validation on the *Design part* of the dataset.

The performance of each classifier can be found in Tab. III, according to which all classifiers achieved similar performance, which shows that our feature vector is very robust

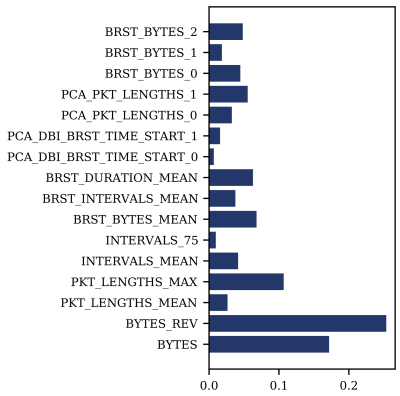

Fig. 2. The final feature vector and features' importances (Gini Index) for the Random Forest Classifier, which was used for feature selection.

TABLE III
COMPARISON OF THE OVERALL ACCURACY AND F1 SCORE OF EVALUATED CLASSIFIERS.

Algorithm	Accuracy	F1 score	Precision	Recall
Random forest	0.987	0.979	0.976	0.982
Extra trees	0.990	0.985	0.982	0.988
Gradient boosting	0.992	0.987	0.985	0.990
5-NN	0.917	0.868	0.857	0.882

and discriminative enough. However, we selected the Gradient Boosting classifier for further evaluation and inspection, which has the best accuracy.

Evaluation of Gradient Boosting Classifier: The Gradient boosting classifier was used with 300 trees; each of them was limited by a maximal depth of 10. The classifier was also used with a standard scaler and was evaluated in two ways. At first, we trained the classifier on the *Design dataset* and evaluated it on the *Validation dataset*. The results can be seen in the form of a confusion matrix in Tab. IV.

VII. CONCLUSION

HTTPS is one of the most prevalent protocols on the internet but is also very challenging for network analysis due to its encryption and wide usage. Current network monitoring tools are trying to deal with it mainly by using unencrypted information from TLS handshake; however, this information might not be available in the future [2]. Therefore, we have studied the feasibility of traffic type classification based on IP flows that do not rely on content inspection. Traffic type recognition helps to maintain situational awareness by security specialists in incident handling or response situations. The main contributions of this paper are the i) new dataset, ii) a novel extension of IP flow for SBLT, iii) the method for HTTPS traffic category classification, which achieves an F1 score of more than 0.99.

The proposed extension of IP flow data called SBLT overcomes the problems of SPLT with limited length. The SBLT proved to be an essential source of information for traffic type recognition, and it is the reason behind such high accuracy of our classification method. Overall, our results show that already existing SPLT combined with novel SBLT is a viable source of information for encrypted traffic analysis, particularly demonstrated in HTTPS.

TABLE IV
CONFUSION MATRIX OF GRADIENT BOOSTING TLS TRAFFIC CLASSIFIER. THE COLUMN HEADERS ARE AS FOLLOWS: **L** – LIVE VIDEO STREAM, **V** – VIDEO PLAYER, **M** – MUSIC PLAYER, **U** – FILE UPLOAD, **D** – FILE DOWNLOAD, AND **W** – WEBSITE AND OTHER TRAFFIC, **CP** – CLASS PRECISION, **CR** – CLASS RECALL

		Predicted Label					
		D	**L**	**M**	**P**	**U**	**W**
True Label	**D**	6018	23	33	32	12	0
	L	13	3073	12	12	0	2
	M	19	11	3176	2	0	2
	P	9	8	1	3743	0	5
	U	4	0	0	0	3255	0
	W	23	11	79	40	3	24 081
	CR	0.98	0.99	0.99	0.99	1.00	0.99
	CP	0.99	0.98	0.97	0.98	1.00	1.00

ACKNOWLEDGEMENT

This work was supported by the Technology Agency of the Czech Republic under grant No. TH04010073 (Smart ADS), and grant No. SGS20/210/OHK3/3T/18 by the Grant Agency of CTU in Prague funded by the MEYS of the Czech Republic.

REFERENCES

[1] D. Paraskevi et al., *Encrypted Traffic Analysis, Use Cases & Security Challenges*, 2020. [Online]. Available: https://www.enisa.europa.eu/publications/encrypted-traffic-analysis
[2] E. Rescorla et al., "TLS Encrypted Client Hello," Internet Engineering Task Force, Internet-Draft, 2021, work in Progress.
[3] S. Chen et al., "Side-channel leaks in web applications: A reality today, a challenge tomorrow," 2010.
[4] M. Lotfollahi et al., "Deep packet: a novel approach for encrypted traffic classification using deep learning," *Soft Computing*, vol. 24, 2019.
[5] Y. He and W. Li, "Image-based encrypted traffic classification with convolution neural networks," in *2020 IEEE Fifth DSC*, 2020.
[6] W. Zheng et al., "Learning to classify: A flow-based relation network for encrypted traffic classification," in *The Web Conference*. ACM, 2020.
[7] J. Luxemburk, K. Hynek, and T. Čejka, "Detection of https brute-force attacks with packet-level feature set," in *2021 IEEE 11th CCWC*, 2021.
[8] Rick Hofstede et al., "Flow-based web application brute-force attack and compromise detection," 2017. [Online]. Available: https://doi.org/10.1007/s10922-017-9421-4
[9] K. P. Dyer et al., "Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail," in *2012 IEEE SSP*, 2012.
[10] Y. Shi and S. Biswas, "Website fingerprinting using traffic analysis of dynamic web," in *2014 IEEE Global Communications Conference*, 2014.
[11] S. Leroux et al., "Fingerprinting encrypted network traffic types using machine learning," in *NOMS 2018*, 2018.
[12] Sarvotham et al., "Connection-level analysis and modeling of network traffic," in *1st ACM SIGCOMM*. New York, NY, USA: ACM, 2001.
[13] V. F. Taylor el al., "Robust smartphone app identification via encrypted network traffic analysis," vol. 13, 2018.
[14] E Cotterill et al., "Burst detection methods." Springer, 2019. [Online]. Available: https://doi.org/10.1007/978-3-030-11135-9_8
[15] B. Claise et al., "Export of Structured Data in IP Flow Information Export (IPFIX)," RFC 6313, 2011.
[16] Sandvine Inc., *Global Internet Phenomena: COVID-19 Spotlight*, 2020.
[17] Tropkova el al., "Dataset used for https traffic classification using packet burst statistics," 2021. [Online]. Available: https://zenodo.org/record/4911550
[18] L. González et al., "An empirical study of oversampling and undersampling methods for lcmine an emerging pattern based classifier," 2013.