MaMPF: Encrypted Traffic Classification Based on Multi-Attribute Markov Probability Fingerprints

Chang Liu[1,2], Zigang Cao[1,2], Gang Xiong[1,2], Gaopeng Gou[1,2], Siu-Ming Yiu[3], Longtao He[4]
1.Institute of Information Engineering, Chinese Academy of Sciences
2.School of Cyber Security, University of Chinese Academy of Sciences
3.Department of Computer Science, The University of Hong Kong
4.National Computer Network Emergency Response Technical Team/Coordination Center of China
caozigang@iie.ac.cn

***Abstract*—With the explosion of network applications, network anomaly detection and security management face a big challenge, of which the first and a fundamental step is traffic classification. However, for the sake of user privacy, encrypted communication protocols, e.g. the SSL/TLS protocol, are extensively used, which results in the ineffectiveness of traditional rule-based classification methods. Existing methods cannot have a satisfactory accuracy of encrypted traffic classification because of insufficient distinguishable characteristics. In this paper, we propose the Multi-attribute Markov Probability Fingerprints (MaMPF), for encrypted traffic classification. The key idea behind MaMPF is to consider multi-attributes, which includes a critical feature, namely "length block sequence" that captures the time-series packet lengths effectively using power-law distributions, and relative occurrence probabilities of all considered applications. Based on the message type and length block sequences, Markov models are trained and the probabilities of all the applications are concatenated as the fingerprints for classification. MaMPF achieves 96.4% TPR and 0.2% FPR performance on a real-world dataset from campus network (including 950,000+ encrypted traffic flows and covering 18 applications), and outperforms the state-of-the-art methods.**

***Index Terms*—Encrypted Traffic Classification, Power-law Division, Markov Model, Network Management**

I. Introduction

As information technology and network intercommunication are developing rapidly, the data volume of network traffic explodes at an amazing speed. For better network management, enormous network traffic data needs to be reasonably handled. The first step is the traffic classification which is significant for anomaly detection, which has drawn increasing attentions of academia and industries [1]–[12].

Traditional traffic classification methods can be summarized into two classes: port-based [2]–[4] and payload-based [5]–[7]. These methods rely heavily on matching with predefined rules with the assumption that we are able to see the plaintext in the traffic. However, these methods cannot handle encrypted traffic classification easily due to the encrypted contents. The problem of encrypted traffic classification has become a research hotspot [8]–[11].

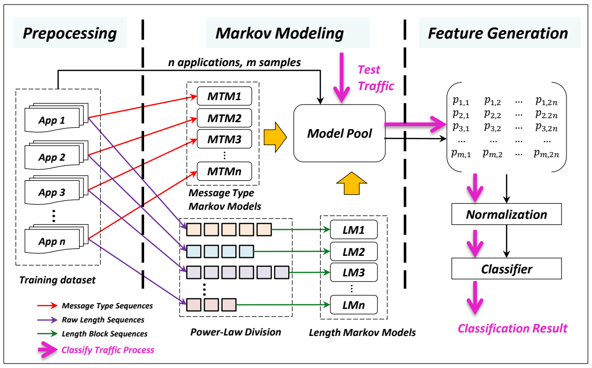


Fig. 1. The MaMPF Framework

To tackle this problem, machine learning methods with features from plaintext fields in the SSL/TLS handshake process [8], [9] and various packet/flow statistical information [10], [12] are being used. Moreover, the information embedded in the SSL/TLS sessions naturally constitutes a time series. Markov-based method was proposed by [13] to capture the fingerprints under message type sequences, which presents better performance.

However, we observed that it is very difficult to acquire discriminating fingerprints only based on the message type Markov model, because of the overlaps in Message Type Sequences (MTSs) from different applications as analyzed in Section III-B. Subsequent works tried to add the certificate packet length [14] and the first communication packet length [15] to improve the differentiating power of the fingerprints. However, these two length values from different applications could be clustered into one class, which finally results in the misclassification. Morevoer, the previous Markov models only consider individual application with maximum occurrence probability, while the classification result may depend on the relative occurrence probabilities from all the applications.

To further improve the differentiating power of fingerprints, we propose the "Length Block Sequence (LBS)", which considers the context of packets in a time series manner. To capture the relative occurrence probabilities of all applications, these probabilities are considered as fingerprints

978-1-5386-2542-2/18/$31.00 © 2018 IEEE

for classification, which considers the overall opinions of all the applications as the classification basis. We refer these two considerations as Multi-attribute property. We propose the Multi-attribute Markov Probability Fingerprints (MaMPF) framework, as shown in Figure 1, to solve the problem of encrypted traffic classification with insufficient differentiating fingerprints. Firstly, the raw packet length sequence is transformed into LBS based on the power-law distribution we discovered. Then, we separately use MTSs and LBSs of flows to build application Markov models, and concatenate the normalized probabilities as fingerprints to train the classifiers. We verify MaMPF with a real-world dataset which contains 950,000+ encrypted traffic flows and covers 18 popular applications. MaMPF achieves 96.4% TPR and 0.2% FPR, and outperforms the state-of-the-art methods.

Our contributions can be briefly summarized as follows:

- We propose the MaMPF framework for encrypted traffic classification based on Multi-attribute property, which not only considers both MTSs and LBSs of the flows, but also assigns importance to the attitudes of all the applications on the flows.
- Considering the processing of packet length sequences, power-law division is proposed based on our findings on the regularity of power-law distribution of application packet length values. Power-law division can transform length sequences into LBSs to build effective Markov models.
- MaMPF is applicable to various classifiers (linear or non-linear classifiers) with satisfactory performances on the real-world network traffic data, and outperforms the state-of-the-art methods.

The rest of this paper is organized as follows. We summarize the background and related work in Section II. The real-world encrypted traffic dataset and analysis are introduced in Section III. The power-law division to generate LBS is shown in Section IV. Section V describes the MaMPF building process, and Section VI presents the experiment results. Some discussions about traffic covering percentage and characteristic validity are presented in Section VII. Finally, we conclude this paper in Section VIII.

II. Background and Related Work

A. SSL/TLS Encrypted Traffic Classification Problem

The Secure Sockets Layer (SSL) [16] and its successor Transport Layer Security (TLS) protocol [17] are the most popular encrypted protocol, chosen by most applications. Taking full advantage of cryptographic technology, SSL/TLS protocol protects the user communication data from monitoring of attackers, but troubles network management. The SSL/TLS traffic flow generally includes the handshake process and the communication process. The handshake process is used to negotiate with secret keys with plaintext communication. However, not all handshake processes have the same procedures, which makes some classification methods lose efficiency because of missing information in some SSL/TLS flows. We just give an example here. When the client reconnects to the server with the same session ID which exists in the server session ID table, it is no need to exchange certificate and negotiate the secret key again. The client and the server will omit server certificate sending and verifying packets in the handshake process, and directly make a faster session with the ever session key. And in real-world network environment, this situation is very common. For example, when visiting the website under a poor network environment, frequent clicking and refreshing behaviors within a short period of time always happen, which produces many SSL/TLS traffic flows without the certificate procedure. Various real-world situations make SSL/TLS encrypted traffic more difficult to be classified.

B. Traffic Classification Methods

Conventional Traffic Classification: Port-based method [2] is provided by the Internet Assigned Numbers Authority (IANA) to identify the application type with a given list. However, more and more applications use dynamically assigned ports [1] or disguise their traffic with common communication protocol port [18]. Payload-based method [7] is also called as Deep Packet Inspect (DPI) technology, which finds out the unique signatures, i.e., some specific strings in the payload, used for matching in real time. S. Sen et al. choose application level signatures to classify P2P application traffic [19] while M. Roughan et al. uses statistical application signatures [20]. However, these methods can not be applied to encrypted traffic classification because of not parsing randomized ciphertext directly to acquire the signatures.

Encrypted Traffic Classification: Encrypted protocol, e.g. the SSL/TLS protocol, becomes more and more popular in application communication for user privacy, which makes the aforementioned approaches not practical. Machine learning (ML) techniques, independent to payload content, have been proposed for encrypted traffic classification [21]. Various ML algorithms, such as SVM [22], Naive Bayes [23] and Random Forest [24], are applied, but the primary challenge in ML-based encrypted traffic classification is feature construction. Although communication contents could not be parsed after encryption, some statistical features (e.g. packet length sequence [9] and arrival time sequence [25]) and the plaintext messages in the handshake process (e.g. ciphersuites and extension list [8]) could be used as basic classification features. However, to use these information as fingerprints directly could ignore the temporal relationship of packets in the flows.

Markov Model Fingerprints: First-order homogeneous Markov model fingerprints [13] is firstly provided by Maciej et al. as the state-of-the-art method for encrypted traffic classification. It is established with the message type field in each SSL/TLS packet header of application single-direction flows. However, this first Markov model only uses two states (current state and the previous one) of a flow to calculate the maximum likelihood. It loses the information of previous states and has many overlaps (weak distinguishing power) of similar flows. Based on it, [14] and [15] extend

TABLE I
THE STATISTICS OF 18 APPLICATION TRAFFIC DATA

ID	Applications	Strings in Domain Names	Flows	Packets
1	Alicdn	*.alicdn	16,560	124,206
2	Alipay	*.alipay	20,299	137,542
3	Apple	*.apple.*	111,471	779,756
4	Baidu	*.baidu, *.bdstatic	373,177	2,500,996
5	Github	*.github.com, *.github.io	7,488	84,618
6	Gmail	*.gmail	100,339	437,284
7	iCloud	*.icloud	22,993	150,278
8	JD	*.jd.*	48,146	177,041
9	Kaipanla	*.kaipanla.com	12,168	529,550
10	Mozilla	*.mozilla.*, *.cdn.mozilla.*	4,265	29,596
11	NeCmusic[2]	music.163.*	9,001	38,267
12	OneNote	*.onenote.*	6,486	52,840
13	QQ	*.qq.com	114,985	757,202
14	Sogou	*.sogou.com	4,498	24,251
15	Taobao	*.taobao.com	17,267	127,501
16	Weibo	*.weibo.*	24,289	12,1138
17	Youdao	*.youdao.com	46,545	163,557
18	Zhihu	*.zhihu.com	16,318	71,541

1. IDs are the corresponding codes of applications, which are used in later results of other contrast experiments.
2. NeCmusic means Netease Cloud Music.

to three states of one flow to build second-order Markov models, and incorporate certificate packet length and the first communication packet length. However, certificate packet may not exist in each SSL flow as discussed in Section II-A. In addition, W. Pan combines MTS and first several packet length to construct Markov model and Hidden Markov Model (HMM), and then use weighted ensemble classifiers to improve results [26]. However, the unsupervised HMM learning process is of high computational complexity and supervised HMM learning process needs the labeled hidden states. Furthermore, combining length and message type into one state increases the sparseness of the Markov transition matrix, which can easily lead to overfitting.

III. PRELIMINARIES

In this section, we show how we build the ground truth dataset. Based on our dataset, we analyze the overlaps in MTSs which are the limitation of previous Markov-based models.

A. Ground Truth Dataset

We capture traffic flows through specific routers in a campus network, meanwhile filter the non-SSL/TLS encrypted traffic. We collect the traffic flows for 7*24 hours long traces starting from July 20, 2017, and obtain 1.6 million SSL/TLS flows with 18.6 million packets as an initial dataset. We focus on the message type and packet length in each packet. In order to label these traffic flows, the value of Server Name Indication (SNI) is extracted and used, because its substring corresponds to the domain of one application. However, informal implementations of the SSL/TLS protocol [27] and fake SNI values [28] exist universally, which weaken the credibility of SNI values. Therefore, we refer to the method of [14] and take another two steps to enhance the reliability of the ground truth dataset. Firstly, we parse IP address of a flow into the corresponding domain name with the open web service, Whois [29]. Secondly, we confirm whether the specific exclusive string of the domain is consistent with the substring of the SNI value, as shown in Table I. The approach can indeed build a ground truth dataset to a certain degree, but there are three situations which could lead to the loss of the ground truth dataset: 1) The SNI value is null and 2) Whois cannot resolve an IP address into a domain name.

TABLE II
THE SSL/TLS MESSAGE TYPES

Type Index	Message Type	Type Index	Message Type
20	Change Cipher Spec	21	Alert
22:0	Hello Request	22:1	Client Hello
22:2	Server Hello	22:3	Hello Verify Request
22:4	New Session Ticket	22:11	Certificate
22:12	Server Key Exchange	22:13	Certificate Request
22:14	Server Hello Done	22:15	Certificate Verify
22:16	Client Key Exchange	22:20	Finished
23	Application Data		

* The type index of encrypted handshake message is only 22 which usually follows "Change Cipher Spec".

With packet recombination and flow reduction techniques, 956+ thousands of traffic flows corresponding to 18 applications are extracted from the initial traffic dataset by the above approach. The detailed flow information about each application with the corresponding ID is shown in Table I. Due to the fact that traffic data is directly captured from the campus network, there are some distinctions on the quantity scales of different application data. For example, Mozilla has the minimum traffic flow number, but still contains thousands of flows, which is typical and enough for our experiments.

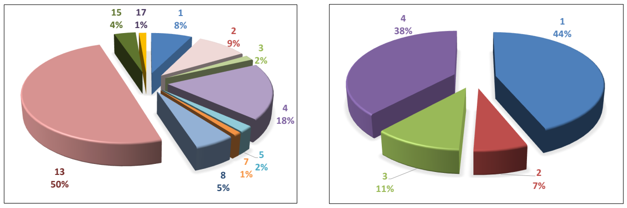
B. The Overlaps in Message Type Sequences

The MTS in an SSL/TLS flow is composed of message types which are shown in Table II. Considering client individual configurations, only server-side message types are used by [13] and [14] to build Markov models. The nature of Messgage Type transition (MT-transition) is the probability of current state on the basis of the previous states. For example, given a discrete random variable S_t that changes at each time step $t = t_0, t_1, ..., t_n$, the value s_t at time step t is the message type, e.g. "22:2" or "22:2,22:11", of t-th packet of MTS in one TCP segment of a traffic flow. Based on properties of the homogeneous first-order Markov model, the current status only depends on the previous one state as shown in Eq. (1).

$$\begin{aligned} &P(S_t = s_t | S_{t-1} = s_{t-1}, ..., S_1 = s_1) \\ =&P(S_t = s_t | S_{t-1} = s_{t-1}) \end{aligned} \tag{1}$$

The accuracy of the MT-transition method is usually unstable for various applications. There are two reasons:

1) One is that the current state transition may be influenced by the previous several states rather than one or two states. Either two discrete states [13] or three discrete states [14] cannot represent the entire flow. Although more states may solve this problem to some extent, the effect of classification accuracy decreases as the order of Markov model becomes larger. Incorporating more



(a) (b)

Fig. 2. Two examples of the MTS coincidence situations (The label of each part is the corresponding ID). (a) shows the application distribution of the state sequence "22:2,20:,22:-23:-23:". (b) shows the application distribution of state sequence "22:2,22:11,22:12-22:14-22:4,20:,22:-23:-23:-23:-23:-21:".

states, the Markov transition matrix is sparse and the Markov model is under a high risk of overfitting. The size of transition matrix increases exponentially with the order number too.

2) The other reason is the overlaps of MTSs from different applications. We count the same MTSs of different applications from our traffic dataset and display 2 examples selected from 2000+ coincidence situations in Figure 2. Many applications have the same MTS, which is ambiguous to determine which application the flow belongs to only with MTS. In other words, the MTS offers limited expressiveness and is insufficient to discriminate all the applications.

Additional information beyond MTSs needs to be mined and used. An intuition idea is to model the entire packet length sequences together with the MTSs to enhance the classification performance.

IV. LBS with Power-Law Division

In this section, we firstly define the LBS from packet length sequence. Then, the power-law distribution is presented. Finally, we provide the power-law division to generate LBS.

A. Length Block Sequence

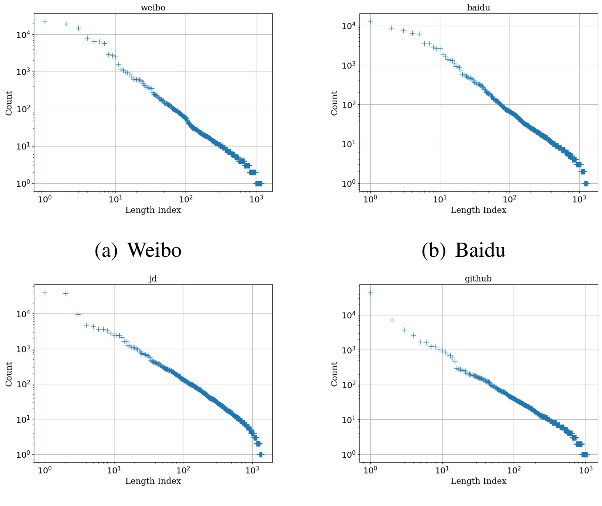
Directly uses the origin packet length values may increase the risk of overfitting due to the large range of length values. One essential way to generalize the packet length values into several representative length blocks, i.e. ranges of packet length that occur frequently for a specific application.

Definition 1: **Length blocks** are the split points on the length value range. Length blocks split the length value range into several blocks, and the length values that do not fall in any length blocks can be represented by the nearest length block.

With the length blocks, we can transform packet length sequences into LBSs.

Definition 2: **Length block sequence (LBS)** is a sequence whose value at time step t is the length block value transformed from the origin packet length value.

LBS has the same length as the raw packet length sequence with a smaller value ranges by omitting the precise length information which may not be effective and cause the overfitting problem. Next, we show how we can make use of power-law distributions to construct LBS effectively for different applications.



(a) Weibo (b) Baidu

(c) JD (d) Github

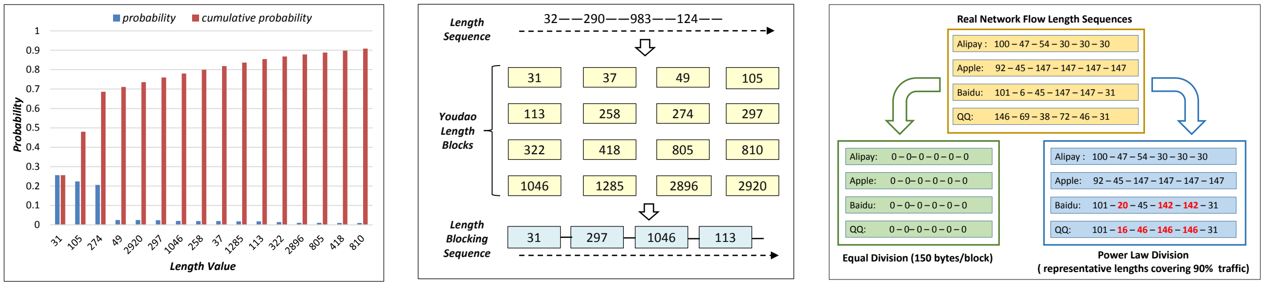
Fig. 3. The Distribution of Sorted Application Packet Length

B. Power-Law Distribution

For each application, we investigate the packet length sequences. We firstly count the frequency of each length value, and get the set of [length value, frequency] pairs. We sort the [length value, frequency] pairs with the frequency from large to small to obtain a sorted list. Then, we re-index the length values by their corresponding rank in the sorted list. For example, "*[120, 24], [180, 15], [232, 36], [256, 56]*" can be sorted as "*[256, 56], [232, 36], [120, 24], [180, 15]*". After re-indexing, it becomes "*[1, 56], [2, 36], [3, 24], [4, 15]*". With the above transformation methods, we can draw the packet length distribution of each application. Due to the space limitation, we display the distribution of Weibo, Baidu, JD and Github in Figure 3, and other omitted applications show similar distributions. When the x-axis and y-axis take the logarithm, the scatter diagram seems like a straight line. That is to say, the packet length of each application is consistent with the **power-law distribution**, i.e., $y = ce^{bx}$. Furthermore, the c and b of different applications have significant differences. We take JD and Github as an example. The curve function of JD is $y = 189395 * x^{-1.56}$ while that of Github is $y = 31221x^{-1.436}$, which indicates that the distribution could become a distinctive characteristic for classification.

C. Power-Law Division

Based on the power-law distribution, we can select length blocks of an application that can cover the majority of packet length values of the application to transform packet length sequence into LBS.


probability
cumulative probability
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Probability
31
105
274
49
2920
297
1046
258
37
1285
113
322
2896
805
418
810
Length Value
Length Sequence
32——290——983——124——
Youdao Length Blocks
31
37
49
105
113
258
274
297
322
418
805
810
1046
1285
2896
2920
Length Blocking Sequence
31
297
1046
113
Real Network Flow Length Sequences
Alipay : 100 – 47 – 54 – 30 – 30 – 30
Apple: 92 – 45 – 147 – 147 – 147 – 147
Baidu: 101 – 6 – 45 – 147 – 147 – 31
QQ: 146 – 69 – 38 – 72 – 46 – 31
Alipay: 0 – 0– 0 – 0 – 0 – 0
Apple: 0 – 0– 0 – 0 – 0 – 0
Baidu: 0 – 0– 0 – 0 – 0 – 0
QQ: 0 – 0– 0 – 0 – 0 – 0
Equal Division (150 bytes/block)
Alipay : 100 – 47 – 54 – 30 – 30 – 30
Apple: 92 – 45 – 147 – 147 – 147 – 147
Baidu: 101 – 20 – 45 – 142 – 142 – 31
QQ: 101 – 16 – 46 – 146 – 146 – 31
Power Law Division
(representative lengths covering 90% traffic)

(a) Distribution of Youdao Packet Length

(b) LBS Transformation of Youdao Flows

(c) The Representability of Power-Law Division

Fig. 4. (a) and (b) are examples of power-law distribution and division of Youdao Traffic Flows. And (c) is the comparison between power-law division and equal-length segment.

1) Obtain Length blocks: To obtain the length block of an application, let $C_{pj} = \sum_{i,i \geqslant j} c_{pi}$ be the cumulative length count for application p, where j is the length rank and c_{pi} means the count corresponding to the length rank i. T_p is defined as the whole length count for p-th application. Thus, the traffic covering percentage for p-th application is $R_p = C_{pj}/T_p$. Due to the properties of power-law distribution [30], the growth speed of the length count decreases as the length rank increases, which means that the representation ability of the length value decreases. Therefore, the length values at the prior of power-law distribution are the length blocks.

2) Obtain LBSs: Based on Definition 2, we can obtain the LBS of a flow. We take the cumulative coverage of YouDao as an example shown in Figure 4(a) and 4(b). From Figure 4(a), we can see 16 length blocks can cover almost 90% YouDao communication traffic because of power-law distribution. Then, we can turn any YouDao traffic length value into the nearest one of these 16 length blocks with minimum euclidean distance. As Figure 4(b) displays, each packet length sequence can be translated into the corresponding LBS with the length blocks of YouDao covering 90% traffic.

3) Advantages: Our power-law based division method for each application shows representability and robust to transform the original packet length sequences.

From the application perspective, our power-law based division method considers the views of all the applications. Different applications have distinguishable parameters as described in Section IV-B, which can increase the discriminating power of LBSs. However, traditional division method is equal-length segment for all the applications [8]. All the application share the same split points, which results in the overlaps in the LBSs. In our datasets, over 600 thousand raw length sequences are transformed into 1940 LBSs by equal-length segment. Here, we take flow samples with the same MTS of 4 applications from our dataset as an example, and the results in Figure 4(c) show the LBSs from equal-length segment and power-law division. Obviously, when these packets of flows are divided by equal length (150 bytes/block), they will be classified into the same class, no matter what classification methods are used. On the contrary, the power-law division can still maintain their discrimination power.

Moreover, power-law division is robust. The discrimination of packet lengths is kept under circumstance of limited block numbers exhibited in Figure 5. Figure 5 displays the length blocks with 90% traffic covering percentage. we can see that the amount of packet length values can be substituted by length blocks according to the aggregation characteristic of power-law distribution. Under the requirement of covering 90% traffic packets, we only need 10.8% of the total number of length values for one application in average. In other words, a lot of packet lengths remain unchanged due to the properties of power-law distribution, which maintains the discrimination power. Even with a relative low covering percentage, the power-law division still performs well as shown in Section VII-A.

In summary, the LBSs from power-law division provides a more discriminating fingerprint than the previous methods, which enhances the performance of encrypted traffic classification.

V. Multi-Attribute Markov Probability Fingerprints

MaMPF for encrypted traffic classification consists of four modules, as shown in Figure 1.

- *Prepocessing Module* is composed of filtering and extracting procedures. The filtering procedure is to pick the SSL/TLS traffic flows. The extracting procedure extracts the raw sequences (i.e., MTSs and packet length sequences) from these SSL/TLS traffic flows, and sends these sequences to Markov modeling module.
- *Markov Modeling Module* transforms the packet length sequence into LBSs, learns Markov models from the MTSs and LBSs, and saves these models in the model pool.
- *Feature Generation Module* transforms each flow into normalized probability features for classification.
- *Classification Module* learns a classifier from the feature vectors given by the feature generation module. And it predicts the application labels for the test flows.

In the following, the Markov modeling module, feature generation module and classification module are introduced in detail.

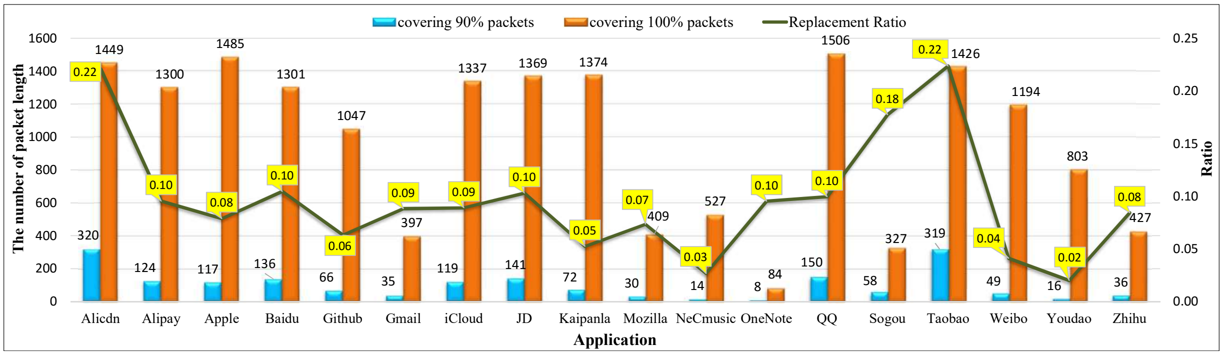


Fig. 5. The Length Block Numbers Cover 90% Packets and 100% Packets

A. Markov Modeling Module

In Markov modeling module, the packet length sequences are transformed into LBSs based on the power-law distribution of different applications as described in Section IV-C. And MTSs and LBSs are used to learn the Markov models.

The Markov modeling module is mainly a model pool which consists of message type Markov models with MT-Transition matrices and length Markov models with LB-Transition matrices. For each application, the MTSs and LBSs are used to train the Markov models respectively, i.e., the message type sequences and length block sequences of all the traffic flows for one application are respectively used to build the MT-transition matrix and the LB-transition matrix. We use first-order Markov models like [13] to model MTSs and LBSs for simplicity and generalization. Given discrete time random variable S_t and L_t for any $t \in \{t_0, t_1, ..., t_n\}$, the message type value at time step t of a flow is s_t, and the length block at time t is l_t. The MTS Markov model can be modeled based on Eq. (1), while the LBS Markov model can be similarly modeled as shown in Eq. (2).

$$\begin{aligned} &P(L_t = l_t | L_{t-1} = l_{t-1}, ..., L_1 = l_1) \\ =&P(L_t = l_t | L_{t-1} = l_{t-1}) \end{aligned} \tag{2}$$

The corresponding enter and exit probability distributions, i.e., EN and EX, are defined as in [13]. The EN_i represents the probability when flows start with state i, while the EX_i represents the probability when the flows end with state i. Finally, the trained message type Markov models and length Markov models are saved in the model pool to generate probability features. Specifically, there are $2n$ Markov models (i.e., n message type Markov models and n length Markov models) in the model pool, if n applications are needed to be classified.

B. Feature Generation Module

In feature generation module, each training flow needs to be put into all the Markov models in the Markov modeling module and get the probabilities of all these models as features for classification. In order to eliminate the effect of different packet numbers of flows and improve the classification accuracy, reasonable normalization is applied to the original probability features. The final probability features is our MaMPF, which models the views of all the applications on one flow.

1) Probability Feature Vectors: The MTS and LBS of each training traffic flow are sent to all the corresponding Markov models respectively in Markov modeling module to get the raw probability features. The Markov models produce the occurrence probabilities of flows, and the probabilities are concatenated as the probability vectors for flows. As Figure 1 shows, each row of the probability matrix is the probability vectors for a flow, and each column is generated by the same Markov model.

2) Root Normalization: The number of packets in one flow has a direct effect on the probability of multiplication, i.e., the occurrence probability of a flow is nearly exponential decay with the length of the flow, due to the fact that the transition probability between any two states is less than 1. This leads to most of the original probabilities prefer to concentrate around 0, which cannot distinguish applications easily. For example, the output probability of a 20-packet flow is far smaller than that of a 2-packet flow, however, these two flows may belong to the same application. To eliminate the effect of flow length, the root normalization is used. Given a flow with n packets, the final probability feature of the flow is the n-th root of the origin probability. The probability after root normalization measures the average contribution of all the packets in one flow to the occurrence of the flow. The root normalization makes each probability feature homogeneous, which is benefit for classification.

C. Classification Module

The classification module contains the training part and prediction part. In the training part, the normalized probability feature vectors with the corresponding application labels are used to train a classifier. And in the prediction part, the normalized probability feature vector of one test flow from the feature generation module is sent to the trained classifier to predict the application label. The core task is to choose the suitable classification method. The comparisons with different classifiers are shown in Section VII-B. The MaMPF probability features show satisfactory results with both linear (i.e., linear support vector machine and logistic regression) and non-linear (i.e., gradient boosted decision tree and random forest) classifiers. Therefore, the MaMPF features are robust with different classifiers. Depending on the usage, the classification module can be designed in a flexible manner. If only a reasonable quality, but fast real-time classification is needed, linear classification can be used. If a high classification quality is strictly required, non-linear classifiers can be adopted.

VI. Experiments

In this section, we first introduce the comparison methods and the assessment criteria. Then, comprehensive experiments are presented and discussed.

A. Experimental Setting

1) Methods in Comparison: We conduct experiments to compare some variants of our MaMPF with the state-of-the-art methods as follows:

- *FoSM* uses message type sequences to build first-order message type Markov models, and adopt maximum likelihood estimation to classify applications [13].
- *SoSM* is analogous to FoSM, but takes second-order message type Markov models.
- *SOCRT* blends with the certificate packet length based on SoSM [14] to classify applications.
- *SOB* considers the certificate packet and the first communication packet lengths based on SoSM [15].
- *SMPF*, namely State Markov Probability Fingerprints, is the variant of our MaMPF which only models message type sequences.
- *SLaveMPF*, namely State and Length-average Markov Probability Fingerprints, as the variant of our MaMPF, uses the message type sequences and length sequences with equal-segment (i.e., length average sequences). The segment length is 150.

MaMPF, SMPF and SlaveMPF take 90% traffic covering percentage and adopt Logistic Regression with L2 regularization.

2) Cross Validation: With the purpose of obtaining a reliable and stable model and eliminating contingency, we establish a 5-fold cross-validation. More specifically, we split the total dataset into five folds, and every time four shares are used for training while one share is used for testing. All the process repeats five times with different parts. And due to the limitation of space, all the results shown in Tables are the average value of five-fold cross validation results.

3) Assessment Criteria: We focus on the True Positive Rate (TPR), False Positive Rate (FPR) and FTF for evaluation. TPR means the rate of correctly identified as a given application, while FPR means the rate of wrongly identified as another application. We define TPR_{AVE} as the ratio between all the rightly classified flows and the total flows in Eq. (3):

$$TPR_{AVE} = \frac{1}{AFlN}\sum_{i=0}^{n} TPR_i * FlN_i \tag{3}$$

and FPR_{AVE} as the ratio between all the wrongly classified flows and the total flows in Eq. (4):

$$FPR_{AVE} = \frac{1}{AFlN}\sum_{i=0}^{n} FPR_i * FlN_i \tag{4}$$

where n means the number of applications, i.e., 18 in our dataset. TPR_i and FPR_i represent two measures of application i. FlN_i is the flow number of application i and $AFlN$ means the total traffic flow number. Therefore, TPR_{AVE} and FPR_{AVE} are two overall classification measures of all the traffic flows rather than considering the specific application flows separately.

We also adapt FTF [14], which considers both TPR_i and FPR_i with the weight w_i of application i. The definition of FTF is shown in Eq. (5).

$$FTF = \sum_{i=0}^{n} w_i \frac{TPR_i}{1 + FPR_i} \tag{5}$$

where n means the number of applications, and w_i means the weight of each application i, which is the ratio between the flow number of application i and the total flow number. Higher TPR_i and lower FPR_i contribute higher FTF. Furthermore, FTF considers the different weights of applications, which means the classification accuracy of one application can affect the effect of the model to a greater extent if it is given more weight whereas less affected.

B. Comparison Results

1) Comparison with the State-of-the-art Methods: We applied FoSM, SoSM, SOCRT, SOB and MaMPF on the dataset described in Section III-A, and the results are shown in Table III. MaMPF has the best performance in TPR_{AVE} (94%), FPR_{AVE} (0.33%) and FTF (0.9333).

Overall speaking, certificate packet length (SOCRT, 0.6433 FTF) and the first communication packet length (SOB, 0.6563 FTF) can indeed improve the FTF of application classification, compared to FoSM (0.6125 FTF) and SoSM (0.6415 FTF). However, we can also see the improvement is not very significant. Similarly, TPR_{AVE} and FPR_{AVE} become better when using FoSM, SoSM, SOCRT, SOB, however, the improvement is also not very obvious. There are two reasons for these phenomena: 1) The certificate lengths and the first communication packet lengths of different applications may be clustered as one class, which weakens their discriminating power; 2) High accessing frequencies for one application often occur in a poor network environment, which leads to reconnection without the certificate verification process as described in Section II-A. Therefore, some traffic flows could have no certificate packet as an efficient feature for classification. However, our MaMPF solves the above problems well by importing LBSs and probability features of all the applications.

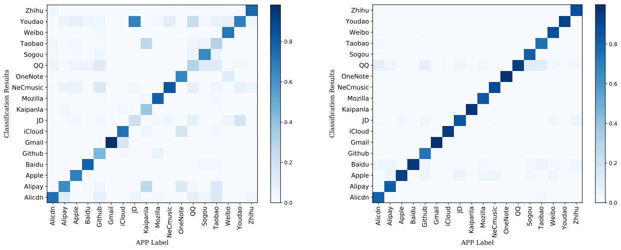
We further look at the detailed experimental results for each application. We can see that TPR and FPR are not stable for all the applications among the state-of-the-art methods (i.e., FoSM, SoSM, SOCRT and SOB). In other

TABLE III
EXPERIMENT RESULTS ON TPR, FPR AND FTF (THE BEST RESULTS ARE IN BOLD)

ID	APP	FoSM		SoSM		SOCRT		SOB		SMPF		SLaveMPF		MaMPF	
		TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR
1	Alicdn	0.5152	0.0061	0.6175	0.0242	0.6277	0.0258	0.6603	**0.0017**	0.5420	0.0064	0.7839	0.0029	**0.8042**	**0.0017**
2	Alipay	0.5417	0.0190	0.4535	0.0175	0.5689	0.0213	0.6357	**0.0012**	0.4547	0.0148	0.6167	0.0043	**0.8345**	0.0023
3	Apple	0.6391	0.0026	0.6455	**0.0023**	0.6465	0.0055	0.6471	0.0072	0.7516	0.0167	0.8253	0.0190	**0.9364**	0.0099
4	Baidu	0.7501	**0.0002**	0.7794	0.0040	0.7848	0.0044	0.8085	0.0244	0.8505	0.0428	0.8819	0.0349	**0.9608**	0.0085
5	Github	0.4703	0.0100	0.4340	0.0027	0.4472	0.0035	0.4500	**0.0009**	0.2986	0.0012	0.5403	0.0020	**0.7685**	0.0024
6	Gmail	0.9985	0.0076	0.9993	0.0040	0.9854	0.0040	0.9735	**0.0001**	**0.9998**	0.0077	**0.9998**	**0.0001**	**0.9998**	**0.0001**
7	iCloud	0.6352	0.0272	0.7360	0.0116	0.7169	0.0127	0.6852	0.0008	0.5621	0.0008	0.9529	0.0010	**0.9623**	**0.0006**
8	JD	0.0284	**0.0010**	0.1858	0.0258	0.1863	0.0261	0.1836	0.0083	0.7833	0.1067	0.4795	0.0166	**0.8610**	0.0074
9	Kaipanla	0.5237	0.0098	0.7344	0.0108	0.3295	0.0070	0.4016	0.0010	0.6044	0.0058	**0.9808**	0.0009	0.9685	**0.0006**
10	Mozilla	0.7977	0.0064	0.8293	0.0060	0.7719	0.0030	0.5196	**0.0004**	0.6335	0.0032	0.7461	0.0008	**0.8828**	0.0005
11	NeCmusic	0.8311	0.0334	0.8349	0.0323	0.8401	0.0323	0.8374	0.0009	0.0000	**0.0001**	0.7793	0.0029	**0.9051**	0.0011
12	OneNote	0.9851	0.0055	0.9692	0.0036	0.9689	0.0029	0.8555	**0.0000**	0.9488	0.0054	0.9871	0.0006	**0.9884**	0.0003
13	QQ	0.0931	**0.0148**	0.1299	0.0155	0.1307	0.0158	0.1923	0.0216	0.4108	0.0723	0.7031	0.0586	**0.9493**	0.0159
14	Sogou	0.7457	0.0477	0.6523	0.0300	0.4191	0.0227	0.6201	0.0001	0.0000	**0.0000**	0.5503	0.0006	**0.8653**	0.0004
15	Taobao	0.0665	0.0090	0.1323	0.0120	0.3291	0.0260	0.3002	0.0006	0.0012	**0.0001**	0.4922	0.0025	**0.7627**	0.0021
16	Weibo	0.5022	0.0066	0.7444	0.0216	0.8218	0.0182	0.7902	0.0037	0.7527	0.0168	0.8178	**0.0017**	**0.8941**	0.0023
17	Youdao	0.8549	0.1421	0.6806	0.1134	0.6764	0.1032	0.6561	0.0021	0.0000	**0.0000**	0.8551	0.0240	**0.9375**	0.0027
18	Zhihu	0.7518	0.0304	0.7755	0.0142	0.7774	0.0148	0.7805	**0.0006**	0.6988	0.0034	0.8577	**0.0006**	**0.8828**	0.0011
	AVE	0.6206	0.0211	0.6488	0.0195	0.6509	0.0194	0.6652	0.0186	0.6962	0.0169	0.8261	0.0097	**0.9400**	**0.0033**
	FTF	0.6125		0.6415		0.6433		0.6563		0.6725		0.8064		**0.9333**	

words, one method which improves the TPR and FPR of some applications may reduce the TPR and FPR of other applications. For example, although SOB can improve the performance in general, such as Alipay and Zhihu, but for some applications, it also decreases the performance, such as Gmail and Youdao. This instability on different applications is due to the overlaps of similar traffic flows. On the other hand, TPRs generated by MaMPF have significant advantages over other existing methods. Although the FPRs of several applications in FoSM, SoSM or SOB approaches can get a little better result (not over 1% improvement) than ours, such as Apple, Baidu and Github, their TPRs with other methods have obvious reduction (at least 20%) compared to MaMPF. Specially, MaMPF increases significantly the classification performances of JD (almost 70% improvement), QQ (70%-80% improvement) and Taobao (40%-70% improvement) by adding LBSs. There are two reasons why MaMPF can fit for various applications: 1) LBS not only takes the total packet length sequence into consideration, but also takes advantage of power-law distributions of different applications, which increase the discriminating power than other methods; 2) The features for one flow consist of the probabilities generated by all the application Markov models, which finally decides the classification results based on the relative probabilities of all applications.

In order to observe the overlaps of encrypted traffic classification from different applications, we give the classification matrices of SOB and MaMPF. In Figure 6(a), SOB mixes several applications, e.g. JD and Youdao. Most JD traffic flows are wrongly classified as Youdao traffic flows because of the overlaps of MTSs, certificate packet length and the first communication packet length. There are many coincident flows with these three kinds of information. Comparatively speaking, the classification matrix of MaMPF is more diagonalizable as shown in Figure 6(b), which is consistent with the excellent classification result.

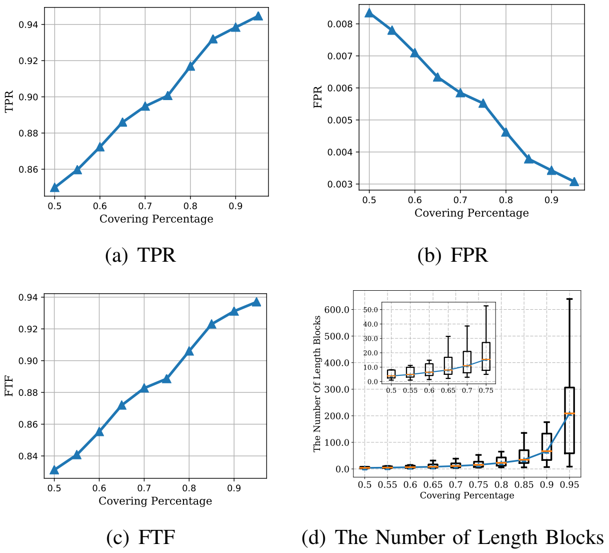


(a) SOB (b) MaMPF

Fig. 6. Classification Matrix Comparison of Different Methods. (The horizontal axis is the label of each application, and the longitudinal axis is the classification result.)

2) *Comparison on Variants of MaMPF:* SMPF only adopts MTSs to build Markov models whose outputs are used as fingerprints. And SLaveMPF considers MTSs and length average sequences to establish Markov models. The experiment results are also shown in Table III. On average, the FTF of MaMPF (0.9333) is better than that of SLaveMPF (0.8064) and SMPF (0.6725). From these experimental results, probability features from LBSs play important roles on encrypted traffic classification, and power-law division performs better than equal segment.

SMPF vs SLaveMPF: In particular, for five applications (Taobao, NeCmusic, Sogou, Youdao and iCloud), adding length average sequences enhances more than 40% TPR (Youdao increases even up to 85% TPR). The TPRs of other applications also increase, except JD, which is the only one of 18 applications that is not fit for SLaveMPF. For FPR criteria, SLaveMPF (0.97%) performs better than SMPF (1.69%). There are seven applications that SMPF gets better results (less than 0.5% improvement) than SLaveMPF, however, the TPRs of SLaveMPF grows 7% at least, to be specific, Apple (7% up), Github (25% up), iCloud (39% up), Taobao (49% up), Sogou (55% up), NeCMusic (77% up) and Youdao (85%



(a) TPR

(b) FPR

(c) FTF

(d) The Number of Length Blocks

Fig. 7. Tendencies of Traffic Covering Ratio

up). Except these seven applications, SLaveMPF decreases the FPRs of another 11 applications. Using length average sequences seems to be helpful for the overlaps of MTSs, and increases the diversity of fingerprints, which consequently enhances the results.

SLaveMPF vs MaMPF: Although SLaveMPF with length average sequences can improve TPR and FPR of almost all the applications, MaMPF with LBSs by power-law division can get a better result as Table III shows. MaMPF on 17 applications gets better performances on TPRs than SLaveMPF. Kaipanla seems to be identified with higher TPR with SLaveMPF (98.08%), but MaMPF can still get 96.85% TPR. Only considering SMPF, SLaveMPF and MaMPF methods, the FPRs of MaMPF on 11 applications are the best. For the other 7 applications, MaMPF achieves the acceptable results on FPR. These FPRs only have little differences with the best ones, and the corresponding TPRs can even improve much more than that of variants.

VII. Discussion

A. *Traffic Covering Percentage*

As we stated in Section IV-C, the amount of length values can be split into some length blocks because of the power-law distribution. The traffic covering percentage of representative length blocks affects the efficiency of the generated features for our MaMPF significantly. With more length blocks, the discriminating power of the length Markov probability vectors may be better, which leads to a better classification results. However, as the number of length blocks accumulates, LB-Transform matrix grows in space complexity of $O(N^2)$. And it is very sparse because only several length values occur in one application, which easily leads to the overfitting. Therefore, we test different traffic covering percentage, including 50%, 55%, 60%, 65%, 70%, 75%, 80%, 85%, 90% and 95%.

TABLE IV
Comparison Results Among Different Classifiers

ID	LinearSVM		LogicR		GBDT		RandomF	
	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR
1	0.829	0.002	0.804	0.002	0.822	**0.001**	**0.841**	**0.001**
2	0.835	**0.002**	0.835	**0.002**	0.894	**0.002**	**0.925**	**0.002**
3	0.936	0.010	0.936	0.010	0.963	**0.005**	**0.974**	**0.005**
4	0.960	0.008	0.961	0.009	0.967	0.007	**0.973**	**0.006**
5	0.769	0.003	0.769	0.002	0.853	**0.001**	**0.864**	**0.001**
6	**1.000**	**0.000**	**1.000**	**0.000**	**1.000**	**0.000**	**1.000**	**0.000**
7	0.963	**0.001**	0.962	**0.001**	0.958	**0.001**	**0.975**	**0.001**
8	0.852	0.007	0.861	0.007	**0.926**	**0.004**	0.924	**0.004**
9	0.973	0.001	0.969	0.001	0.958	0.001	**0.986**	**0.000**
10	0.877	0.001	0.883	0.001	0.953	0.000	0.**961**	**0.000**
11	0.913	**0.001**	0.905	**0.001**	0.974	**0.001**	**0.976**	**0.001**
12	0.989	**0.000**	0.988	**0.000**	**0.994**	**0.000**	0.992	**0.000**
13	0.945	0.015	0.949	0.016	0.957	0.008	**0.961**	**0.007**
14	0.876	0.001	0.865	**0.000**	0.908	**0.000**	**0.910**	**0.000**
15	0.766	**0.002**	0.763	**0.002**	0.800	**0.002**	**0.809**	**0.002**
16	0.884	0.003	0.894	**0.002**	0.937	0.003	**0.941**	**0.002**
17	0.936	0.003	0.938	0.003	0.979	0.005	**0.981**	**0.002**
18	0.886	**0.001**	0.883	**0.001**	0.942	**0.001**	**0.958**	**0.001**
AVE	0.939	0.003	0.940	0.003	0.958	**0.002**	**0.964**	**0.002**
FTF	0.932		0.933		0.953		**0.960**	

The results shown in Figure 7 is consistent with our observation. As the covering percentage grows from 50% to 95%, the TPR increases from around 84% to over 94% and the FPR decreases from more than 0.8% to about 0.3%, which makes the FTF rise from around 0.82 to nearly 0.94. Moreover, even if adopting 50% traffic covering percentage, the results are better than the state-of-the-art methods. However, the number of length blocks increases exponentially as shown in Figure 7(d). The largest number of length blocks is still less than 200 when the covering percentage is not over 90%, while it reaches more than 600 when the covering percentage arrives at 95%. Therefore, to find the balance of the performance and the number of length blocks is based on requirement (e.g. focusing on the memory overhead or classification accuracy). In this paper, we expect to get an acceptable result in our dataset with a reasonable memory overhead. Although the FTF can get a better classification result once traffic covering percentage is over 95% traffic, the amount of length blocks increases about 4 times. Therefore, we take 90% traffic covering percentage as a default option of our method MaMPF in this paper.

B. *Classifier Adaptation*

Once the probability features have been generated after normalization, the classifier could be trained to identify applications. There are many common classification algorithms which could be used in our system. Although the classifier is not the point we want to emphasize, we showed the validity of features we have generated for different classifiers. Due to the space limit, we only have a short discussion on LinearSVM, Logistic Regression (LogicR), Gradient Boosting Decision Tree (GBDT) and Random Forest (RandomF). The experiment results are shown in Table IV.

The four classifiers with our MaMPF all achieve satisfactory performances and outperform the state-of-the-art methods as shown in Table III, e.g. the lowest FTF is over

0.93. Different classifiers also cause a little difference on results. The performances of GBDT and RandomF (both over 95% FTF) are better than LinearSVM and LogicR, which may lie in the advantage of non-linear classifiers. From Table IV, the best results in our experiments belong to RandomF, with 96% FTF, which also get an obvious improvement in most application classification. The excellent performances of various classifiers on the real-world dataset certify our MaMPF are representative.

VIII. Conclusions

In this paper, we proposed the MaMPF for encrypted traffic classification which makes use of LBS from the power-law division of packet length sequence and the relative probability of all applications. In particular, both MTSs and LBSs are used to build Markov models for each application, and the occurrence probabilities with root normalization of all the applications are concatenated as the fingerprints. Experimental results reveal that MaMPF achieves a better performance compared to the state-of-the-art methods on the real-world datasets, and demonstrate the effectiveness of LBSs with power-law division. Moreover, MaMPF is robust for hyper parameter and classifier. Further researches include how to further improve MaMPF to fit more applications, identify more useful features on encrypted traffic with even higher discriminating power, and considering deep learning in solving this problem.

Acknowledgment

This work is supported by The National Key Research and Development Program of China (No.2016QY05X1000 and No.2016YFB0801200) and The National Natural Science Foundation of China (No.61602472). Research is also supported by the CAS/SAFEA International Partnership Program for Creative Research Teams and IIE, CAS international cooperation project. Zigang Cao is the corresponding author.

References

[1] F. Constantinou and P. Mavrommatis, "Identifying known and unknown peer-to-peer traffic," in *IEEE International Symposium on Network Computing and Applications*, 2006, pp. 93–102.

[2] S. H. Yoon, J. W. Park, J. S. Park, Y. S. Oh, and M. S. Kim, "Internet application traffic classification using fixed ip-port," in *Asia-Pacific Network Operations and Management Conference on Management Enabling the Future Internet for Changing Business and New Computing Services*, 2009, pp. 21–30.

[3] Q. Zhang, Y. Ma, J. Wang, and X. Li, "Udp traffic classification using most distinguished port," in *Asia-Pacific Network Operations and Management Symposium*, 2014, pp. 1–4.

[4] P. Zejdl, S. Ubik, V. Macek, and A. Oslebo, "Traffic classification for portable applications with hardware support," in *International Workshop on Intelligent Solutions in Embedded Systems*, 2008, pp. 1–9.

[5] F. Risso, M. Baldi, O. Morandi, and A. Baldini, "Lightweight, payload-based traffic classification: An experimental evaluation," in *IEEE International Conference on Communications*, 2008, pp. 5869–5875.

[6] Y.-H. Goo, K.-S. Shim, S.-K. Lee, and M.-S. Kim, "Payload signature structure for accurate application traffic classification," in *Asia-Pacific Network Operations and Management Symposium*, 2016, pp. 1–4.

[7] J.-S. Park, S.-H. Yoon, and M.-S. Kim, "Performance improvement of payload signature-based traffic classification system using application traffic temporal locality," in *Asia-Pacific Network Operations and Management Symposium*, 2013, pp. 1–6.

[8] B. Anderson, S. Paul, and D. McGrew, "Deciphering malware's use of tls (without decryption)," *arXiv preprint arXiv:1607.01639*, 2016.

[9] B. Anderson and D. Mcgrew, "Identifying encrypted malware traffic with contextual flow data," in *ACM Workshop on Artificial Intelligence and Security*, 2016, pp. 35–46.

[10] B. Anderson and D. McGrew, "Machine learning for encrypted malware traffic classification: accounting for noisy labels and non-stationarity," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 1723–1732.

[11] A. Dainotti, A. Pescape, and K. C. Claffy, "Issues and future directions in traffic classification," *IEEE network*, vol. 26, no. 1, 2012.

[12] H. Xiong, H. Xiong, H. Xiong, H. Xiong, H. Xiong, and H. Xiong, "Effective and real-time in-app activity analysis in encrypted internet traffic streams," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 335–344.

[13] M. Korczynski and A. Duda, "Markov chain fingerprinting to classify encrypted traffic," in *IEEE Conference on Computer Communications*, 2013, pp. 781–789.

[14] M. Shen, M. Wei, L. Zhu, M. Wang, and F. Li, "Certificate-aware encrypted traffic classification using second-order markov chain," in *IEEE/ACM International Symposium on Quality of Service*, 2016, pp. 1–10.

[15] M. Shen, M. Wei, L. Zhu, and M. Wang, "Classification of encrypted traffic with second-order markov chains and application attribute bigrams," *IEEE Transactions on Information Forensics & Security*, vol. PP, no. 99, pp. 1–1, 2017.

[16] A. Freier, P. Karlton, and P. Kocher, "The secure sockets layer (ssl) protocol version 3.0," 2011.

[17] T. Dierks, "The transport layer security (tls) protocol version 1.2," 2008.

[18] A. W. Moore and K. Papagiannaki, "Toward the accurate identification of network applications," in *International Conference on Passive and Active Network Measurement*, 2005, pp. 41–54.

[19] S. Sen, O. Spatscheck, and D. Wang, "Accurate, scalable in-network identification of p2p traffic using application signatures," in *International Conference on World Wide Web*, 2004, pp. 512–521.

[20] M. Roughan, S. Sen, O. Spatscheck, and N. Duffield, "Class-of-service mapping for qos: a statistical signature-based approach to ip traffic classification," in *ACM SIGCOMM Conference on Internet Measurement*, 2004, pp. 135–148.

[21] P. Velan, "A survey of methods for encrypted traffic classification and analysis," *Networks*, vol. 25, no. 5, pp. 355–374, 2015.

[22] S. Hao, J. Hu, S. Liu, T. Song, J. Guo, and S. Liu, "Improved svm method for internet traffic classification based on feature weight learning," in *International Conference on Control, Automation and Information Sciences*, 2015, pp. 102–106.

[23] J. Zhang, C. Chen, Y. Xiang, W. Zhou, and Y. Xiang, "Internet traffic classification by aggregating correlated naive bayes predictions," *IEEE Transactions on Information Forensics & Security*, vol. 8, no. 1, pp. 5–15, 2013.

[24] C. Wang, T. Xu, and X. Qin, "Network traffic classification with improved random forest," in *International Conference on Computational Intelligence and Security*, 2016, pp. 78–81.

[25] M. Conti, L. V. Mancini, R. Spolaor, and N. V. Verde, "Analyzing android encrypted network traffic to identify user actions," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 1, pp. 114–125, 2016.

[26] W. Pan, G. Cheng, and Y. Tang, "Wenc: Https encrypted traffic classification using weighted ensemble learning and markov chain," in *IEEE Trustcom/BigDataSE/ICESS*, 2017, pp. 50–57.

[27] W. M. Shbair, T. Cholez, J. Francois, and I. Chrisment, "Improving sni-based https security monitoring," in *IEEE International Conference on Distributed Computing Systems Workshops*, 2016, pp. 72–77.

[28] W. M. Shbair, T. Cholez, A. Goichot, and I. Chrisment, "Efficiently bypassing sni-based https filtering," in *IFIP/IEEE International Symposium on Integrated Network Management*, 2015, pp. 990–995.

[29] P. T. Endo and D. F. H. Sadok, "Whois based geolocation: A strategy to geolocate internet hosts," in *IEEE International Conference on Advanced Information Networking and Applications*, 2010, pp. 408–413.

[30] L. A. Adamic, B. A. Huberman, A. L. Barabsi, R. Albert, H. Jeong, and G. Bianconi, "Power-law distribution of the world wide web," *Science*, vol. 287, no. 5461, p. 2115, 2000.