Identifying DApps and User Behaviors on Ethereum via Encrypted Traffic

Yu Wang[1,2], Zhenzhen Li[1,2], Gaopeng Gou[1,2], Gang Xiong[1,2(✉)], Chencheng Wang[1,2], and Zhen Li[1,2]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{wangyu1996,lizhenzhen,gougaopeng,xionggang,wangchencheng,lizhen}@iie.ac.cn

[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

Abstract. With the surge in popularity of blockchain, more and more Decentralized Applications (DApps) are deployed on blockchain platforms. DApps bring convenience to people, but cause security and efficiency problems. In this paper, we focus on the security and efficiency problems of DApps on Ethereum. Our research is divided into three application scenarios. In DApps classification, we analyze characteristics of DApps and extract efficient features to recognize 11 representative DApps. In DApps user behaviors classification, we propose behavior-sensitive features and improved time features to recognize 88 DApps user behaviors, which would help to identify malicious behaviors in encrypted traffic. In general user behavior classification, different categories of features are proposed to recognize 15 general user behaviors which represent the performance of DApps. DApps developers can obtain valuable data to improve the quality of service through analyzing the classification results. Experimental results in the three application scenarios achieve excellent performance (99.5% accuracy for DApps classification, 95.65% accuracy for DApps user behaviors classification, 98.58% accuracy for general user behaviors classification) and outperform the state-of-the-art methods.

Keywords: DApps and user behaviors · Encrypted traffic classification · Features extraction · Traffic analysis · Machine learning

1 Introduction

Ethereum [4] is the first major blockchain to support the Turing-complete scripting via smart contracts, which allow parties to create virtual trusted third parties that behave according to arbitrary agreed-upon rules. It attracts people to write smart contracts for building Decentralized Applications (DApps), which run on a peer-to-peer network of computers. DApps become one of the development trends of the internet. As of May 2020, there are more than three thousand DApps, and 82% of them are deployed on Ethereum [1].

N. Park et al. (Eds.): SecureComm 2020, LNICST 336, pp. 62–83, 2020.
https://doi.org/10.1007/978-3-030-63095-9_4

DApps deployed on Ethereum use SSL/TLS protocols to encrypt transmission data between entities. There is no difference in encrypted protocol details between different DApps, because these DApps are adopted on the same blockchain platform, so the dissimilarity of encrypted traffic generated by different DApps is reduced. Some traditional classification methods begin to lose effect, such as deep packet inspection [11]. There are lots of research on application identification [3,5,18,21], with few articles mentioned DApps. In [20], the authors use high-dimensional features for DApps classification, which results in low efficiency. So in the case of ensuring high accuracy, how to improve the efficiency of DApps classification is a challenge.

DApps are similar to traditional apps, with privacy and security risks [7,22]. As for the classification on DApps user behaviors, it can identify specific user behaviors (e.g., checking account, communication, purchase, comment) in encrypted traffic, which may help network operators detect suspicious behaviors, thereby enhancing the protection of user privacy. For example, the attacker recruits employees through posts in Ethlance, which are actually phishing posts. This classification may help to identify the dubious behavior which is repeated many times by the same IP in a short period of time. Additionally, some DApps containing Trojan codes disguise as normal DApps, which steal confidential data of unwitting users, and the classification may help to identify suspicious behaviors other than normal ones. The operator could then take appropriate actions to protect user privacy. Researchers have been studying user behaviors classification for many years, but most of them focus on traditional applications [7,8,24]. Due to the invisibility and confusion of DApps user behaviors traffic, some features do not perform well, such as packet sizes [17], packet flags [13,19], statistical features of packet length [21]. Statistical-time features are used to identify Bitcoin wallets and user actions [2], but these statistics have little improvement on DApps user behaviors classification. Therefore, we need to propose a specific set of features according to the characteristics of DApps user behaviors.

Improving Qos is a topic of ongoing interest in many aspects, such as applications [12,15], distributed multimedia services [10], cloud services [16]. As an emerging technology, most DApps cannot provide good quality of user experience. DApps run on a decentralized network and each node can be regarded as a central server, which bring challenges to improve Qos of DApps. As for the classification on general user behaviors, it ignores the differences between DApps. Combined with traffic data, it can provide valuable information (e.g., user preferences, throughput, latency) within an organization. The data is invaluable for network administrators to optimize the networks. For example, the administrators can trigger the automated re-allocation of network resources for priority behaviors after getting user preferences through the usage frequency of general user behaviors. They can obtain the latency of each behavior by extracting time series from the traffic data. Administrators could configure their networks to help DApps perform more efficiently, thereby improving Qos of DApps.

In this paper, we focus on the classification for three application scenarios of DApps encrypted traffic. After analyzing the characteristics of DApps and

user behaviors, we propose different sets of features for three scenarios to get better performance. Then, we use three classification models to conduct the classification and compare their performance. In addition, we find that most flows of DApps traffic are short flows through experiment. The contributions of this article are as follows:

- We classify encrypted DApps traffic. After analyzing DApps characteristics and network traffic, we use less features to improve the efficiency of DApps classification. The effectiveness of the proposed features is verified by experimenting on encrypted traffic collected from 11 representative DApps. The experimental results are better than the state-of-the-art methods.
- We explore the fine-grained classification on encrypted DApps traffic and research on DApps user behaviors. After analyzing DApps, 88 available user behaviors are totally extracted. We propose DApps user behaviors-sensitive features and the improved time features, which strengthen the discrimination of DApps user behaviors in encrypted traffic. The experimental results demonstrate that our method can identify DApps user behaviors with up to 95% accuracy. To the best of our knowledge, we are the first one to classify user behaviors of multiple types of DApps.
- We categorize 88 user behaviors into 15 general user behaviors which represent the performance of DApps. Different kinds of features are proposed according to the characteristics of general user behaviors. Compared with the existing methods, the performance of our proposed method is preferable to them, with 98% accuracy. This work is the first to perform classification on general user behaviors.

The rest of this paper is organized as follows. Section 2 briefly reviews related work. Next, we elaborate the process of extracting features for different application scenarios in Sect. 3. Section 4 details the dataset and shows the experimental results. Finally, we conclude this paper in Sect. 5.

2 Related Work

Researches on traffic classification emerge in an endless stream. According to our research, we provide an overview of related work in three aspects: encrypted application traffic classification, application user behavior classification and analysis of network traffic of blockchain.

Encrypted Application Traffic Classification. In [6], Chen et al. found that despite encryption, web applications also suffered from side-channel leakages. They leveraged fundamental features of web applications: stateful communication, low entropy input and significant traffic distinction. But web applications consist of browser-side and server-side components, DApps can be split into browser-side and smart contracts. Cai et al. presented a website fingerprinting attack and proved its effectiveness through traffic analysis countermeasures [5]. They used an SVM with a custom kernel based on an edit-distance. The edit distance allowed for delete and transpose operations, that are supposed to capture

drop and re-transmission of packets respectively. Shen et al. incorporated the certificate packet length clustering into the second-order homogeneous Markov chains [18]. The work could lead to a 29% improvement on average compared with existing Markov-based methods, in terms of classification accuracy. In [17], the authors presented a website fingerprinting method at Internet Scale. The accumulated sum of packet sizes was used to represent the progress of webpage loading. Gil et al. extracted time related features such as flow bytes per second, inbound and outbound inter-arrival time to characterize the network traffic by using C4.5 and KNN [9]. They detected six major classes of VPN traffic including browsing, streaming, chat, email, file transfer and VoIP. Alan et al. found that popular Android apps can be identified with 88% accuracy, only using packet sizes of the first 64 packets [3]. Vincent et al. used 54 statistical features of packet length on Random Forest to build an Appscanner [21] which can identify 110 applications with 96% accuracy. They also proved that app fingerprints persist in varying extents across devices and app versions.

Application User Behaviors Classification. Coull et al. utilized the size of exchanged packet between the target user and Apple's server to identify iMessage user actions, such as start writing, stop writing, message sending, attachment sending [8]. In [24], the authors used a suite of inference techniques to reveal a specific user action (i.e., send a tweet) on the Twitter app installed on an Android smartphone. Condi et al. clustered the streams of each application user behavior by clustering methods [7], then they calculated the dynamic warping distance for each stream, in terms of packet length. But they experimented on each app and ignored the similarity of encrypted traffic between different apps. Yan et al. segmented WeChat traffic into several bursts to describe different actions [23], and extracted packet length, number of TCP handshakes, statistics from each burst to identify red packet transactions and fund transfers.

Analysis of Network Traffic of Blockchain. In 2014, Koshy et al. developed heuristics to apply highly conservative constraints to Bitcoin network traffic [14], and they found that nearly 1,000 Bitcoin addresses can be mapped to IPs by leveraging anomalous relaying behaviors. Shen et al. generated high-dimensional features by fusing time series, packet length and packet burst [20]. The accuracy of DApps traffic classification reached 90%. But the training time and testing time of this method is much longer than other methods because of the large input vector, which results in low efficiency. Aiolli et al. identified bitcoin wallet apps and user actions only through statistical-time features, such as length, maximum, minimum, mean, etc. [2]. The classifier was trained by SVM and Random Forest algorithm, and the accuracy achieved 95%.

Due to the same encrypted protocol and blockchain platform, how to effectively and accurately classify DApps and user behaviors in encrypted traffic are challenging. In this paper, we propose suitable features for three application scenarios, and explain them with DApps characteristics.

3 Methodology

In this section, we introduce the features we extracted for three application scenarios: DApps classification, DApps user behaviors classification and general user behaviors classification. The main process of our model is shown in Fig. 1. At first, we collect DApps network traffic and pre-process DApps traffic. After analyzing characteristics of different application scenarios, we propose different features. Finally, we apply different sets of features to the existing machine learning algorithms for different application scenarios classification.

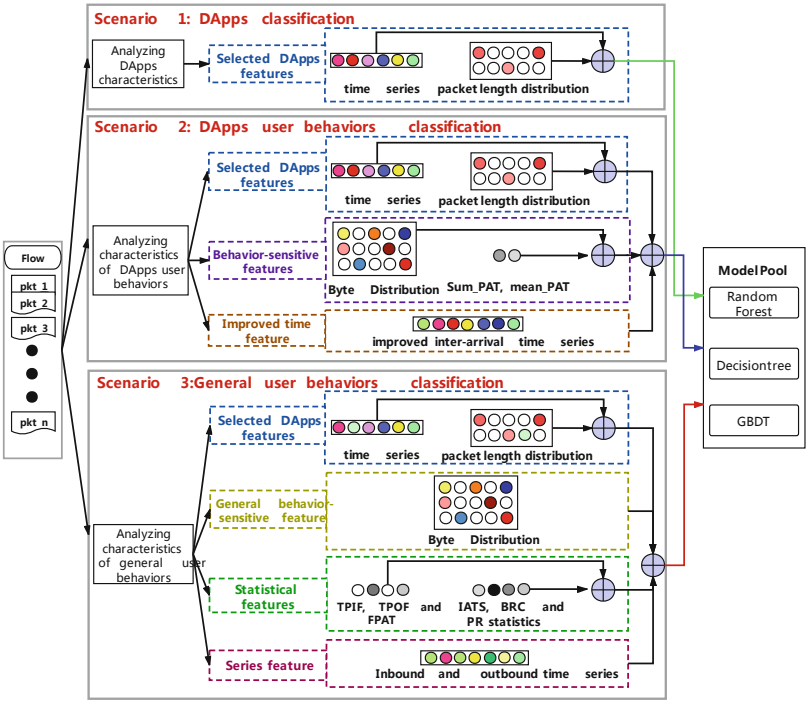

Fig. 1. Main process of Modeling. Sum_PAT, mean_PAT represent sum of packet arrival time, mean of packet arrival time. IATS, BRC and PR statistics represent statistical features of inter-arrival time series, byte rate change and packet rate. TPIF, TPOF and FPAT represent total packets of inbound flow, total packets of outbound flow and the first packet arrival time.

3.1 Feature Extraction for DApps Classification

There are many different characteristics between DApps and traditional applications, such as operation mode (e.g., decentralized), data storage (e.g., Ethereum platform, IPFS), construction (e.g., web applications consist of browser-side and server-side components, DApps consist of browser-side and smart contracts),

etc., which result in the differences of packet arrival time and packet length for different DApps.

Selected DApps Features
Selected DApps features consist of packet length distribution and time series. The features are detailed as follows.

Packet Length Distribution. The size of the payloads for the first 100 packets of a session are recorded in dataset. We assume a 1500 byte Maximum Transmission Unit, and create 150 bins of 10 bytes each. Then, the number of packets whose length is in the range [0,10) is taken as the value of the first bin, the number of packets whose length is in the range [10,20) is taken as the value of the second bin, and so on. We get the number of packets that fell into different intervals in a flow. Finally, we construct a length-150 array.

Time Series. The packet arrival time for the first 100 packets of a session are recorded. we construct a length-100 array.

3.2 Feature Extraction for DApps User Behaviors Classification

Compared with DApps classification, DApps user behaviors classification is more fine-grained. We divide proposed features into three categories: selected DApps features, behavior-sensitive features and improved inter-arrival time series.

Selected DApps Features
We use the two features which are mentioned in Sect. 3.1: packet length distribution and time series.

Behavior-Sensitive Features
Behavior-sensitive features consist of sum of packet arrival time, mean of packet arrival time, and byte distribution. The features are detailed as follows.

Sum of Packet Arrival Time and Mean of Packet Arrival Time. For some behaviors, such as creating project, users can submit applications to rent things in Staybit. If other users want to rent, they need to pay through Ethereum accounts. This process causes distinction of transmission data, so sum of packets arrival time may be different. Each behavior has different sequence of actions. For example, behavior 'like the artwork' needs five actions in a precise order, behavior 'comment' needs seven actions, behavior 'open Superrare' needs three actions. An action could be simple (e.g., a click on a button, a selection of edit box) or complex (e.g., a connection of Ethereum wallet, type a text, which is randomly selected from a set of sentences). So the number of packets and response time of each behavior are different. We propose two features: sum of packet arrival time and mean of packet arrival time.

Byte Distribution. DApps utilize encrypted protocol to encrypt transmission data between entities. For different DApps user behaviors, DApps need to call different part of smart contracts, and the action sequence of each behavior is also unique. Although the communication data is encrypted, the encrypted payload is subtle distinction due to the difference of the original payload. So we extract byte distribution which is a length-256 array that keeps a count for each value encountered in the payloads of the packets for each packet in the flow. We use Principal Components Analysis to reduce the 256-dimensional data to 2-dimensional data. Each behavior is represented by two-dimensional data in the figure.

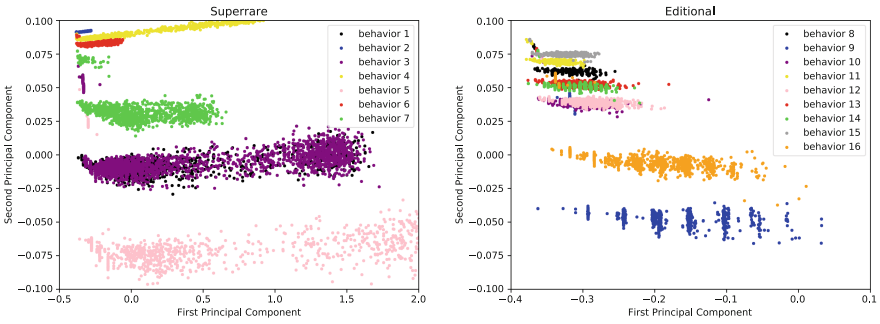

Fig. 2. Byte Distribution of DApps user behaviors. Each color represents a user behavior. Behavior 1–16 refer to the ID in Table 5

We only show the results of 16 DApps user behaviors in Fig. 2 because of space limitation. There are some DApps user behaviors overlap, such as 'open Superrare' and 'view details of the artwork' in Superrare, probably because the action sequences and the required data resources of the two behaviors are similar. However, it can be seen from the figure that the byte distribution is obviously effective in distinguishing DApps user behaviors.

The Improved Inter-arrival Time Series

Users use DApps through web user interface which interacts with wallets (e.g., Metamask, TrustWallet, Imtoken) for some user behaviors (e.g., comment on a post, like or dislike an artwork, etc., but not all user behaviors need this step), then it interacts with blockchain nodes rather than central servers. User behaviors that must connect to wallets are marked with ☆ in Table 5. The back-end code (smart contract) of DApps runs on the nodes of decentralized peer-to-peer network. Some DApps may store data as the metadata of transactions on Ethereum. Some DApps may build separate storage system on IPFS, but the developer needs to consider some complex things while building it, such as access management system. Therefore, the response time from Ethereum nodes to the front-end is different for different user behaviors. we improve inter-arrival time series as follows. We convert the original inter-arrival time

series $PTS = (s_1, s_2, \ldots, s_j, \ldots, s_{n-1})$ to the improved inter-arrival time series $G\_PTS = (a_1, a_2, \ldots, a_j, \ldots, a_{n-1})$, where $s_j = t_{j+1} - t_j$. The a_j is computed by:

$$a_j = \Delta t \cdot b + \frac{\Delta t}{2} \tag{1}$$

where b satisfies the condition, $\Delta t \cdot b \leqslant s_j < \Delta t(b+1)$, we select $\Delta t = 0.005\,\mathrm{s}$ after testing.

Figure 3 reports the statistical distribution of the improved inter-arrival time of bidirectional flows for each user behavior. The first quartile, the median and the third quartile are highlighted by using a notched box plot. Some behaviors have a long tail distribution for the improved inter-arrival time such as behavior 33, 34, 46, 50. User behaviors in the same DApps (e.g., Thomas Crown Art, Latium) are very similar, although they are different behaviors. Behavior 7, 35, 60, 65 show a very long inter-arrival time distribution. From the figure we can see that the improved inter-arrival time may be a discriminative feature.

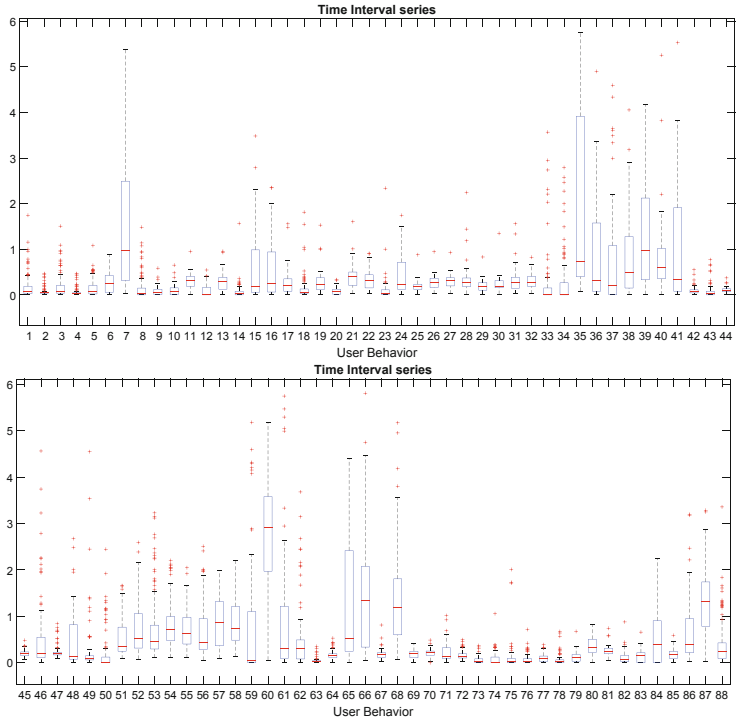

Fig. 3. Statistical distribution of the improved inter-arrival time of bidirectional flows for each user behavior. The median is represented as the red line. First quartile and third quartile are represented as the down and up side of the box. The + represents improved inter-arrival time beyond the first quartile and third quartile. The X-axis label user behavior 1–88 refer to the ID in Table 5 (Color figure online)

3.3 Feature Extraction for General User Behaviors Classification

For this application scenario, we select 15 general user behaviors which represent the performance of these DApps. Each general user behavior consists of the same type of user behavior of different DApps. For example, all DApps exist behavior 'opening DApps' and we group them into a general user behavior. Suitable features are proposed for this scenario, and they are divided into four kinds: selected DApps features, general behavior-sensitive features (behavior-sensitive features without features related to DApps), series features, and statistical features.

Selected DApps Features
We use the two features which are mentioned in Sect. 3.1: packet length distribution and time series.

General Behavior-Sensitive Features
We also use the behavior-sensitive features which are mentioned in Sect. 3.2. But we remove features related to DApps to improve efficiency. So we only select byte distribution.

Statistical Features
Statistical features consist of total packets of inbound flow, total packets of outbound flow, the first packet arrival time of outbound flow, statistical features of inter-arrival time series, byte rate change and packet rate.

Total Packets of Inbound Flow, Total Packets of Outbound Flow, and the First Packet Arrival Time of Outbound Flow. Action sequences of general user behaviors of different DApps are similar, but different general user behaviors are composed of different action sequences. The required data are retrieved from Ethereum peers, then they are transmitted to the front-end through API and displayed on the interface. We consider that these proposed features may be different because of different action sequences.

Statistical Features of Inter-arrival Time Series, Byte Rate Change and Packet Rate. The ten statistical features of inter-arrival time series are the mean, standard deviation, maximum, minimum, length of unique number, mode, frequency of mode, percentile. In particular, we choose 0.25, 0.5, 0.75 percentile of inter-arrival time series. These statistics are also collected for byte rate change and packet rate.

Since there is no central server in DApps, the impact of different DApps on general user behaviors classification is reduced. Many DApps store data on Ethereum, different general user behaviors may result in different transmission rates at different stages. Bytes rate change, $BRC = (\mathrm{brc}_1, \mathrm{brc}_2, \mathrm{brc}_3, \ldots, \mathrm{brc}_{n-1})$, calculated through the bytes rate sequence $BRS = (\mathrm{brs}_1, \mathrm{brs}_2, \mathrm{brs}_3, \ldots, \mathrm{brs}_{n-1})$. brc_i represents the D-value between $brs_{i+1} - brs_i$. brs_i represents rate of bytes in Δt time. We try Δt from 0 to 3 s, and we find the result is the best when $\Delta t = 0.25$ s. However, we do not use this feature directly. Skewness coefficient is a feature that describes the symmetry of data distribution. The skewness brc^s is calculated as follows:

$$brc^s = \frac{\frac{1}{n-1}\sum_{i=1}^{n-1}\left(brc_i - \bar{brc}\right)^3}{\left(\frac{1}{n-2}\sum_{i=1}^{n-1}\left(brc_i - \bar{brc}\right)^2\right)^{\frac{3}{2}}} \tag{2}$$

Besides, kurtosis is a descriptor of the shape of a probability distribution. The kurtosis brc^k is calculated as follows:

$$brc^k = \frac{\frac{1}{n-1}\sum_{i=1}^{n-1}\left(brc_i - \bar{brc}\right)^4}{\left(\frac{1}{n-1}\sum_{i=1}^{n-1}\left(brc_i - \bar{brc}\right)^2\right)^2} - 3 \tag{3}$$

Similar to brc^s and brc^k, for packet rate $PR = \left(\mathrm{pr}_1, \mathrm{pr}_2, \mathrm{pr}_3, \ldots, \mathrm{pr}_{n-1}\right)$. The skewness pr^s and the kurtosis pr^k are computed by:

$$pr^s = \frac{\frac{1}{n}\sum_{i=1}^{n}\left(pr_i - \bar{pr}\right)^3}{\left(\frac{1}{n-1}\sum_{i=1}^{n-1}\left(pr_i - \bar{pr}\right)^2\right)^{\frac{3}{2}}}; pr^k = \frac{\frac{1}{n}\sum_{i=1}^{n}\left(pr_i - \bar{pr}\right)^4}{\left(\frac{1}{n}\sum_{i=1}^{n}\left(pr_i - \bar{pr}\right)^2\right)^2} - 3 \tag{4}$$

Series Features

Inbound and Outbound Arrival Time Series. In order to enhance the impact of the action sequence on classifier, we consider that packet arrival time series play an important role, and the inbound and outbound packet arrival time series have different characteristics. Therefore, we extract not only bidirectional arrival time series, but also inbound and outbound arrival time series.

4 Performance Evaluation

In this section, we first describe how we collect the labeled DApps traffic. Then, in order to evaluate the performance of proposed features for three application scenarios classification, we utilize Precision, Recall, F1-measure and Accuracy. Ten-fold cross validation is used to evaluate our method. We introduce experimental results from the following subsections: results of different models, evaluation of DApps classification, evaluation of DApps user behaviors classification, evaluation of general user behaviors classification and proposed features analysis.

4.1 Dataset

Figure 4 depicts our traffic collection platform. Users use DApps through virtual machines that are connected to the same access point, and all captured files are transferred to the point for traffic pre-processing and experiments. In order to assess our proposed features, we select 11 representative DApps of diverse categories on Ethereum, most of which have a lot of users, and all of them are close to our lives. After analyzing each DApp, we extract 88 available user

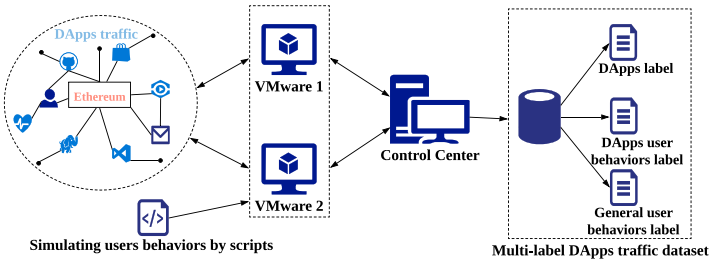

Fig. 4. Process of DApps traffic capture.

behaviors in total. Then, we categorize 88 user behaviors into 15 general user behaviors which represent the performance of these DApps. It is noteworthy that different sets of features proposed for these application scenarios are applicable to all DApps, the 11 representative DApps are used to evaluate our methods.

Table 1. Flow of 11 DApps

DApps	Category	Flow	Percentage (%)
Superrare	Marketplace	17593	9.1
Editional	Social	11066	5.8
John Orion Young	Property	13937	7.2
Thomas Crown Art	Marketplace	12105	6.3
Cryptoboiler	Social	15147	7.9
Ethlance	Social	12349	6.4
Knownorigin	Marketplace	15200	7.9
Staybit	Property	15535	8.1
Crowdholding	Social	24500	12.7
Latium	Exchanges	30245	15.8
Viewly	Media	24696	12.8
Total		**192373**	**100**

In order to achieve a particular target, a user must perform several actions in a precise order, which is the same as the action sequence in the real world. For example, when we comment on a post in Cryptoboiler, we need to perform exactly seven actions before we comment success: 1) open browser 2) enter Cryptoboiler 3) slide window 4) select edit box 5) fill the box with some text, which is randomly selected from a set of sentences 6) click publish button 7) close browser. We write 88 automatic scripts with Microsoft Visual Basic Script Edition, and each script represents a user behavior. Each user behavior is conducted for 200 times. For some user behaviors, we utilize different content to enrich the

data set (e.g., for behavior 'select artwork' in Superrare, there is an action in the action sequence to view the detailed information of an artwork, we totally view the detailed information of 40 artworks, five times for each artwork, instead of 200 times for an artwork). The scripts are used to simulate user behaviors within DApps, and the traffic generated by DApps was captured.

For some user behaviors (e.g., comment post in Cryptoboiler, agree article in Crowdholding, like article in Editional), we need two different categories of users. "Passive" user is used to passively use the DApps, by receiving posts or comments. "Active" user simulates the behavior of users that actively use DApps by sending posts, comment, giving "Passive" user a like, etc. The main purpose of this step is to protect the normal use of other users.

Table 2. Flow of 15 general user behaviors (Color figure online)

Behaviors	Flow	Percentage (%)
Open DApps	28426	15
Open market	17176	9
View detail	29502	15
Follow a user	4685	2
Like or dislike	10885	6
Create project	16641	9
Search	22872	12
View homepage	17825	9
Activities	6176	3
Add to cart	2871	1
Watch video	2882	1
Comment	4078	2
DApps introduction	4748	3
Refresh cart	2977	2
Else behaviors	20629	11
Total	**192373**	**100**

We collected 192,373 flows and millions of packets on two virtual machines from August 4, 2019 to September 30, 2019. As seen in Fig. 4, we get the multi-label dataset. For each flow, it has three labels, including DApps label, DApps user behaviors label, and general user behaviors label. By simulating user behaviors through scripts, it is possible to label the flows. Their detailed information is showed in Table 1, Sect. 4.4, and Table 2, respectively. Finally, we extracted data from Wireshark capture files to form the Json file. The data includes time, IP addresses, packet lengths, payload, TCP/IP flags, extension information, etc.

4.2 Results of Different Models

The evaluated classifiers include Random Forest(RF), Gradient Boosting Decision Tree(GBDT), Decision Tree(DT), and the experimental results are shown in Table 3. The Random Forest classifier achieves the best performance for three application scenarios. So we select the Random Forest classifier for the following experiments. The classifier is trained using 100 estimators (the tree number of Random forest).

Table 3. Accuracy of different scenarios with different machine learning algorithms

Accuracy	GBDT	DT	RF
DApps classification	0.9032	0.9884	**0.995**
DApps user behaviors classification	0.8472	0.9483	**0.9565**
General user behaviors classification	0.6878	0.9581	**0.9858**

4.3 Evaluation of DApps Classification

As for the DApps traffic classification, we compare our proposed features with two other methods to classify 11 DApps, which are summarized as follows:

– Appscanner [21], which uses the statistical features of packet length (e.g., mean, percentiles) about incoming, outgoing and bi-directional flows in the RF classifier.

Table 4. Comparison of Appscanner, FFP and our features

DApps	Appscanner			FFP			Our features		
	P	R	F1	P	R	F1	P	R	F1
Superrare	0.9809	0.9039	0.9408	0.9870	0.9903	0.9885	0.9989	0.9989	0.9989
Editional	0.9857	0.9972	0.9915	0.9887	0.9866	0.9876	0.9865	0.9813	0.9838
John Orion Young	0.9914	0.9957	0.9936	0.9906	0.9750	0.9825	0.9958	0.9905	0.9931
Thomas Crown Art	0.9993	0.9921	0.9957	0.9898	0.9915	0.9906	0.9947	0.9969	0.9958
Cryptoboiler	0.9936	0.9984	0.9959	0.9967	0.9956	0.9961	0.9985	0.9980	0.9982
Ethlance	0.9987	0.9928	0.9957	0.9917	0.9892	0.9904	0.9944	0.9935	0.9940
Knownorigin	0.9917	0.9926	0.9922	0.9937	0.9976	0.9956	0.9972	0.9992	0.9982
Staybit	0.9999	0.9993	0.9997	0.9983	0.9979	0.9981	0.9999	0.9988	0.9993
Crowdholding	0.9996	0.9988	0.9992	0.9973	0.9977	0.9975	0.9988	0.9980	0.9984
Latium	0.9912	0.9923	0.9917	0.9886	0.9909	0.9898	0.9918	0.9927	0.9922
Viewly	0.9254	0.9744	0.9492	0.9757	0.9778	0.9766	0.9905	0.9944	0.9925
Average	0.9870	0.9852	0.9859	0.9907	0.9885	0.9902	0.9951	0.9947	0.9949
Accuracy	0.9844			0.9901			0.9950		

*P and R in this table means the metric Precision and Recall.

– FFP [20], which uses three sequence features: time series, packet length and burst. Then these features are fused through a kernel function to become high-dimensional features, which are used in the RF classifier.

As seen in Table 4, all of the methods perform good, the accuracy of the worst classifier reaches 98%. The accuracy of our method is 99.50%, which is higher 1.06% than Appscanner. And our features outperform FFP with about 0.5% improvement. The performance gap between our method and FFP is not large. However, the vector input into the classifier of FFP is much larger than the classifier of our method, so FFP takes longer time to build a model. Our model uses less features to improve efficiency and get better results.

4.4 Evaluation of DApps User Behaviors Classification

To implement our proposed features are better, we compare our approach with AppScanner (AppS) [21] and Aiolli [2] to classify 88 DApps user behaviors. Random forest classifier is used in comparison experiments. Aiolli method is summarized as follows:

– Aiolli, which uses statistical-time features (e.g., mean, median, mode) about incoming, outgoing and bi-directional flows in the RF classifier.

According to Table 5, the precision of our method is more than 98% for most DApps user behaviors. But we can see that the accuracy of behavior 1 (open Superrare) and behavior 3 (select artworks) is really poor according to Table 5. Since the result of DApps classification reaches 99%, the poor results are not caused by the similarity of DApps. Our classifier may confuse the two behaviors, which have similar action sequences. In Fig. 2 and Fig. 3, the distributions of the two behaviors are close to each other, which reflect the distinctiveness of the two features (i.e., byte distribution, the packet inter-arrival time series) in DApps user behaviors classification. The accuracy of the classification is above 95%, so our proposed features can effectively distinguish DApps user behaviors.

The comparison results are shown in Table 5. Our approach achieves the best performance, and outperforms the other methods. The accuracy of the proposed method achieves about 95.66% and the F1 score achieves about 95.53%. Compared to AppScanner and Aiolli, the accuracy of our features increases by 27.58% and 48.06%, and the F1 score of our features increases by 28.89% and 49.28%. The improvement effect is very obvious. Their approaches are prone to misclassification on DApps user behaviors. We think our proposed features (e.g., behavior-sensitive features: byte distribution, the improved inter-arrival time series features) can extract more information and details so that our approach is far more effective than others.

Table 5. Description of user behaviors and classification results with different methods. AppS, AF, OF represent AppScanner, Aiolli features, Our features, respectively.

DApps	ID	User behavior	Description	AppS	AF	OF	Flow
				F1	F1	F1	
Superrare	1	open superrare	open the Superrare	0.46	0.11	0.54	2819
	2	open market	browse artworks on market page	0.69	0.24	1.00	4268
	3	select artwork	view details of an artwork	0.48	0.19	0.56	3018
	4	☆like artwork	like or dislike an artwork	0.79	0.01	1.00	2841
	5	user homepage	browse one's homepage	0.79	0.68	0.99	1294
	6	view activities	view activities happened in DApp	0.79	0.69	1.00	1498
	7	search	search artworks or artists	0.91	0.63	0.98	1855
	Average			0.70	0.36	0.87	2513
Editional	8	open editional	open the Editional	0.73	0.68	1.00	1419
	9	learn DApp	look Editional introduction	0.82	0.53	0.99	1220
	10	select collectible	view details of a collectible	0.85	0.73	0.98	1238
	11	artist homepage	browse a user homepage	0.75	0.31	0.99	1216
	12	artist create	look collectibles created by artist	0.86	0.46	0.98	1207
	13	artist collect	look collectibles collected by artist	0.80	0.18	0.98	1072
	14	view support page	view the support homepage	0.80	0.65	0.98	1135
	15	search support	search questions on support page	0.71	0.71	0.99	1329
	16	☆like article	like an article and send feedback	0.82	0.70	1.00	1230
	Average			0.79	0.55	0.99	1230
John Orion Young	17	☆add to shopping cart	add things to shopping cart	0.40	0.34	0.99	1428
	18	open market	browse joys on market page	0.89	0.64	0.97	3085
	19	select joy	view details of a joy	0.37	0.36	0.94	1350
	20	open shop	look clothes on the shop page	0.84	0.65	0.94	1342
	21	☆refresh shopping cart	refresh the shopping cart page	0.73	0.29	0.98	1585
	22	open john orion young	open the John Orion Young	0.94	0.48	0.99	2719
	23	view collector	view a collector homepage	0.87	0.68	0.98	2428
	Average			0.72	0.49	0.97	1991
Thomas Crown Art	24	☆add to shopping cart	add artworks to shopping cart	0.76	0.72	0.99	1443
	25	browse all artists	browse all artists on page	0.93	0.47	0.94	1117
	26	browse all artworks	browse all artworks on page	0.87	0.41	0.91	1251
	27	open the DApp	open the Thomas Crown Art	0.85	0.33	0.99	1264
	28	search	search artworks or artists	0.80	0.67	0.99	1491
	29	view blog	view details of a blog	0.74	0.45	0.91	1408
	30	look shopping cart	open the shopping cart page	0.82	0.25	0.97	1392
	31	select artist	view details of an artist	0.82	0.28	0.90	1341
	32	select artwork	view details of an artwork	0.71	0.29	0.99	1398
	Average			0.81	0.43	0.95	1345
Cryptoboiler	33	open Cryptoboiler	open the Cryptoboiler	0.80	0.52	1.00	1329
	34	view questions	view questions page	0.80	0.46	0.99	1398
	35	☆comment post	comment on a post	0.88	0.81	0.99	2735
	36	☆post a problem	post a problem in Cryptoboiler	0.94	0.86	0.99	1631
	37	☆like post	like or dislike a post	0.89	0.82	0.99	1832
	38	user homepage	browse one's homepage	0.70	0.43	0.99	1662
	39	view blog	view details of a blog	0.71	0.73	0.98	1678
	40	view question	view details of a question	0.80	0.40	0.98	1276
	41	search	search by keywords	0.95	0.84	0.99	1606
	Average			0.83	0.65	0.99	1683
Ethlance	42	☆open Ethlance	open the Ethlance	0.47	0.26	0.92	937
	43	☆user homepage	browse one's homepage	0.15	0.27	0.97	659
	44	☆look work	look suitable work	0.06	0.10	0.86	737
	45	☆look worker	look suitable workers	0.32	0.25	0.90	731
	46	☆become employer	fill information to be an employer	0.55	0.29	0.97	2123
	47	☆learn DApp	look Ethlance introduction	0.29	0.14	0.87	976
	48	☆different category	classified by different categories	0.49	0.38	0.70	1926
	49	☆search	search by keywords	0.83	0.36	0.86	2142
	50	☆become employer	fill information to be an employee	0.44	0.33	0.97	2118
	Average			0.40	0.26	0.89	1372

(*continued*)

Table 5. (*continued*)

DApps	ID	User behavior	Description	AppS	AF	OF	Flow
				F1	F1	F1	
Knownorigin	51	☆open Knownorigin	open the Knownorigin	0.61	0.49	0.98	1578
	52	☆view gallery	browse all artworks on the page	0.76	0.61	0.91	1388
	53	☆select artwork	view details of an artwork	0.45	0.44	0.85	1714
	54	☆like artwork	like or dislike an artwork	0.05	0.02	0.98	2504
	55	☆view activities	view activities happened in DApp	0.25	0.16	0.99	1592
	56	☆browse all artists	browse all artists on the page	0.71	0.59	0.99	1859
	57	☆select artist	view details of an artist	0.29	0.22	0.99	2761
	58	☆search	search by keywords	0.04	0.02	0.85	1804
	Average			0.40	0.32	0.94	1900
Staybit	59	☆open Staybit	open the Staybit	0.69	0.74	0.98	3909
	60	☆create payment	create a payment request to rent	0.73	0.79	1.00	2706
	61	☆view contract	view my contracts	0.81	0.83	1.00	4632
	62	☆accept payment	retrieve a payment request	0.75	0.76	1.00	4288
	Average			0.75	0.78	1.00	3884
Crowdholding	63	open Crowdholding	open the Crowdholding	0.87	0.70	1.00	2648
	64	select an article	view details of an article	0.62	0.35	0.98	2422
	65	☆comment	comment on an article	0.63	0.46	1.00	1343
	66	☆agree article	agree or disagree an article	0.64	0.39	0.96	2478
	67	user homepage	browse one's homepage	0.58	0.37	0.97	2118
	68	☆follow a person	follow a person in Crowdholding	0.63	0.44	0.97	2391
	69	☆create a project	create a project to find workers	0.35	0.23	0.98	2601
	70	view project	view details of a project	0.54	0.27	1.00	2749
	71	search	search by keywords	0.82	0.73	0.99	3198
	72	learn DApp	look crowdholding introduction	0.59	0.28	1.00	2552
	Average			0.63	0.42	0.99	2450
Latium	73	open Latium	open the Latium	0.55	0.33	0.97	3800
	74	select task	view details of a task	0.60	0.27	0.99	3128
	75	user homepage	browse one's homepage	0.64	0.24	0.93	3749
	76	look transaction web	look transaction page	0.76	0.62	0.96	6761
	77	look my tasks	look my tasks page	0.52	0.26	0.90	3885
	78	homepage	view homepage	0.58	0.40	0.97	4632
	79	☆create a task	create a task to find workers	0.84	0.33	0.98	4290
	Average			0.64	0.35	0.96	4321
Viewly	80	open Viewly	open the Viewly	0.84	0.75	0.98	6004
	81	watch video	watch video	0.68	0.45	0.97	2882
	82	user homepage	browse one's homepage	0.76	0.54	0.98	2420
	83	look video transaction	view video trading information	0.48	0.33	0.98	2475
	84	☆follow person	follow a person in the Viewly	0.69	0.60	0.98	2294
	85	look game rank page	look the rank of distribution game	0.65	0.51	0.97	2887
	86	☆create new channel	create a channel to store video	0.61	0.48	0.99	1329
	87	☆upload video to draft	upload local video to draft	0.73	0.65	0.97	1172
	88	search	search by keyword	0.78	0.67	0.96	3233
	Average			0.69	0.55	0.98	2744
Accuracy				0.6808	0.4760	0.9566	
Precision				0.7027	0.4654	0.9617	
Recall				0.6541	0.4588	0.9565	
F1 score				0.6664	0.4625	0.9553	

*☆ means that users need to connect Ethereum wallet.

4.5 Evaluation of General User Behaviors Classification

Since we are the first one to classify encrypted traffic in this scenario, and in order to confirm the validity of proposed features for general user behaviors clas-

sification, we compare our proposed features with Appscanner [21], FFP [20] and the features mentioned in Aiolli [2] to classify 15 general user behaviors. Appscanner represents the conventional traffic analysis approach, FFP and Aiolli are the state-of-the-art classification approaches about blockchain network traffic.

Table 6. Experimental results with different approaches

User behaviors	Appscanner			Aiolli features			FFP			Our features		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Open DApps	0.6981	0.5715	0.6081	0.3923	0.394	0.3827	0.642	0.5223	0.5578	0.9861	0.9877	0.9869
Open market	0.6807	0.5843	0.5254	0.3197	0.2942	0.2631	0.6153	0.5394	0.4729	0.9846	0.9809	0.9828
View detail	0.3655	0.3452	0.3357	0.2564	0.2697	0.2538	0.3849	0.327	0.3335	0.9784	0.9847	0.9816
Follow a user	0.6891	0.6073	0.5836	0.5377	0.4822	0.4858	0.6617	0.6065	0.5778	0.9816	0.991	0.9863
Like or dislike	0.5093	0.4668	0.4514	0.2743	0.2593	0.2409	0.4413	0.4865	0.4243	0.992	0.9923	0.9922
Create project	0.6160	0.4895	0.5121	0.3302	0.3186	0.3165	0.5532	0.5091	0.4978	0.9883	0.9899	0.9891
Search	0.6304	0.6076	0.5967	0.5033	0.4811	0.4835	0.656	0.6266	0.6191	0.9903	0.9873	0.9889
View homepage	0.6107	0.5297	0.5434	0.3448	0.3637	0.3447	0.5662	0.5226	0.5271	0.9829	0.9785	0.9807
Activities	0.5923	0.5447	0.5238	0.5172	0.4659	0.4602	0.6089	0.6096	0.5671	0.9864	0.9817	0.984
Add to cart	0.6764	0.4598	0.5050	0.6257	0.4874	0.5380	0.716	0.5929	0.6108	0.9985	0.9833	0.9909
Watch video	0.8488	0.5729	0.6470	0.5823	0.3876	0.4209	0.7895	0.5885	0.6302	0.986	0.9972	0.9916
Comment	0.7108	0.7506	0.6718	0.6353	0.6722	0.6238	0.6693	0.7631	0.6838	0.9972	0.9961	0.9967
DApps introduction	0.6887	0.5233	0.5120	0.3371	0.2612	0.2874	0.5982	0.5396	0.5021	0.9948	0.9753	0.985
Refresh cart	0.8006	0.7910	0.7581	0.4871	0.3445	0.3688	0.8136	0.7844	0.7743	0.9821	0.9902	0.9861
Else behaviors	0.6908	0.7468	0.6980	0.5452	0.5389	0.5178	0.7219	0.6724	0.6750	0.984	0.9847	0.9844
Average	**0.6539**	**0.5727**	**0.5648**	**0.4459**	**0.4014**	**0.3992**	**0.6292**	**0.5794**	**0.5635**	**0.9875**	**0.9867**	**0.9871**
Accuracy	0.5485			0.3826			0.5357			0.9857		

*P and R in this table means the metric Precision and Recall.

The comparison results are shown in Table 6. Our approach outperforms the other methods. The average precision of our method is 98.75%. Compared with Appscanner, FFP and Aiolli method, the precision of our method increases by about 33.36%, 35.83% and 54.16%, respectively. As for F1 score, our method also performs best, which can reach 98.71%. Compared with the other classification methods, the F1 of our method increases by about 42.23%, 42.36% and 58.79%, respectively. The improvement effect is very obvious. According to the classification results, we can intuitively see that our classifier is superior to the current classifiers.

4.6 Proposed Features Analysis

To implement our proposed method is useful and extracted features are better for general user behaviors classification. Seven feature sets are conducted from these features, as you can see in Table 7. PL distribution represents packet length distribution. IO time series represents inbound and outbound time series. IATS, BRC and PR statistics represent statistical features of inter-arrival time series, byte rate change and packet rate. TPIF, TPOF and FPAT represent total packets of inbound flow, total packets of outbound flow and the first packet arrival time.

The experimental results of different feature sets are shown in Fig. 5, and we draw some conclusions.

Table 7. Features information of experiments for 15 user behaviors classification

Feature	F I	F II (base experiment)	F III	F IV	F V	F VI	F VII
Time series	✓	✓	✓	✓	✓	✓	✓
Packet length series	✓	-	-	-	-	-	-
Burst	✓	-	-	-	-	-	-
PL distribution.	-	✓	✓	✓	✓	✓	✓
IO time series	-	-	✓	-	-	-	✓
Byte distribution	-	-	-	✓	-	-	✓
IATS, BRC and PR statistics	-	-	-	-	✓	-	✓
TPIF, TPOF and FPAT	-	-	-	-	-	✓	✓

*F in this table means feature sets.

1. One experiment only used the features mentioned in [20], which is called Feature I. Feature II is called base experiment. In order to know the impact of packet length distribution, we compare Feature I to Feature II. The comparison results show the uselessness of burst and packet length series in this classification. In terms of general user behaviors classification, packet length distribution is more useful than packet length series. As for accuracy, the classifier with Feature II is higher 39.59% points than the classifier with Feature I.

We conduct an additional experiment to find the reason for the poor performance of burst feature. Due to characteristics of DApps user behaviors, which are analyzed in Sect. 3.2, we extract number of packets for each user behavior, and find that more than 60% flows are short flows. The top-5 number of packets for each general user behavior are shown in Fig. 6. Most of flows have less than ten packets. Therefore, the classifier performs bad with this feature.

2. We consider that extracting time series of bidirectional flow may lose detailed information. The model using Feature III is higher 3.42% points than base experiment. So inbound and outbound time series can provide more details for general user behaviors classification.

3. Although transmission data is encrypted, behaviors with different action sequences produce some changes in payload, which cause the changes in traffic. So we use byte distribution to get more details, which can not be obtained from

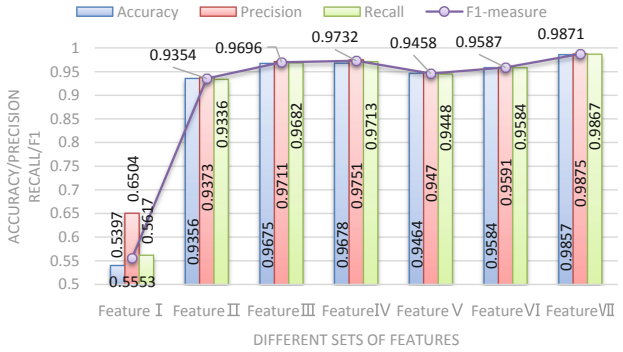

Fig. 5. Comparison results of Random Forest classifier with different sets of features.

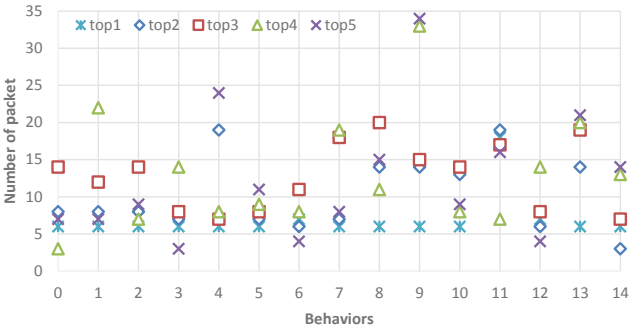

Fig. 6. The top-5 number of packets for each general user behavior. The X-axis label 0–14 represent 15 behaviors: open DApps(0), open market(1), view detail(2), follow a user(3), like or dislike(4), create project(5), search(6), view homepage(7), activities(8), add to cart(9), watch video(10), comment(11), DApps introduction(12), refresh cart(13), and else behaviors(14).

other features. Feature IV have the largest improvement, about 3.78% points in accuracy.

4. Compared with base experiment, Feature V and Feature VI contain different statistical features, and they improved the accuracy of 1.04% and 2.28%, respectively. These features can extract more information to improve the accuracy of the model. For example, byte rate change and packet rate can extract the intensity of user behaviors (i.e., action sequences).

5. According to Fig. 5, we can see that the result of Feature VII is the best. Compared with base experiment, the accuracy of the model is improved by 5.01%, the precision is improved by 5.02%, the recall is improved by 5.31%, the F1 is improved by 5.17%. We think the experiment of Feature VII achieves desirable results case it combines all features then obtains the maximum information.

5 Conclusion

In this paper, we focus on classification of DApps encrypted traffic for three application scenarios: DApps classification, DApps user behaviors classification and general user behaviors classification. For different scenarios, we extract different sets of effective features from DApps encrypted traffic after analyzing DApps characteristics. The experimental results show that our method has achieved satisfactory results in the encrypted traffic collected from 11 representative DApps. The accuracy of DApps classification reaches 99.5%, the accuracy of DApps user behaviors classification reaches 95.65%, and the accuracy of general user behaviors classification reaches 98.58%. The results demonstrate that our proposed features outperform the state-of-the-art methods. In the future, we plan to expand more application scenarios of DApps encrypted traffic classification and improve classification accuracy.

Acknowledgement. This work is supported by The National Natural Science Foundation of China (No. U1636217) and The Development Program for Guangdong Province under grant No. 2019B010137003 and National Key Research and Development Program of China (No. 2016QY05X1000) and The National Natural Science Foundation of China No. 61702501.

References

1. Dapps 2020. https://www.stateofthedapps.com/
2. Aiolli, F., Conti, M., Gangwal, A., Polato, M.: Mind your wallet's privacy: identifying bitcoin wallet apps and user's actions through network traffic analysis. In: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC 2019, Limassol, Cyprus, 8–12 April 2019. pp. 1484–1491 (2019). https://doi.org/10.1145/3297280.3297430
3. Alan H F, K.J.: Can android applications be identified using only TCP/IP headers of their launch time traffic? In: Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks, WISEC 2016, pp. 61–66 (2016)
4. Buterin, V., et al.: A next-generation smart contract and decentralized application platform. White Paper 3(37) (2014)
5. Cai, X., Zhang, X.C., Joshi, B., Johnson, R.: Touching from a distance: website fingerprinting attacks and defenses. In: the ACM Conference on Computer and Communications Security, CCS 2012, pp. 605–616 (2012). https://doi.org/10.1145/2382196.2382260
6. Chen, S., Wang, R., Wang, X., Zhang, K.: Side-channel leaks in web applications: a reality today, a challenge tomorrow. In: 31st IEEE Symposium on Security and Privacy, S&P 2010, pp. 191–206. IEEE Computer Society (2010), https://doi.org/10.1109/SP.2010.20
7. Conti, M., Mancini, L.V., Spolaor, R., Verde, N.V.: Analyzing android encrypted network traffic to identify user actions. IEEE Trans. Inf. Forensics Secur. **11**(1), 114–125 (2016). https://doi.org/10.1109/TIFS.2015.2478741

8. Coull, S.E., Dyer, K.P.: Traffic analysis of encrypted messaging services: Apple iMessage and beyond. Comput. Commun. Rev. **44**(5), 5–11 (2014). https://doi.org/10.1145/2677046.2677048
9. Draper-Gil, G., Lashkari, A.H., Mamun, M.S.I., Ghorbani, A.A.: Characterization of encrypted and VPN traffic using time-related features. In: Proceedings of the 2nd International Conference on Information Systems Security and Privacy, ICISSP 2016, Rome, Italy, 19–21 February 2016, pp. 407–414 (2016)
10. Duflos, S., Gay, V., Kervella, B., Horlait, E.: Integration of security parameters in the service level specification to improve QoS management of secure distributed multimedia services. In: International Conference on Advanced Information Networking & Applications (2017)
11. Finsterbusch, M., Richter, C., Rocha, E., Muller, J.A., Hanssgen, K.: A survey of payload-based traffic classification approaches. IEEE Commun. Surv. Tutor. **16**(2), 1135–1156 (2013)
12. Katsarakis, M., Teixeira, R.C., Papadopouli, M., Christophides, V.: Towards a causal analysis of video QoE from network and application QoS. In: Proceedings of the 2016 Workshop on QoE-Based Analysis and Management of Data Communication Networks, pp. 31–36. ACM (2016)
13. Korczyński, M., Duda, A.: Markov chain fingerprinting to classify encrypted traffic. In: IEEE INFOCOM 2014-IEEE Conference on Computer Communications, pp. 781–789. IEEE (2014)
14. Koshy, P., Koshy, D., McDaniel, P.: An analysis of anonymity in Bitcoin using P2P network traffic. In: Christin, N., Safavi-Naini, R. (eds.) FC 2014. LNCS, vol. 8437, pp. 469–485. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45472-5_30
15. Liang, Q., Wu, X., Lau, H.C.: Optimizing service systems based on application-level QoS. IEEE Trans. Serv. Comput. **2**(2), 108–121 (2009)
16. Nguyen, B.M., Dang, T., Nguyen, Q.: A strategy for server management to improve cloud service QoS. In: IEEE/ACM International Symposium on Distributed Simulation & Real Time Applications (2015)
17. Panchenko, A., et al.: Website fingerprinting at internet scale. In: NDSS (2016)
18. Shen, M., Wei, M., Zhu, L., Wang, M.: Classification of encrypted traffic with second-order Markov chains and application attribute bigrams. IEEE Trans. Inf. Forensics Secur. **12**(8), 1830–1843 (2017). https://doi.org/10.1109/TIFS.2017.2692682
19. Shen, M., Wei, M., Zhu, L., Wang, M., Li, F.: Certificate-aware encrypted traffic classification using second-order Markov chain. In: 2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS), pp. 1–10. IEEE (2016)
20. Shen, M., Zhang, J., Zhu, L., Xu, K., Du, X., Liu, Y.: Encrypted traffic classification of decentralized applications on Ethereum using feature fusion. In: Proceedings of the International Symposium on Quality of Service, IWQoS 2019, pp. 18:1–18:10 (2019). https://doi.org/10.1145/3326285.3329053
21. Taylor, V.F., Spolaor, R., Conti, M., Martinovic, I.: Appscanner: utomatic fingerprinting of smartphone apps from encrypted network traffic. In: IEEE European Symposium on Security and Privacy, EuroS&P 2016. pp. 439–454 (2016). https://doi.org/10.1109/EuroSP.2016.40
22. Wang, Q., Yahyavi, A., Kemme, B., He, W.: I know what you did on your smartphone: inferring app usage over encrypted data traffic. In: 2015 IEEE Conference on Communications and Network Security (CNS) (2015)

23. Yan, F., et al.: Identifying WeChat red packets and fund transfers via analyzing encrypted network traffic. In: 2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE), pp. 1426–1432 (2018)
24. Zhou, X., et al.: Identity, location, disease and more: inferring your secrets from android public resources. In: 2013 ACM SIGSAC Conference on Computer and Communications Security, CCS 2013, pp. 1017–1028. ACM (2013)