Classifying User Activities in the Encrypted WeChat Traffic

Chengshang Hou*†, Junzheng Shi*†, Cuicui Kang*†, Zigang Cao*†, Xiong Gang*†
*Institute of Information Engineering, Chinese Academy of Sciences
†School of Cyber Security, University of Chinese Academy of Sciences
Beijing, China
{houchengshang, shijunzheng, kangcuicui, caozigang, xionggang}@iie.ac.cn

***Abstract*—The security and privacy of encrypted mobile applications have attracted the attention of researchers. However, most of the existing researches focus on analysis of SSL/TLS traffic, while few studies focus on proprietary encrypted traffic, which is also important and challenging. In this paper, we make a deep study of WeChat, which is one of the most popular social applications in the world with over one billion active users. The application uses a proprietary encryption protocol called as MMTLS for most of its communications. It is designed based on Transport Layer Security (TLS) 1.3 drafts for both performance and security. We explore the fine-grained classification of typical user activities inside the MMTLS encrypted channels and compare the MMTLS with the HTTPS (e.g. flow duration and packet size), which are jointly used in WeChat. It is found that MMTLS is suitable for scenarios of low latency and lightweight messaging. With the WeChat traffic collected from different platforms (Android, iOS) and devices (Huawei, Samsung, iPhone, iPad, etc.) by different users, we classify seven typical activities, encrypted by MMTLS protocol such as payment, advertisement click, browsing moments and so on. The experimental results show that both of the average precision and recall can reach over 92%. Our work is the first to perform classification on this proprietary encrypted protocol and understanding the difference between MMTLS and TLS. It is believed that the work will benefit the security and privacy of WeChat and other proprietary encryption applications.**

***Index Terms*—WeChat, Encrypted traffic, proprietary protocol**

I. Introduction

With the popularity of smart mobile devices, mobile applications become one of the development trends of the Internet. Due to the rich content and convenient functions, these applications attract a large number of users. Application developers generally make use of cryptographic protocols to provide privacy and security for users' network activities and data. One typical direction is to identify user behaviors inside applications by traffic analysis without any decryption of users content. Hackers can eavesdrop on Wi-Fi and obtain user behavior meta data [1], [2] through this technic, while for network operators and application designers, it is very useful for network management, network optimization and user privacy enhancement.

Currently, two types of encryption protocols are exploited, namely the public encryption such as SSL/TLS and proprietary encryption. User action classification inside the SSL/TLS encrypted applications [1] [3] are quite common since SSL/TLS is widely used in cloud and has a lot of development libraries such as OpenSSL, GNUTLS and LibreSSL. Meanwhile, many social applications take use of non-SSL security protocols to improve efficiency, or enhance privacy and security, including WeChat MMTLS protocol, Telegrams MTP protocol, and the Signal protocol used by social applications such as Signal, WhatsApp, and Messenger. For instance, as one of the most popular mobile social networking applications, WeChat uses light weight MMTLS to improve resource usage efficicency and save on server resources since sustaining long time alive network connections is a burden for servers, while at the same time many conversations are short time. These social applications using proprietary protocols are popular around the world. Therefore, it is meaningful and necessary to study the privacy and security issues of them. For example, with the user data encrypted, enterprise network management and user behavior mining can benefit from the user activities meta data. However, user activities identification inside such applications is insufficient, especially for applications with both social and payment functions.

In this paper, we focus on WeChat, a feature-rich social application with more than one billion monthly active users [4] , which supports text and voice chat, voice and video call. It also offers services such as online payment, red packet (pay to a friend), mini program (third party services interface inside WeChat) and subscriptions. WeChat above version 6.6.x adopts a new proprietary security protocol, called MMTLS [5] as its main encryption protocol based on TLS 1.3 drafts [6]. MMTLS handshake retains TLS mechanism including 0-RTT mode and 1-RTT mode. However, WeChat employs MMTLS and SSL/TLS for different service scenarios. Therefore, WeChat is very representative.

First, we investigate the MMTLS protocol and WeChat traffic encrypted by MMTLS. Then, we compare the network characteristics between HTTPS and MMTLS and conduct the fine-grained user activities classification by traffic analysis. The results demonstrate that using basic statistical attributes of flow can accurately identify the user activities. The paper's contribution lies in two folds. On one fold, we in-depth analyze the security and usage scenarios of MMTLS, and compare it with HTTPS. We find that the MMTLS protocol is primarily

978-1-5386-6808-5/18/$31.00 ©2018 IEEE

used for encryption of user activities for non-web access. On the other fold, we use five classification models to conduct fine-grained user activities classification and compare their performance. The classifiers are decision tree, random forest, Naive Bayes, logistic regression and SVM. The results show that the random forest classifier has the best performance, achieving both precision and recall over 92%.

The rest of this paper is organized as follows. In Section II, the related works are reviewed. In Section III, we analyze the MMTLS protocol and introduce the WeChat service. Traffic traces collection and comparison analysis is described in Section IV. Then, several popularly used machine learning algorithms are used to deal with the fine-grained classification problem of MMTLS traffic in Section IV. Finally, Section VI concludes this paper.

II. Related Work

In this section, we provide an overview of related work on the encrypted traffic classification or encrypted traffic fingerprint. We categorize related work into encrypted traffic analysis, which includes encrypted website fingerprinting, encrypted application identifying and encrypted user activities classification, and related work focus on WeChat.

In the literature, previous traffic analysis on encrypted traffic has been a long time focus on encrypted website fingerprinting. The encrypted web identification was first studied by [7]. By collecting request size and html file size of target website, the authors infer the pages visited by users on a website. A descendant is [8], which computes the Jaccard coefficient of the observed web page objects fetched by the attack and victim. Subsequent studies extended to the fingerprint website transports in VPN (e.g. SSH tunnels) and Tor [9]–[12]. Liberatore and Levine employ the naive Bayes classifier [11]. In [9], packet sequence information is extracted to enhance website fingerprinting. Panchenko et al. presented a method called CUMUL, which utilizes the cumulative packet lengths [12]. Gonzalez et al. perform a more accurately profiling by fingerprint the 1st-level pages rather than main page [13]. Hayes and Danezis proposed a method that is comparing Hamming distance for the fingerprint of captured against the target [14].

Besides website fingerprinting, there are many works in the field of mobile application classification [15]. The earliest work is [16], which using a UI fuzzing technique and extracting fingerprinting of both HTTP and HTTPS Android applications. Taylor et al. proposed AppScanner, which is a framework for automatic fingerprinting and real-time identification mobile application [17].

In the perspective of the encrypted user activities, there are a lot of new works [1]–[3], [18]–[21]. Conti et al. [1] identified user activities within mobile applications encrypted by HTTPS. The authors apply their method to three applications including Gmail, Facebook and Twitter. Each application contains 4, 7, and 6 user action categories, respectively. Grolman et al. [19] applied transfer learning to identify user activities, which can build unlabeled user activities classifier from labeled user activities. Saltaformaggio et al. [2] proposed Netscope, which classifying the cluster of the behavior model extracting from multiple flow, archive 78% and 76% accuracy and recall . However, these works do not apply to the proprietary protocol. Park and Kim [20] proposed a traffic analysis on KakaoTalk, an instant message, that also used an proprietary protocol. The authors classify the user actions, including adding or deleting a friend. Our work is similar to them, which is both classify focus on a popular application. In contrast, our work is deeper that the user actions we choose are more popularly used . A slight different work is [21]. The authors classified the service type of skype, which also uses a proprietary protocol.

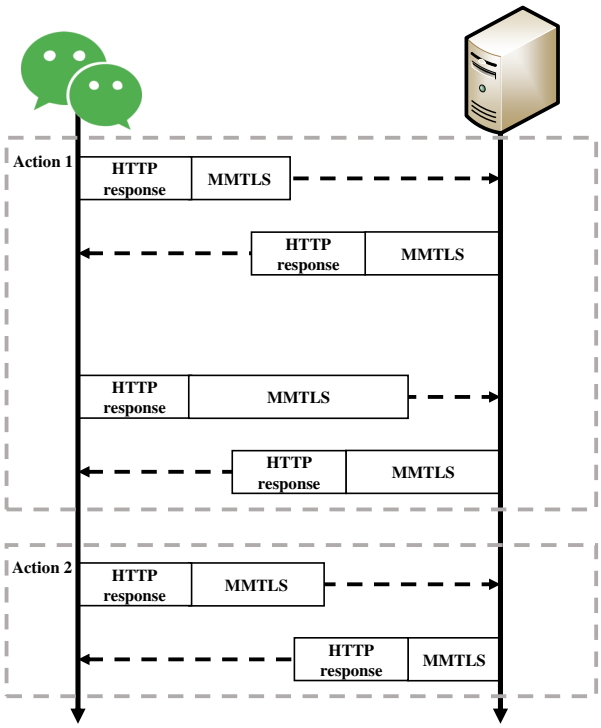


Fig. 1: The communication progress of WeChat using HTTP based MMTLS. User action 1 consists of multiple MMTLS sessions, while user action 2 consists of single HTTP session.

Furthermore, there are several works pay attention to analysis WeChat [22]–[25]. Specifically, Li et al. [22] compared the transoceanic vedio of WeChat with others e.g. Skype and FaceTime. Huang et al. [23] inferred the proprietary protocol format of WeChat traffic and measured WeChat in a cellular network. However, the work is not analysis on MMTLS since the MMTLS protocol was not used in WeChat versions of 4.5 and 5.0. In [25], the authors focused on the classification of WeChat text and picture message using machine learning algorithms. However, their work only considers two services, namely text and picture.

III. MMTLS Analysis

In this section, we introduce the MMTLS protocol and categorize the WeChat traffic into seven different services.

TABLE I: The list of WeChat services and corresponding application layer protocols

Service	**Activity**	HTTP based MMTLS	TCP based MMTLS	HTTPS	Others
Chat	-		•		•
File Transfer	-				•
Moments	browsing moments	•		•	
Payment	pay to service pay to a friend pay to a group	•			
Subscriptions	browsing subscription	•		•	
Mini program	open mini program	•		•	
Advertisement	advertisement click	•		•	

A. MMTLS introduciton

To conduct our analysis, we first describe the features of the MMTLS protocol. Due to there is no public specification of the MMTLS protocol, we only acquired the basic information of the protocol from an article about design of the MMTLS protocol, written by a developer of MMTLS protocol [5]. Two design goals of MMTLS protocol are high efficiency and security. This protocol is designed based on TLS 1.3 drafts. This protocol is a binary protocol without any cleartext. While the TLS protocol carries cleartext in it's handshake stage. To improve security, MMTLS uses ECDH (Elliptic Curve Diffie-Hellman) algorithm for key exchange. In order to reduce handshake latency, pre-shared key (PSK) is used to implement 0-RTT handshake.

Compared to TLS, MMTLS establishes an encrypted connection using the built-in public key distributed with WeChat instead of exchanging certificates for each connection. The attacker cannot compromise the encrypted channel by man-in-the-middle attack, which can be used to intercept the HTTPS channel [26]. Man-in-the-middle attack is an attack that attacker who is located in the communication link between the victim and the real server and they can manage to substitute the server certificate for a fake one. By doing this, the attacker can eavesdrop and distort the message. Due to the app built-in has a public key, which is similar with HTTP Public Key Pinning (HPKP), the attacker cannot disguise a real server.

B. WeChat services

WeChat provides users with various services. It is difficult to enumerate all the services of this highly integrated social networking application. In this work, we categorize the WeChat traffic into seven services. The detailed information of the categorization is listed as follows.

- **Chat**: It includes message and voice chat, voice massage and video call, which are basic function of WeChat.
- **File Transfer**: It includes sending pictures and other files.
- **Moments**: It is a social networking service of WeChat. WeChat users share content such as text, picture, and hyperlink to their friends, which is similar to the Facebook timeline.
- **Payment**: The payment is a function that a user pays other users money through scanning the QR code that generated by WeChat. In addition, WeChat also enables a user directly paying money to a friend or a WeChat group , which is popular in China.
- **Subscriptions**: It supports that WeChat users read articles published by the authors of subscriptions.
- **Mini program**: It enables users to use applets in WeChat, which is implemented by H5 and JavaScript technology. There are thousands of mini programs supplied by third parties.
- **Advertisement**: WeChat gets profit when users click on advertisements that appeared at the bottom of subscriptions article.

TABLE II: Overview of MMTLS over HTTP format

```
POST /mmtls/xxxxxxxx HTTP/1.1
Accept: */*
Cache-Control: no-cache
Connection: close
Content-Length: xxx
Content-Type: application/octet-stream
Host: short.weixin.qq.com
Upgrade: mmtls
User-Agent: MicroMessenger Client
...
```

```
HTTP/1.1 200 OK
Connection: close
Content-Type: application/octet-stream
Content-Length: xxx
...
```

In the study, we observe that the MMTLS protocol runs on the top of both TCP and HTTP protocol. The communication procedure of WeChat using HTTP based MMTLS is shown in Figure 1. The main difference of the usage between TCP-based MMTLS and HTTP-based MMTLS lie in the connection duration. The TCP-based MMTLS is usually used to maintain the connection. Different types of MMTLS are applied to encrypt the different services. Table I shows relationship between WeChat services and it's application layer protocols (i.e. HTTP based MMTLS, TCP based MMTLS, HTTPS). From the table, it can be obviously find that several services use both MMTLS and HTTPS simultaneously. Most WeChat services are encrypted with HTTP-based MMTLS, except chat and file transfer services. For example, the chat service is encrypted via MMTLS over TCP, while the payment activity and browsing moments activity are encrypted via MMTLS over HTTP. In the next section, we will focus on the MMTLS over HTTP for it cover mostly kinds of services.

The HTTP header of the HTTP based MMTLS protocol is shown in Table II. From the table, it is clearly seen that the HTTP header encapsulating the MMTLS protocol is different to ordinary HTTP header. Specifically, the HTTP header Content-Type is always set *application/octet-stream*. The header contains a special HTTP header field *Upgrade: mmtls*. The User-Agent of the header is always set to *MicroMessenger Client*. The HTTP payload is the MMTLS protocol. It should be noted the `xxxxxxxx` in the URL of Post method field represents a random 8-bit hexadecimal code varied in different requests.

C. User activities

A service may associate with several users' activities. For example, pay to a friend or a group. The user model is defined as the user who use services to generate activities in a sequential manner. In other word, two services won't be used at the same time, e.g. a user cannot pay someone and browse the moments, simultaneously. Figure 1 illustrates the process that WeChat invokes user activities. As it is shown, a user action may trigger either single or multiple HTTP sessions. In case of multiple HTTP sessions, the session initiation may be sequential or concurrent.

We discover that WeChat uses both MMTLS encryption protocol and TLS encryption protocol in the user activities. A typical user activity trigger service request is as follows. First, the application fetches meta information from WeChat server by MMTLS. Then, according to the MMTLS response, application visits an HTTP site or HTTPS site.

The user activities chosen in the fine-grained classification are based on two criteria. One is privacy related activities and another is commonly used activities. The collection of different user activities is very large. Thus, our targets activities are those unique and critical activities. Considering the time when the user switches the service, once the critical activities is detected, the service currently used by the user will be determined. According to the categories that we give out in subsection III-B, we chose seven typical activities namely browsing moments, browsing subscriptions, opening the mini program (shown as mini program), advertisement click, and pay to service, a group or a friend. The correspondence between services and activities is shown in Table I.

IV. MMTLS Database

In this section, we describe how we collect the labeled MMTLS traffic and analysis of the data.

A. Data collection

We created the wireless network with a portable wireless access point. The traffic of this network was captured by Wireshark. In order to avoid being interfered, the following measures were adopted. First, making sure the promiscuous mode of Wireshark off to avoid capturing other channel signal. Second, each smartphone connected to a wireless network exclusively. Third, applications running background were closed.

TABLE III: Traffic traces summary

Activities	**#activities**	**#MMTLS Flow**	**%**
browsing moments	239	382	5%
open mini program	169	1323	17%
pay to service	162	1188	15%
pay to a friend	171	992	12%
pay to a group	220	1353	17%
browsing subscriptions	219	1965	25%
advertisement click	395	734	9%
total	1575	7937	-

In order to get labeled MMTLS protocol traffic, we captured WeChat traffic traces generated by different user activities. According to the selected activities in subsection III-C, we separately collected traffic trace generated by each activity. To simulate the general user activities scenes, we collected traffic traces from 27 users with their own experimental WeChat account. For each activity, the collection are conducted for at least 160 times. In order to generalize the experimental data, we collect WeChat traffic from different WeChat versions, platforms and device vendors. Specifically, the WeChat versions are ranging from 6.6.5 to the latest release of 6.6.6. WeChat traffic are collected from both Android including versions of 4.4, 6, 7, 8 and iOS including versions of 8, 10, 11. The devices consists of Apple, Huawei, Samsung, etc. With the diverse devices and users, we can get generalized datasets that collect similar with the real world.

The port 80 is used to filter the HTTP traffic and port 443 is used to filter the SSL traffic. To identify the MMTLS flow of WeChat, we check the special HTTP header field *Upgrade: mmtls* (see Table II for more information). The failed MMTLS requests are discarded (i.e. no server response).

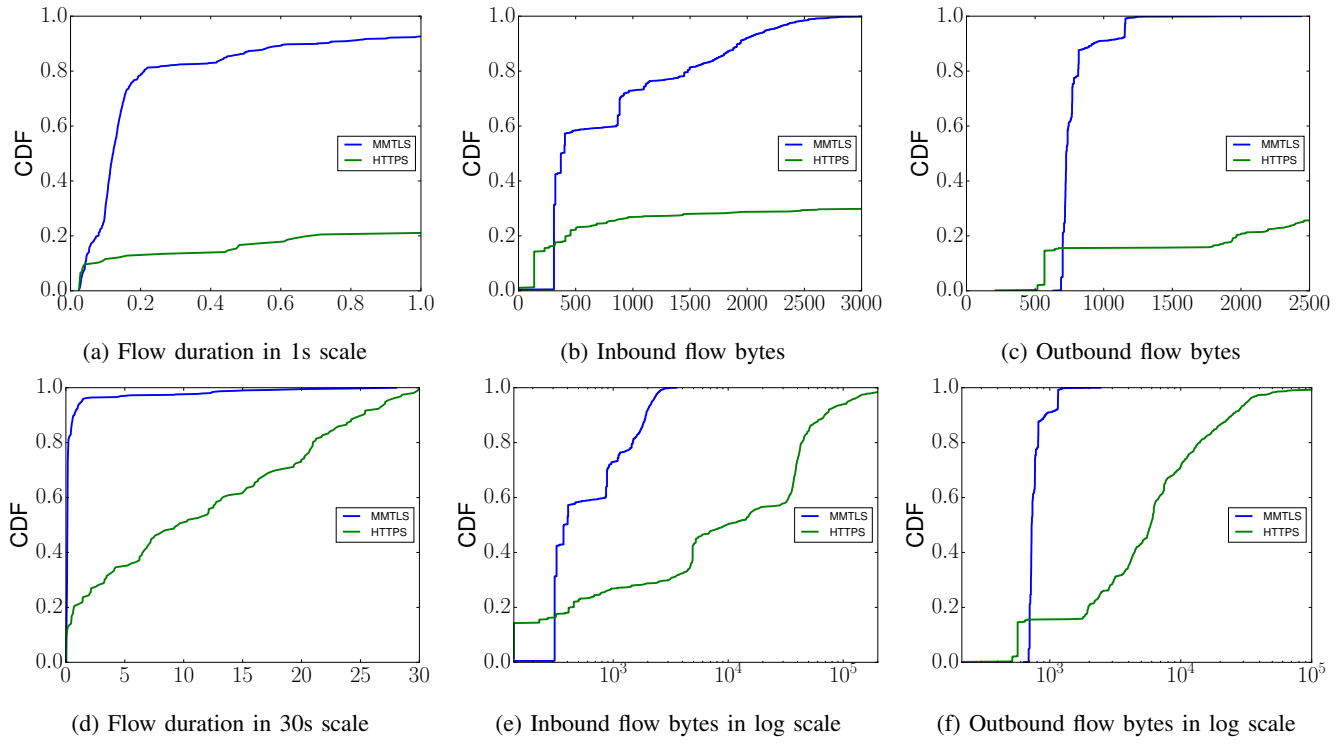
The traffic traces may include background traffic such as heartbeat requests, periodic events that not generated by WeChat. To estimate the impact of this situation, we conducted an experiment that captures the traffic without any activity. It is found that only two short MMTLS appeared with twice waking up the screen during 10min. Thus, The background traffic of WeChat in our collection has a light impact.

B. MMTLS-8K Datasets

With 27 WeChat accounts and users, different devices and platforms, we got 1.2G labeled WeChat traffic traces among seven different activities. Finally, there are we got 7937 MMTLS flows within 1575 activities. The summaries of these activities we captured are reported in Table III. The failed requests is not shown in the Table.

C. The comparison with HTTPS

To gain insight of this special application service model, we analyzed the browsing subscriptions activities. The activity is chosen because it is easy to identify the HTTPS flows of user activities through domain *mp.weixin.qq.com* , which is mainly used for browsing subscriptions activities. From those traffic traces of browsing subscriptions activities in the dataset, we extract 630 HTTPS flows with SNI *mp.weixin.qq.com* and 1965 MMTLS flows.



(a) Flow duration in 1s scale (b) Inbound flow bytes (c) Outbound flow bytes

(d) Flow duration in 30s scale (e) Inbound flow bytes in log scale (f) Outbound flow bytes in log scale

Fig. 2: Comparison between MMTLS and HTTPS in the cumulative distribution of both flow duration and bytes

1) Comparison of duration: As shown in Figure 2a and 2d, 70% of MMTLS flows duration are less than 0.25 seconds and almost 100% of MMTLS flow duration less than 2s connection. While the connection durations of HTTPS flow are uniformly distributed among 0s to 30s. This indicates that the MMTLS protocol is used for the low latency scene.

2) Comparison of byte : Figure 2b, 2e, 2c and 2f show the cumulative distribution of the number of inbound and outbound bytes. The x-axes of Figure 2e and 2f are represented on a logarithmic scale. For approximate 70% of MMTLS flow, its inbound bytes volumn less than 1KB. As for the HTTPS, 60% of flow's length longer than 10KB. From the Figure 2c and 2f we can see that, for 80% of MMTLS flow, the outbound bytes less than 1 KB. In contrast, for HTTPS, more than 50% of flows longer than 10KB. Thus, the MMTLS protocol is used for encrypting lightweight message.

V. Classification and Evaluation

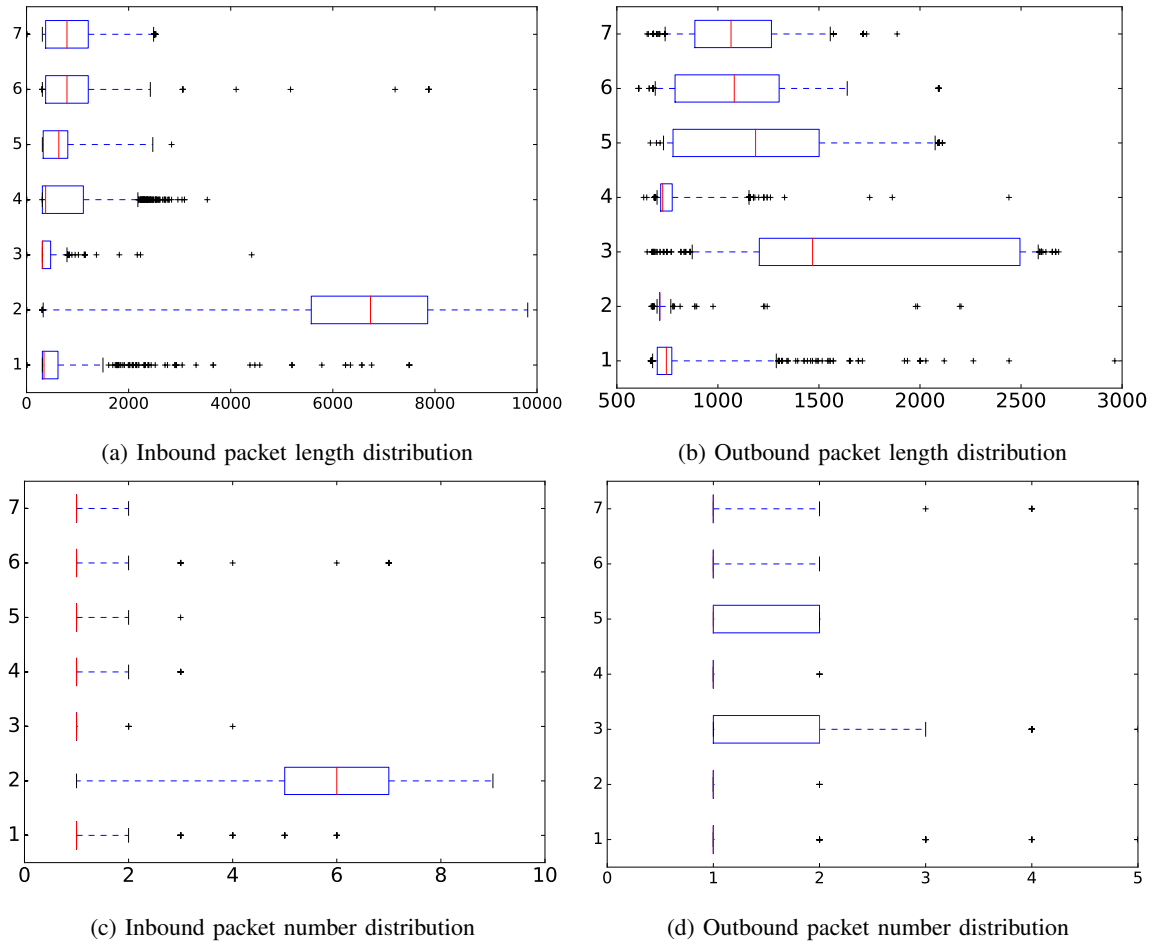
In this section, we try to deal with the fine-grained MMTLS traffic classification problem on the database collected in the previous section. At first, the feature engineering are introduced in detail. Then, five famous and state-of-the-art algorithm are evaluated on the database.

A. Feature engineering

In general, different activities have different packet distributions. In Figure 3, the length distribution of inbound and outbound packets for each activity are displayed. The open mini program, depicted as class 1 in the figure, has a long tail distribution for the sum of inbound and outbound packet length. The browsing moments activity shows a very short and stable outbound packet distribution and a very long inbound packet. In contrast, the advertisement click activity shows a completely opposite ratio. Activity 5, 6, 7 that are similar activity that paying to a service, a friend or a group, show a similar inbound packet length distribution, but they have different outbound packet length distribution. Figure 3c and 3d show that the packet number distribution is similar for most of the activities, except the browse moments activity.

Based on these statistical analysis, the first set of features is chosen as the number of bidirectional packets and the cumulative size of bidirectional packet. The second 7 features are the minimum, maximum, mean, standard deviation, percentile of packet length sequence. In particular, we chose 0.25, 0.5, and 0.75 percentile of the packet length sequence. The third set of features is the packet length distribution. We use bins to represent the packet length distribution. For example, when the bin step size is 100 and MTU is 1500 (the maximum of a packet length), the packet length in the range [0, 100) is assigned to the first bin, the packet length in the range [100,200) was assigned to the second bin and so on. By this method, we partition the packet lengths to 15 bins. We evaluated three different scales to discretize the packets length sequence, namely 10,100, and 500 for the bin steps. According to the 10-fold cross validation result, we finally chose 100 as the bins step size.

To avoid having an impact on the classifier, the retransmission packets and TCP packets with a zero-length payload such



(a) Inbound packet length distribution

(b) Outbound packet length distribution

(c) Inbound packet number distribution

(d) Outbound packet number distribution

Fig. 3: Statistical distribution of the length of packet length of each activity. The median is represented as the red line. First and third quartile are represented as the left and right side of the box. The whiskers represented packet length beyond the first and third quartiles and it's range in 5% to 95%. The Y-axis label 1-7 represent seven activities, which are open mini program (1), browse moments (2), advertisement click (3), browse subscription (4), pay to service (5), pay to a friend (6) and pay to a group (7).

as SYN, FIN, and RST are filtered out. We scale the means and standard deviations of each feature to 0 and 1, respectively.

B. Experimental setting

The evaluated classifiers include Naive Bayes, Logistic regression, SVM, Decision tree and Random forest algorithms. We used the scikit-learn machine learning library [1] to implement those classifier. For the Naive Bayes classifier, we adopt the Gaussian classifiers. For the Logistic regression classifier, the solver 'liblinear' is used, the penalty is set to 'l2' and the inverse of regularization strength is set to 100. We use RBF kernel for SVM classifier with penalty parameter set to 2^{17} and kernel coefficient is set to 2^{8}. Decision tree classifier is implement by the CART algorithm with Gini impurity. We use 10 estimates (the tree number of Random forest) and the maximum depth of a tree is 2/3 the number of total features. When looking for the best split, the number of features is set as the square root of the number of all features. Finally, bootstrap samples are used when building trees to improve performance.

As for the evaluation metrics, the use precision and recall metrics imported for all the classifiers. The $precision$ for each label is computed as $TP/(TP + FP)$ and $Recall$ for each class is computed as $TP/(TP + FN)$, where TP represents the true positive, FP represents false positive, FN represents false negative. The average precision and Recall is weighted by the number of samples for each class. The F1-score is computed as $F_1 = 2 * Precision * Recall/(Precision + Recall)$.

C. Results

The precision and recall are displayed in Table IV, and the $F1$ score is shown in Table V. It should be noted that we randomly select half of the samples in the database as the training set, and leave the remaining samples as the test set. The training set and test set contain traffic traces generated by different users and captured from different portable wireless

[1] **scikit-learn** http://scikit-learn.org/

TABLE IV: Classification performance of different user activities

Id	Activity	Naive Bayes		Random forest		Decision Tree		Logistic Regression		SVM		Support
		precision	recall	precision	recall	precision	recall	precision	recall	precision	recall	
1	Open mini program	0.72	0.39	0.92	0.88	0.91	0.88	0.64	0.43	0.97	0.82	655
2	Pay to a group	0.40	0.54	0.89	0.88	0.89	0.88	0.54	0.70	0.91	0.87	683
3	Browse Moments	0.72	0.84	0.91	0.93	0.92	0.93	0.95	0.83	0.93	0.52	192
4	Browse subscriptions	0.51	0.69	0.96	0.95	0.96	0.95	0.54	0.92	0.80	0.98	985
5	Ad click	0.33	0.80	0.94	0.93	0.95	0.91	0.89	0.70	0.98	0.89	364
6	Pay to service	0.51	0.16	0.92	0.95	0.91	0.95	0.65	0.50	0.93	0.94	585
7	Pay to a friend	0.35	0.04	0.88	0.90	0.86	0.92	0.55	0.08	0.89	0.90	505
-	**Average**	0.50	0.47	0.92	0.92	0.92	0.92	0.63	0.61	0.90	0.89	3969

TABLE V: F-1 score of different user activities

Id	Activity	Naive Bayes	Random forest	Decision Tree	Logistic Regression	SVM	Support
1	Open mini program	0.50	0.90	0.89	0.51	0.89	655
2	Pay to a group	0.46	0.88	0.89	0.61	0.89	683
3	Browse Moments	0.78	0.92	0.93	0.89	0.67	192
4	Browse subscriptions	0.58	0.96	0.95	0.68	0.88	985
5	Ad click	0.46	0.93	0.93	0.78	0.93	364
6	Pay to service	0.24	0.93	0.93	0.56	0.93	585
7	Pay to a friend	0.08	0.89	0.89	0.14	0.90	505
-	**Average**	0.43	0.92	0.92	0.57	0.89	3969

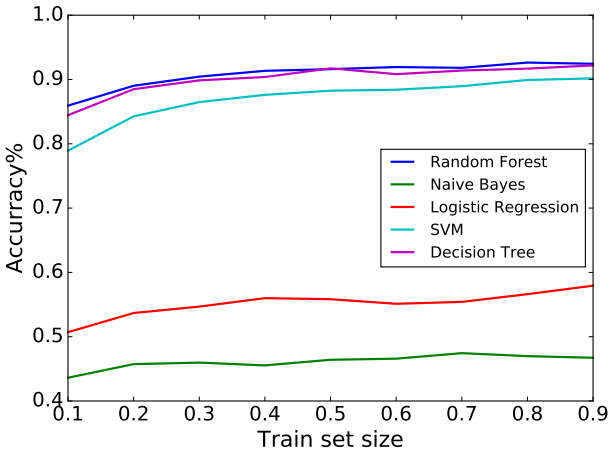


Fig. 4: The performances on different training set size

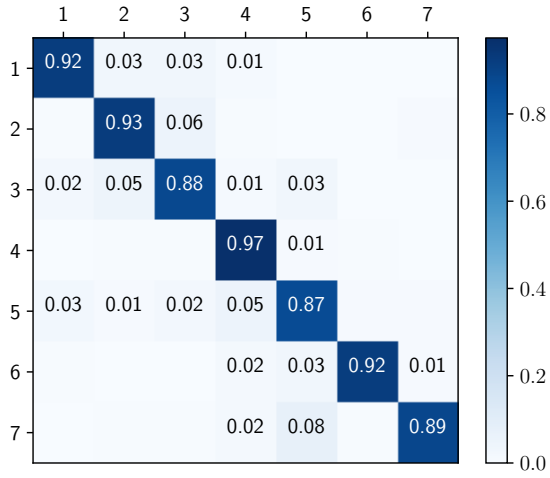


Fig. 5: Random forest classifier: confusion matrix for 7 activities. The X and Y-axes label 1-7 represent seven activities: pay to service (1), pay to a friend (2), pay to a group (3), browse subscriptions (4), open mini program (5), advertisement click (6), browse moments (7).

access points. The Random forest classifier achieves the best performance. The results of the ten fold cross validation of random forests achieve an average weighted F1-score of 92.5% with a standard deviation of 0.8%. The browsing subscriptions activity shows a highest F1-score which is up to 96%. The decision tree classifier shows a little bit lower performance than the random forest classifier. The overall F1-score of the decision tree classifier is 1% lower than the classification result of the random forest classifier. Compared with the random forest classifier, the precision and recall of the decision tree classifier fluctuate by 2%. The SVM classifier is the third best classier. The SVM classifier reaches a slightly higher F1-score than the random forest classifier and the decision tree classifier on the classification of the paying to a friend activity. It is the recall of the browsing moments activity affect its performance. Almost 50% of the browsing moments activities are misclassified to the browsing subscriptions class. We suspect that it is caused by the proportion of the training sample size between browsing moments and subscriptions. The logistic regression classifier and Naive Bayes classifier get a poor performance on the encrypted user activities. These two classifiers rarely correctly predict the activity of pay to a friend, which is got 8% and 4% recall rate respectively.

To evaluate the influence of training size to the classifiers. Random sampling ranges from 10% to 90% of dataset as the training set, the remains of dataset is treated as test set. The performances of classifier in different scales are shown in Figure 4. From the figure, we can clearly see that the Random forest classifier and Decision tree classifier achieve the best performances. However, the Logistic regression classifier and

Naive Bayes classifier do not work well, which is similar with the results in Table IV and V.

In order to see the detailed confusing information of the activities, we plot the confusion matrix of the different activities in Figure 5. The results with the random forest classifier are used since it gets the best overall performance. From the figure, we can clearly see that the browsing subscriptions activity and advertisement click activity are correctly classified for more than 95% of the cases. The three payment activities get a bit mixed up. However these mistakes are not too bad (less than 7% for pay to service and pay to a friend). The opening mini program activity is the most difficult one to be identified, since 13% of the activities are wrongly classified to the other 6 activities. The reason may be that mini program MMTLS flows are varied and diverse due to the flows of mini program activities in the dataset is collected from approximate 80 different mini programs .

VI. Conclusion

In this paper, we conduct fine-grained classification on the WeChat user activities through traffic analysis, which is very representative of encrypted mobile applications. To the best of our knowledge, this is the first work to apply traffic analysis to the proprietary encryption MMTLS protocol. The protocol is analyzed in depth and is compared to HTTPS in its usage in network conversation. We extract effective features from the encrypted traffic. By utilizing five well-known machine learning classification algorithms, we achieve a high performance with 92% precision and recall for the fine-grained user behavior classification. We believe that our work is meaningful for both protocol security design and user privacy enhancement in mobile application services.

Acknowledgment

This work is supported by The National Natural Science Foundation of China (No. 61602472, No. U1636217), the National Key Research and Development Program of China (NO. 2016YFB0801200, NO. 2016QY05X1000). Junzheng Shi is the corresponding author. Email: shijunzheng@iie.ac.cn.

References

[1] Mauro Conti, Luigi V. Mancini, Riccardo Spolaor, and Nino Vincenzo Verde. Can't you hear me knocking: Identification of user actions on android apps via traffic analysis. In *CODASPY*, 2015.

[2] Brendan Saltaformaggio, Hongjun Choi, Kristen Johnson, Yonghwi Kwon, Qi Zhang, Xiangyu Zhang, Dongyan Xu, and John Qian. Eavesdropping on fine-grained user activities within smartphone apps over encrypted network traffic. In *WOOT*, 2016.

[3] Scott E Coull and Kevin P Dyer. Traffic analysis of encrypted messaging services: Apple imessage and beyond. *ACM SIGCOMM Computer Communication Review*, 44(5):5–11, 2014.

[4] Rayna Hollander. Wechat has hit 1 billion monthly active users. Accessed: 2018-05-08. http://www.businessinsider.com/wechat-has-hit-1-billion-monthly-active-users-2018-3, 2018.

[5] Tencent. Mmtls: Introduction of tlsv1.3 based tecent security communication prtocol. Accessed: 2018-05-08. https://github.com/WeMobileDev/article/blob/master/SUMMARY.md, 2017.

[6] IETF. The transport layer security (tls) protocol version 1.3. https://tools.ietf.org/html/draft-ietf-tls-tls13-28.

[7] Heyning Cheng and Ron Avnur. Traffic analysis of ssl encrypted web browsing. *URL citeseer. ist. psu. edu/656522. html*, 1998.

[8] Qixiang Sun, Daniel R Simon, Yi-Min Wang, Wilf Russell, Venkata N Padmanabhan, and Lili Qiu. Statistical identification of encrypted web browsing traffic. In *null*, page 19. IEEE, 2002.

[9] Liming Lu, Ee-Chien Chang, and Mun Choon Chan. Website fingerprinting and identification using ordered feature sequences. In Dimitris Gritzalis, Bart Preneel, and Marianthi Theoharidou, editors, *Computer Security – ESORICS 2010*, pages 199–214, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[10] Tao Wang, Xiang Cai, Rishab Nithyanand, Rob Johnson, and Ian Goldberg. Effective attacks and provable defenses for website fingerprinting. In *USENIX Security Symposium*, pages 143–157, 2014.

[11] Marc Liberatore and Brian Neil Levine. Inferring the source of encrypted http connections. In *ACM Conference on Computer and Communications Security*, 2006.

[12] Andriy Panchenko, Fabian Lanze, Jan Pennekamp, Thomas Engel, Andreas Zinnen, Martin Henze, and Klaus Wehrle. Website fingerprinting at internet scale. In *NDSS*, 2016.

[13] Roberto Gonzalez, Claudio Soriente, and Nikolaos Laoutaris. User profiling in the time of https. In *Proceedings of the 2016 Internet Measurement Conference*, pages 373–379. ACM, 2016.

[14] Jamie Hayes and George Danezis. k-fingerprinting: A robust scalable website fingerprinting technique. In *Proceedings of the 25th USENIX Conference on Security Symposium*, pages 1187–1203. USENIX Association, 2016.

[15] Stanislav Miskovic, Gene Moo Lee, Yong Liao, and Mario Baldi. Appprint: Automatic fingerprinting of mobile applications in network traffic. In *PAM*, 2015.

[16] Shuaifu Dai, Alok Tongaonkar, Xiaoyin Wang, Antonio Nucci, and Dawn Xiaodong Song. Networkprofiler: Towards automatic fingerprinting of android apps. *2013 Proceedings IEEE INFOCOM*, pages 809–817, 2013.

[17] Vincent F. Taylor, Riccardo Spolaor, Mauro Conti, and Ivan Martinovic. Appscanner: Automatic fingerprinting of smartphone apps from encrypted network traffic. *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 439–454, 2016.

[18] Mauro Conti, Luigi V. Mancini, Riccardo Spolaor, and Nino Vincenzo Verde. Analyzing android encrypted network traffic to identify user actions. *IEEE Transactions on Information Forensics and Security*, 11:114–125, 2016.

[19] Edita Grolman, Andrey Finkelshtein, Rami Puzis, Asaf Shabtai, Gershon Celniker, Ziv Katzir, and Liron Rosenfeld. Transfer learning for user action identication in mobile apps via encrypted trafc analysis. *IEEE Intelligent Systems*, 33:40–53, 2018.

[20] Kyungwon Park and Hyoungshick Kim. Encryption is not enough: Inferring user activities on kakaotalk with traffic analysis. In *WISA*, 2015.

[21] Maciej Korczynski and Andrzej Duda. Classifying service flows in the encrypted skype traffic. In *Proceedings of IEEE International Conference on Communications, ICC 2012, Ottawa, ON, Canada, June 10-15, 2012*, pages 1064–1068, 2012.

[22] Jian Li, Zhenhua Li, Yao Liu, and Zhi-Li Zhang. Do twin clouds make smoothness for transoceanic video telephony? In *Parallel Processing (ICPP), 2015 44th International Conference on*, pages 260–269. IEEE, 2015.

[23] Qun Huang, Patrick P. C. Lee, Caifeng He, Jianfeng Qian, and Cheng He. Fine-grained dissection of wechat in cellular networks. In *23rd IEEE International Symposium on Quality of Service, IWQoS 2015, Portland, OR, USA, June 15-16, 2015*, pages 309–318, 2015.

[24] M. Shafiq, X. Yu, A. A. Laghari, L. Yao, N. K. Karn, F. Abdessamia, and Salahuddin. Wechat text and picture messages service flow traffic classification using machine learning technique. In *2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pages 58–62, Dec 2016.

[25] Muhammad Shafiq, Xiangzhan Yu, and Asif Ali Laghari. Wechat traffic classification using machine learning algorithms and comparative analysis of datasets. *International Journal of Information and Computer Security*, 10(2-3):109–128, 2018.

[26] Franco Callegati, Walter Cerroni, and Marco Ramilli. Man-in-the-middle attack to the HTTPS protocol. *IEEE Security & Privacy*, 7(1):78–81, 2009.