information

Tracked for Impact Factor

CITESCORE 4.2

Article

Identifying WeChat Message Types without Using Traditional Traffic

Qiang Zhang, Ming Xu, Ning Zheng, Tong Qiao and Yaru Wang

Special Issue

Machine Learning for Cyber-Security

Edited by

Dr. Xavier Bellekens

MDPI

https://doi.org/10.3390/info11010018

information

MDPI

Article

Identifying WeChat Message Types without Using Traditional Traffic

Qiang Zhang [1], Ming Xu [1,*], Ning Zheng [1,2], Tong Qiao [1] and Yaru Wang [2]

[1] School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310000, China; dq.z@outlook.com (Q.Z.); nzheng@hdu.edu.cn (N.Z.); tong.qiao@hdu.edu.cn (T.Q.)

[2] School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310000, China; 172050077@hdu.edu.cn

* Correspondence: mxu@hdu.edu.cn

Received: 14 November 2019; Accepted: 24 December 2019; Published: 26 December 2019

Abstract: Attackers can eavesdrop and exploit user privacy by classifying traffic into different types of in-app service usage to identify user actions. WeChat is the largest social messaging platform, which is a popular application in China. When WeChat is shut down, it is unable to generate traffic; that is, traditional traffic. However, the traffic still can be generated by system. How to identify the message types within WeChat with traffic generated by a system instead of traditional traffic becomes a new challenge. To deal with this challenge, we designed a system to identify and analyze the traffic of the Apple Push Notification service (APNs) to identify the message types of WeChat. In detail, we designed a system to identify and analyze the traffic of the APNs. First, the system clusters the traffic based on the session and divides it into multiple bursts. Then, it extracts the features of each burst and sends these features to the learning-based classifier to extract APNs's traffic from the background traffic. Finally, it uses a hash-based lookup table method to analyze message types from APNs traffic. Extensive evaluation results show that we can accurately identify the six message types of APN and WeChat. In addition, we propose two coping strategies for the method proposed in this article.

Keywords: traffic classification; APNs analysis; network traffic analysis; message type

1. Introduction

WeChat is China's largest social messaging platform with sending/receiving functions of voice messages, videos, pictures, text, red packet (WeChat red packet is also referred to as "Lucky Money" or "Red Envelope".), and fund transfers (WeChat fund transfer allows the transfer of funds between individual users) [1]. Users can send or receive virtual currencies online through the red packet and fund transfer functions. While consumer technology brings convenience, users also face a lot of privacy and security risks. For example, a malicious sender may send an image depicting a red packet or fund transfer which contains a phishing URL. The study of user action identification has remained a hot topic in recent years.

Network traffic classification intends to identify categories of the traffic of network packets or flows. It is an area that attracts the attention of researchers. Identifying user actions can be used to reveal sensitive information about user action preferences [2–7]. For example, attackers can eavesdrop on private information about appointments (such as frequent chatting and browsing of personal homepages on dating applications) or personal health information (such as querying and consulting disease information on health applications). Therefore, it is necessary to strengthen privacy protection in user action.

Research on identifying users' specific actions (i.e., sending/receiving a red packet) can analyze user preferences and economic conditions. Therefore, the information can be used for advertising push and Internet fraud [8]. As far as we know, most of the previous work was aimed at identifying user action through traditional traffic (i.e., traffic generated by applications). However, few works used system traffic to classify user action.

What is more, users may prohibit the permissions of some sensitive applications from using cellular traffic and Wi-Fi functions when they perceive that the network environment is not secure. Furthermore, some applications, such as Skype, cannot work properly without using a proxy in China. We do not know whether the aforementioned methods are effective when an application uses a proxy. Therefore, when an application does not generate traffic or uses a proxy, the question of how to use the traffic generated by a system to identify user action becomes a challenge.

Due to the remote notification service, the operating systems (Android and iOS) can still receive notifications from an application even if the application is quit. If a notification for an application arrives when that application is not running, the device alerts the user that the application has data waiting for it. Android uses Google's cloud messaging service to notify the user. Owing to the openness of Android, many third-party notification services appear, such as Huawei's Push Service (https://developer.huawei.com/consumer/en/service/hms/catalog/huaweipush_agent.html), especially in China. Thus, it is difficult to choose a representative one to analyze. On the contrary, because of the closure of iOS, the Apple Push Notification service (APNs) is the most powerful remote notification service for iOS [9]. Therefore, we selected the most representative APNs for in-depth analysis of the traffic. What is more, in Android, WeChat uses a daemon process to communicate with the WeChat server, which is responsible for receiving WeChat messages. It belongs to the traffic generated by WeChat, but when the user prohibits WeChat from using the traffic or prevents WeChat from running in the background, WeChat cannot receive the message. It cannot meet the application scenarios mentioned in this study regarding classifying WeChat message types when the application does not generate traffic. On the iOS side, when WeChat is running on the foreground, WeChat directly communicates with the WeChat server to obtain messages, but when WeChat is running in the background or is prohibited from using traffic, WeChat uses APNs to remind users of the message. So when WeChat is banned from using traffic, iOS systems can still analyze WeChat message types but Android cannot.

We further investigated more details of the APNs. Unlike traditional traffic, the APNs can still generate traffic in the following two situations. In the first case, the user disables network traffic which could be captured by an attacker. The application receives notifications from APNs and displays them on the iOS interface. In another case, the user sets the system to disable the application from generating message notifications. APNs traffic can still be generated, but notifications are not displayed on the interface. This leads to the disclosure of personal privacy without the user's knowledge.

Although the APNs uses an encrypted protocol to prevent information from being eavesdropped on, we can still obtain valuable information from the side-channel information of the encrypted traffic generated between iOS and the APNs server to determine the message types. By identifying the message types, we can predict the passive actions of user. Conti et al. [4] defined a passive action of a user as the action of the user receiving the message. Thus, we focused on identifying the types of messages in this study.

1.1. Contributions

In this study, we used a simplified and powerful way of identifying the message types sent on to WeChat. The proposed method analyzes the network traffic generated by APNs rather than WeChat server. Our proposed novel algorithm is efficient and fast. We identified and analyzed the traffic of the APNs that carried WeChat message notifications and successfully identified the six most frequently used message types of WeChat. Hence, the main contributions of this study are the following:

1. First, we enumerated the characteristics of APNs and analyzed the traffic patterns of APNs.

2. We extracted five first-order statistical features to effectively identify APNs traffic from background traffic and compared the performances of four widely used machine learning algorithms for APNs traffic recognition. The results show that the proposed classifier shows better performance in terms of efficiency and accuracy than the state-of-the-art classifiers.
3. We have successfully identified six WeChat message types from APNs traffic using a hash table lookup method.
4. Finally, the coping strategies of resistance analysis method were proposed.

1.2. Organization

The rest of this paper is organized as follows. We first generalize the related works in Section 2. Then, we describe the characteristics of the APNs in Section 3. In Section 4, we introduce our classification system, which includes the identification and analysis of APNs traffic. In Section 5, we evaluate our classification system. Finally, in Section 6, we draw conclusions and discuss how the proposed method can be extended.

2. Related Work

In this study, we mainly focus on identifying the message types of WeChat by using system's network traffic. Unlike traditional web traffic data from web browsers, most traffic from smartphones often implements end-to-end encryption designed to protect user privacy. Indeed, there have been a lot of works in recent years to analyze user actions from the network traffic of smartphones.

Coull and Dyer et al. [7] detected user actions, OS versions, and languages used in the Apple instant messaging application iMessage, which uses the APNs to deliver messages. However, they only analyzed traffic using the APNs to deliver messages instead of delivering notifications. Furthermore, they did not specify how to identify APNs traffic from background traffic. The traffic they classified is between the APNs server and iMessage maintained by Apple. In contrast, the APNs traffic we used in this study is generated only by iOS and the APNs server instead of any applications.

Wang et al. [2] divided the traffic received and sent it by the application into multiple sequences. Then, they extracted a total of 20 statistical values in the sequence as the characteristics using random forest (RF). The algorithm identified applications and the accuracy achieved exceeded 90%. However, they only collected 5 minutes of network traces as a dataset. Additionally, they did not give precision/recall measurements. Thus, it is difficult to further understand the performance of their classifiers.

Park and Kim et al. [3] researched 11 user actions supported by KakaoTalk, a Korean instant messaging application. They obtained a sequence of message features corresponding to each action's pattern. Afterwards, the features were used to identify the corresponding user action in unknown network traffic. This method can reach an accuracy of 99.7%, despite the fact that KakaoTalk traffic is encrypted. In detail, they used a limited Internet protocol (IP) address to tag KakaoTalk traffic. Considering the situation of server load balancing, many untagged IP addresses may appear. Thus, the tagging method did not have long-term stability.

Conti et al. [4] used IP addresses and TCP message header information to identify different operations performed by users in the same application. When users perform different operations in the same application, the characteristics of their outbound and inbound network flows are different. Since the sequence of outbound and inbound bytes in the TCP packet stream is used as the statistical features, it is suitable for encrypted traffic at the transport layer. However, that method becomes invalid when the traditional traffic is not generated.

Similar to [4], Fu et al. [5] proposed a system called CUMMA that recognized users' different operations within the same application. They first divided the collected traffic at two levels; namely, session and dialog. They selected the length and time interval as features in dialog units. Finally, experiments on both WeChat and WhatsApp applications have shown that the overall accuracy of

CUMMA systems can reach over 96%. This system needs to collect a large amount of traffic. Otherwise, the classification accuracy of the system may be reduced.

Shafiq et al. [6] first identified the traffic generated by WeChat users when sending texts and images. The experimental results show that C4.5 (an algorithm used to generate a decision tree) and support vector machine (SVM) achieved the best classification performance on the traffic generated by these actions, achieving 99.91% and 99.57% accuracy, respectively.

With the rapid increase of data size in network traffic classification, how to effectively select traffic characteristics has become a huge challenge. To solve this problem, Wang et al. [10] proposed an efficient network traffic feature selection method based on Spark, a new parallel computing framework. They first preprocessed the complete feature set based on Fisher scores, and then used a sequential forward search strategy for the subset. Then, successive iterations of the Spark computing framework were used to select the best feature subset. The results show that the method can reduce the time cost of modeling and classification and greatly improve the efficiency of feature selection while maintaining the accuracy of classification.

Sultan et al. [11] studied the operational efficiency of cellular networks and user action patterns from the call detail records (CDRs) of mobile networks. The author extracted a feasible solution from the CDRs data, which has strong temporal and spatial predictability in actual network traffic patterns. Based on this, they proposed a framework for mobile traffic classification. Experimental results show that the proposed model based on machine learning technology can accurately model and classify network traffic patterns.

More recently, In order to monitor people's privacy on mobile payment apps in three different manners by analyzing traditional traffic, Wang et al. [1] designed a hierarchical identification system. They used a series of well-performed ensemble learning strategies to deal with three identification tasks. The system can achieve an average 99% accuracy for application identification, 96% accuracy for user action identification, and 94% accuracy for step identification.

Our work differs from previous work on the following points: Other papers filter the background traffic to obtain the network traffic generated by the application and classify the message types accordingly. When the application is prohibited from using the traffic, the above methods will not work. Instead of using the traffic generated by the application, we use the APNs traffic. Our method makes up for the shortcomings in this scenario. Without filtering background traffic, we use the GNB algorithm to extract the APNs traffic from the background traffic and then analyze the WeChat message types carried by the APNs. This study pioneers the analysis of message types from remote message notification traffic.

APNs, as Apple's only official remote message notification service, provides users with powerful, secure, and efficient services that can spread information to indirect watchOS (iOS), tvOS, and macOS devices [12]. There are two types of message notifications for iOS, local message notifications, such as a to-do list notifications, and remote message notifications. Remote notifications for all offline apps on iOS must be delivered by the APNs. The APNs processes a remote notification of WeChat, as shown in case 1 in Figure 1. Case 2 is a schematic of WeChat accepting messages when the client is running in the foreground. Many existing technologies use the traffic generated by case 2 to identify the message type, while we utilize traffic generated by case 1 to identify the message types. The APNs traffic has the following characteristics.

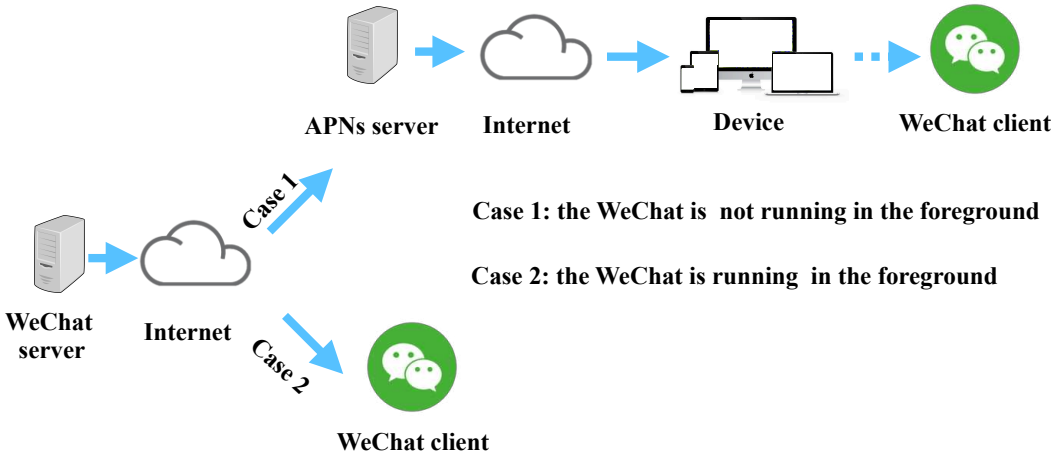

Figure 1. Transferring remote notification from WeChat's server to iOS.

3. Motivation to Deal with Challenges Using the APNs

- Each time iOS connects to the Internet, iOS establishes a persistent IP connection with the APNs server [13]. We can cluster all the packets based on this connection.
- APNs uses transport layer security (TLS) to prevent attackers from obtaining the content of a message. Thus, we cannot directly use the load of the packet. We use its side-channel information as an alternative; e.g., time interval and packet length.
- iOS uses port 5223 or 443 to communicate with the APNs server. The patterns of APNs traffic have no differences using these two different ports [7]. When the APNs uses port 5223, we can identify the traffic of the APNs based on this port number. When the APNs uses port 443, the port-based traffic classification method fails because 443 is a well-known port of the HTTPS protocol.
- The application on iOS has three states: running in the foreground, running in the background, and not running [14]. Different applications can use different remote message notification services when the application is in the foreground. For example, Apple's iMessage still uses the APNs. In contrast, WeChat uses its own remote message notification service. When the application is in the other states, only the APNs can be used to notify the user of new messages at the system level. In these two cases, we can only use APNs traffic to identify the message types.

In the following section, we introduce the classification system that leverages the aforementioned characteristics to identify APNs traffic easily and efficiently.

4. Proposed Classification System

In the preceding section, we introduced the main features of the APNs, which can remind users of the arrival of a message. Those types of message might be varied. In the case of WeChat, the message might be a text, voice message, video, red packet, transfer, picture, etc. In this section, we detail the five major modules of the proposed system for classifying message types in WeChat. Figure 2 illustrates the framework of the proposed method. (Download from http://mumu.163.com/).

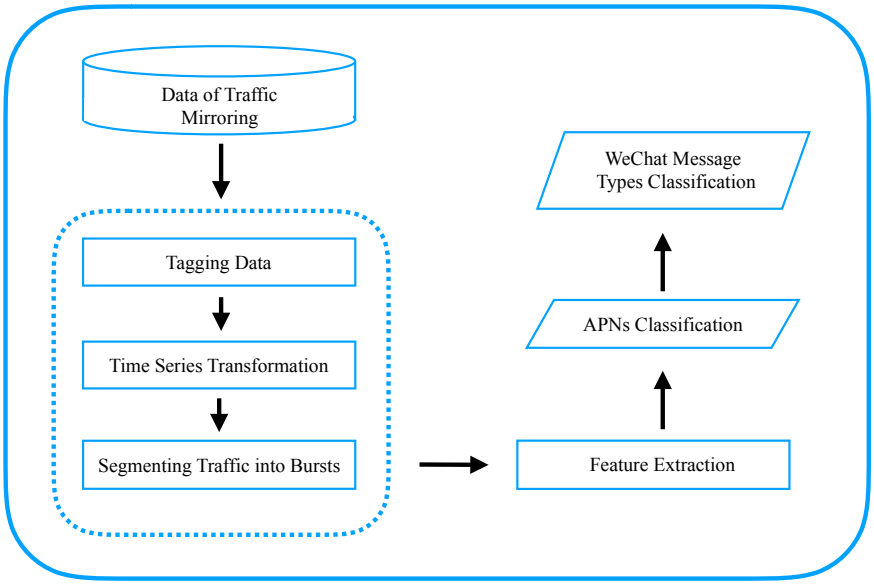

Figure 2. Framework of proposed method.

4.1. Traffic Collection

To test the effectiveness of the proposed method, we first designed a platform for collecting APNs network traffic generated by iOS, as shown in Figure 3. We collected all TCP traffic generated by iOS after the screen was locked (all applications are not running in the foreground). We created two accounts, Alice and Bob, in order to use the services provided by WeChat. Bob was used as a sender running on the Android system that ran on the Mumu emulator. Alice was used as receiver running on iOS that ran on the iPhone 8 Plus. The details are described in Table 1.

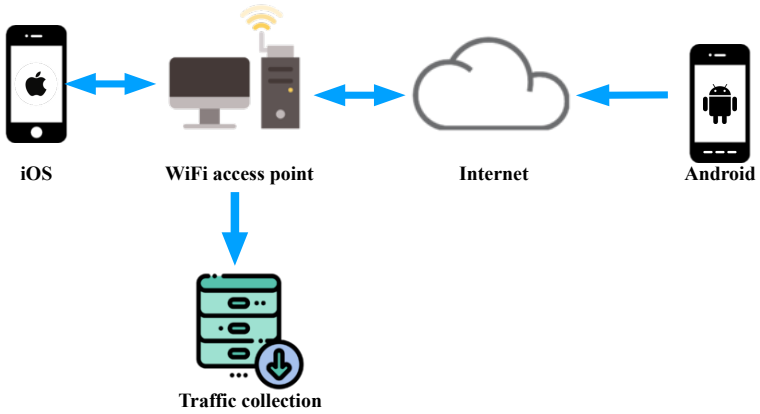

Figure 3. Proposed traffic collection platform for Apple Push Notification service (APNs).

Table 1. Information of sender and receiver.

Information	Sender	Receiver
Device	Mumu simulator [1]	iPhone 8 Plus
OS version	Android 4.4.4	iOS 11.3
WeChat version	6.6.6	6.7.1
Account name	Bob	Alice

[1] download from http://mumu.163.com/.

4.2. Traffic Segmentation

After the data was collected, we filtered out the packets of non-TCP traffic, and then only extracted the side-channel information (i.e., the timestamp and length of the TCP payload). Finally, the burst was segmented according to time interval.

1. **Tagging data**: To improve the performance of our approach, we compared the performance of our approach using different, widely used, supervised machine learning algorithms. Those supervised machine learning algorithms generally work in three steps: feature extraction, training, and testing. During the first step, all the bursts are used to extract features that are expected to be sensitive to the given labeled bursts. It should be noted that the extraction step is applied to all the bursts from the training set; that is, those for which the message type is known. During the training step, the classifier is established based on the feature vectors in the prior step. In the testing step, the extracted features from the testing set are fed into the trained classifier to complete the identification task. Thus, we labeled the data as APNs or non-APNs traffic. Since one of the two ports used by the APNs is a well-known port of the HTTPS protocol, we cannot distinguish APNs and non-APNs traffic using the well-known port. Therefore, we manually mark APNs and non-APNs traffic based on the time of message arrival.
2. **Time series transformation**: Data pre-processing is a data-mining technique that is used to transform the raw data into a useful and efficient format. If there is much irrelevant and redundant information, denoting the noisy and unreliable data, it is more difficult to extract the valuable information during the training phase. The traffic we collect contains encrypted information and some control fields of the IP protocol. The encrypted information and parts of control fields (flags, checksums, etc.) are approximately random strings. The random strings cannot help us distinguish APNs and background traffic, but still occupy storage space. Therefore, we leverage its side-channel information, i.e., packet length and timestamp information, to identify APNs traffic. In this way, we obtain the fixed pattern of APNs traffic, as shown in Figure 4. This is the APNs traffic that carries one WeChat text notification. The horizontal axis represents the time series, while the vertical axis represents the length of the TCP payload. The positive value represents the length of the packet received by iOS, while the negative value represents the length of the packet sent by iOS. The message types of WeChat are only related to the length of the first packet of APNs, and the length of the other four packets does not change. After the APNs server sends a 53 byte packet to iOS, iOS sends a 53 byte packet response. After about 5 s, iOS sends a 69 byte packet to the APNs server and the APNs server sends a 53 byte packet. At that point, a complete message transmission ends.

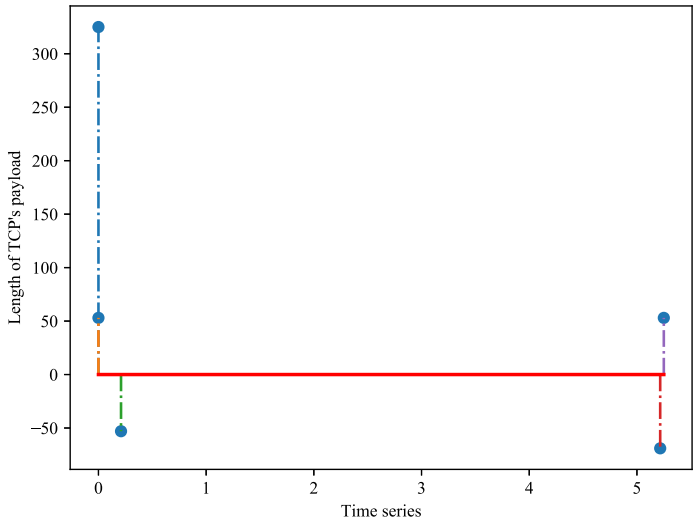

Figure 4. APNs traffic patterns.

3. **Segmenting traffic into bursts**: We define flow as a sequence of packets with a quintuple (source IP, destination IP, source port, destination port, and protocol) and session as the flow in both directions (i.e., exchange the destination and source). As mentioned in Section 3, the APNs is a persistent connection. Thus, we cluster all sessions according to their quintuple. After that, the session is divided into bursts. According to [15], each action produces a traffic burst. To measure the similarity between bursts, the first step is to segment flow into bursts. A burst can be defined as a set of consecutive packets and the time interval between adjacent packets as the time within the threshold of a period.

4.3. Feature Extraction

After dividing the traffic into bursts, we extract the features based on the length of the burst. To characterize the burst more simply, we extract only the first-order statistical features of each burst. The first-order descriptive statistics include sum, mean, standard deviation, maximum, and minimum. Compared with [16], we use fewer features to improve the efficiency of the proposed classifier.

4.4. APNs Traffic Identification

After extracting the features of each burst, we used the features of bursts to form feature vectors that were fed to the classifier. We also compared four machine learning methods presented in [16]; namely, k-nearest neighbors (KNN), SVM, RF, and Gaussian naive Bayes (GNB). According to the results in Figure 5, our method with the GNB algorithm outperforms the others.

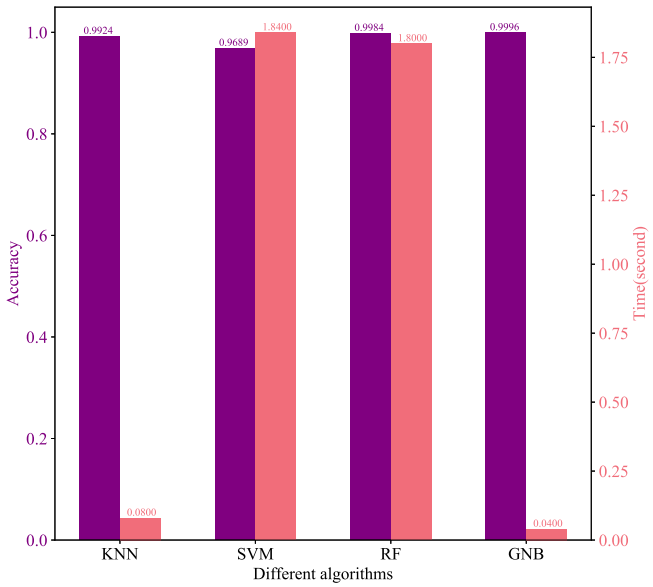

Figure 5. Performance comparison between different algorithms.

4.5. Message Type Identification

Each APNs notification contains an application message. After we successfully identify the APNs traffic, the next important step is to identify the types of messages carried by the APNs. We removed the traffic packets with lengths that were the same as each message. The length of the remaining packets can be used to distinguish the message type. As shown in Figure 6, the horizontal axis represents the number of each category of arrival packets, while the vertical axis represents the length of the TCP payload of arrival packets. Each packet length that appears in the figure represents a message notification sent via the APNs. In addition, the observed length in the training data in the figure can be used to nearly distinguish each category. Thus, we create a hash-based lookup table instead of using

machine learning methods of heavy computational complexity. The hash-based lookup table uses the length as a key and stores the associated class labels.

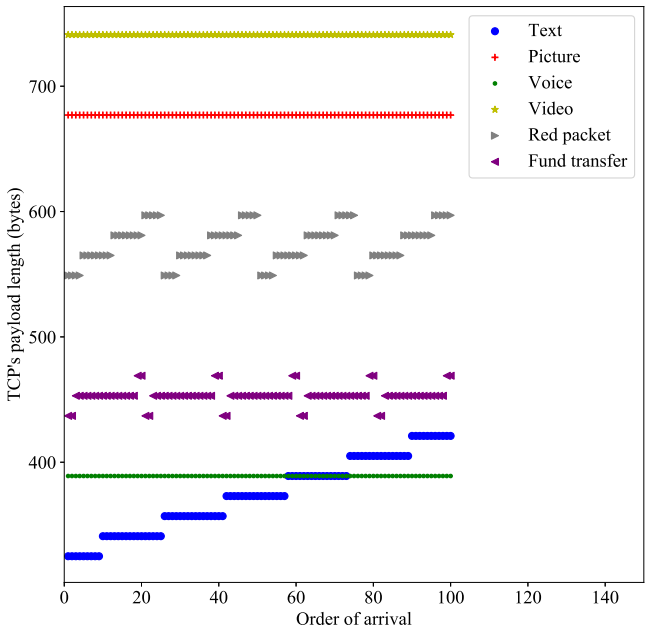

Figure 6. Length distribution of WeChat messages.

5. Evaluation

5.1. Data Description

As an instant messaging application, WeChat provides some of the most commonly used chat features, such as receiving and sending video, voice messages, texts, and pictures. In addition, the functions of fund transfer and red packet have been added. The frequent acceptance of red envelopes and fund transfers is likely to indicate the existence of fraud and gambling activities. Accurate identification of these two types of messages can prevent financial fraud and gambling activities. Thus, the identification of these six message types is widely analyzed and studied using traditional traffic [5,16]. In this study, we identified the traffic of these six message types from another perspective based on system traffic. We collect the notifications of the APNs for the six types of messages on WeChat. To verify the robustness of the proposed method, each type of message contains a variety of different messages. A summary of the diversity of the messages is shown in Table 2.

- Text: The text message of WeChat only contains the variable length of the text. We found that the length of the APNs's TCP segment increases staircase-wise as the length of text message increases from 1 to 91. In addition, when the length of the text increases from 92 to 16,354, the payload length of the segment does not change.
- Red packet: There are two variables in the red packet message: the first is the amount ranging from 0.01 to 200 RMB, and the second is the greeting that explains the purpose of the red packet.
- Fund transfer: Similar to red packet messages, the fund transfer message also has two variables: amount and note. The range of the amount is 0.01 to 200,000 RMB for daily transaction amount. Additionally, the note uses 1 to 20 letters to illustrate the purpose of the fund transfer.
- Picture: A picture in WeChat only contains the variable of image size.
- Video: Similarly to picture messages, video messages contain only one variable; namely, video length, which ranges from one to 10 s.
- Voice: The length of the voice ranges from one to 60 s.

Table 2. Dataset decription.

Message Types	Variable	Number of Messages
Text	length	100
Red packet	amount and greeting	100
Fund transfer	amount and note	100
Picture	size	100
Video	size	100
Voice	size	100

5.2. Evaluation Metrics

Before introducing the evaluation metrics, we introduce the following basic metrics. False negative (FN) error marks a positive sample as a negative sample. False positive (FP) error judges a negative sample as a positive sample. True negative (TN) correctly marks a negative sample as a negative sample. True positive (TP) correctly judges a positive sample as a positive sample. We evaluate the performance of the classifier using the following metrics:

- *Accuracy*: defined as the percentage of the number of instances correctly classified in all samples. It is defined by the following formula:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}. \tag{1}$$

- *Recall*: also called true positive rate or sensitivity in this context. It is given by

$$Recall = \frac{TP}{TP + FN}. \tag{2}$$

- *Precision*: also called positive predictive value, expressed as

$$Precision = \frac{TP}{TP + FP}. \tag{3}$$

- $F1$: takes into account both precision and recall, which is formulated by the following formula:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{4}$$

5.3. Selection of Burst Threshold

Before considering the classification for the message types, it is worth discussing how to choose the burst threshold of segmenting traffic. For determining the optimal burst threshold, we tested a set of values using all data. Furthermore, the results obtained by using 10-fold cross-validation are shown in Figure 7. The horizontal axis represents the burst threshold, while the vertical axis represents the accuracy of our approach using different burst thresholds. It can be seen that an accuracy of 99.85% can be achieved at 0.4 and gradually increases. Then, the accuracy reaches a maximum of 99.96% at 0.6, after which it decreases slowly. The reason is that different burst thresholds have different capabilities for obtaining traffic characteristics. Therefore, we chose 0.6 as the optimal value of the burst threshold.

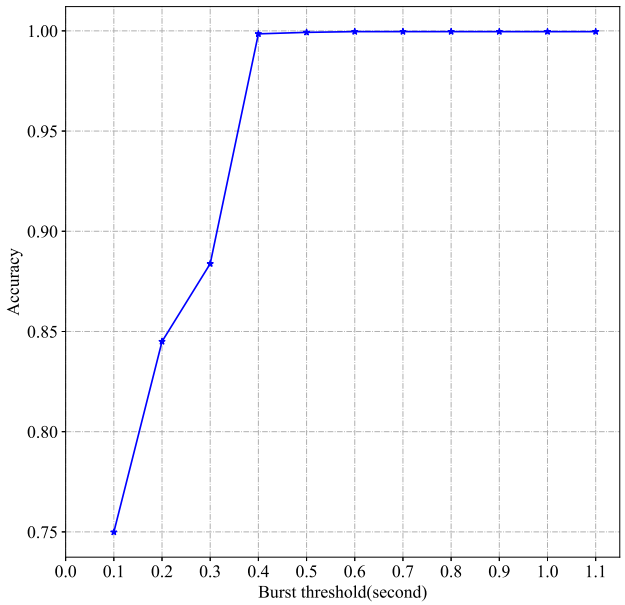

Figure 7. Effectiveness of detection accuracy with different burst thresholds.

5.4. Classification Performance

In this section, we evaluate the performance of our proposed method in terms of identifying the six WeChat message types using the traffic generated by the OS. In addition, we compared our method with the methods that use traditional traffic generated by WeChat.

Before comparing the proposed approach with that of Yan et al. [16], analyzing the performance and efficiency of different machine learning algorithms is helpful for choosing the optimal algorithm. We first analyze the performance of identifying APNs traffic. Furthermore, we compare the performances of these algorithms using different parameters. To achieve improved RF performance, the RF performance of different numbers of decision trees is compared. As Figure 8 illustrates, the highest accuracy is over 99.84% when the number of trees is five. To achieve improved KNN performance, the KNN performance of different number of neighbors was compared. As Figure 9 illustrates, the highest accuracy was over 99.24% when the number of neighbors was one or two. To achieve improved SVM performance, five widely adopted kernel functions were compared. However, the “linear” and “poly” kernels consumed too much time. The “precomputed” kernel needed extra input. Thus, the remaining “rbf” and “sigmoid” kernels were compared. The “rbf” kernel reached an accuracy of 96.89%, while the "sigmoid" kernel reached an accuracy of 54.49%. GNB is a member of simple “probabilistic classifiers” based on applying Bayes’ theorem. GNB assumes that the continuous data associated with each label is distributed according to a Gaussian distribution. In summary, for these machine learning algorithms, we set the number of clusters as two in KNN, the number of decision trees as five in RF, the kernel as “rbf” in SVM, and prior probabilities of the classes as none in GNB.

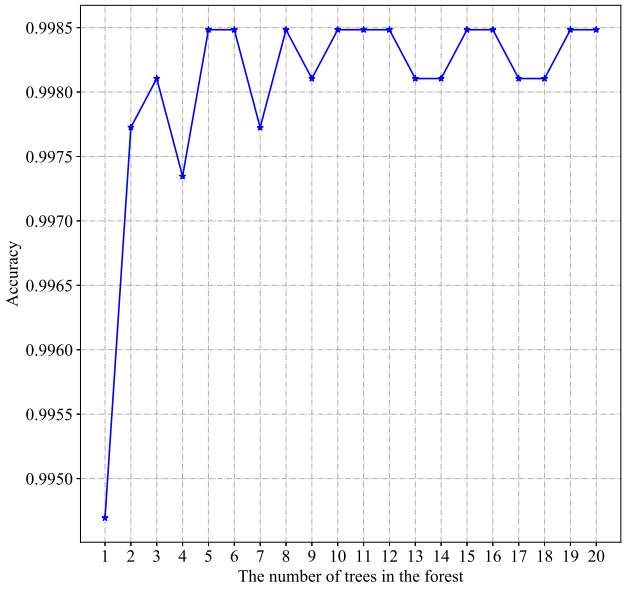

Figure 8. Performance comparison using different random forest (RF) parameters.

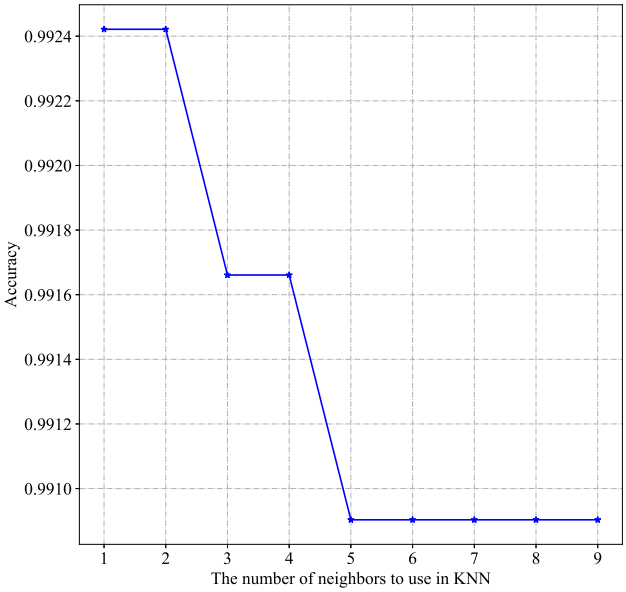

Figure 9. Performance comparison using different k-nearest neighbors (KNN) parameters.

Through the aforementioned analysis, we chose an optimal parameter for each algorithm. The results of the comparison are shown in Figure 5. The horizontal axis represents different classifiers, while the purple and red vertical axes represent accuracy and time, respectively. According to Figure 5, our approach using GNB can effectively identify APNs traffic in the shortest time (0.04 s) and with the highest accuracy (99.96%) when compared to RF, SVM, and KNN. The precision, recall, and F1 values of the classification results are shown in Table 3. According to the results, it is clear that our proposed method with the GNB algorithm is effective in identifying APNs traffic (i.e., accuracy of 99.96%, F1 of 99.96%, recall of 99.93%, and precision of 100%). The KNN classifier works best when the number of neighbors is $n = 2$, but the smaller n is, the more susceptible it is to noise. SVM is less efficient on noisy data sets with overlapping classes. RF is an integrated classifier. Multiple weak classifiers need to be trained, and the final result is determined by voting by the weak classifiers. RF also consumes a lot of time while achieving good performance. GNB first calculates the posterior for each class and then uses the class with the maximum posterior as the label that is called the maximum a posteriori estimation

decision rule. Furthermore, GNB can achieve better results than other classifiers with a small amount of training data because it has a low propensity to overfit. What is more, the GNB assumes that the attributes are independent of each other, the computational complexity is relatively small, and the classification efficiency is high. And the features we extracted are less relevant, which meets the premise of GNB.

Table 3. Results of APNs identification using different algorithms.

Algorithms	Accuracy	F1	Precision	Recall
KNN	0.9924	0.9930	0.9911	0.9951
SVM	0.9689	0.9683	1.0000	0.9430
RF	0.9984	0.9986	0.9979	0.9993
GNB	0.9996	0.9996	1.0000	0.9993

For comparison, we collected network traffic on WeChat receiving 600 messages. Different from Section 4.1, traffic was collected while WeChat was running; namely, traditional traffic. We used the C4.5 algorithm proposed by Shafiq et al. [6] and the AdaBoost algorithm proposed by Wang et al. [1] to extract the traditional traffic generated by WeChat from the background traffic. The results are shown in the Figure 10. As can be seen from the figure, our method extracts the APNs traffic generated by WeChat from the background traffic with higher accuracy, F1 value, and recall. The methods of Shafiq et al. [6] and Wang et al. [1] have lower accuracies, F1 values, and recalls. It can be seen that the background traffic has a greater impact on the performance of the two methods. Our method can well identify APNs traffic from background traffic. This is because APNs traffic has a fixed pattern and is easier to identify.

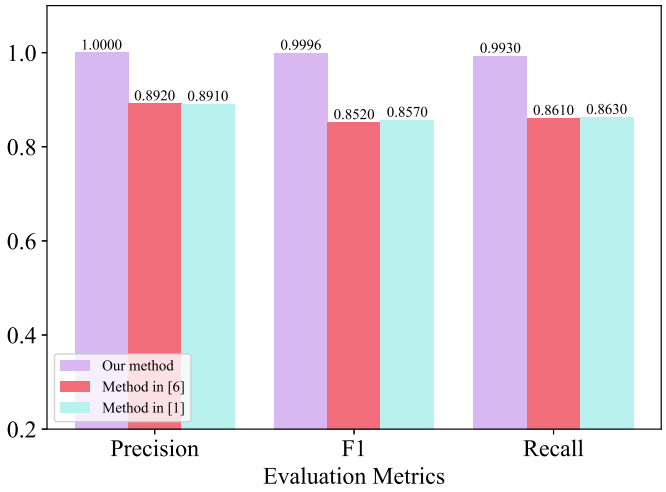

Figure 10. Performance comparison between our method and that of [1,6].

After identifying the APNs traffic, another important step is to identify the message types of WeChat from the APNs traffic generated by iOS. In detail, we first assumed that the early classification phase of the APNs is 100% accurate in order to focus on the classification of message types. The APNs traffic of messages of WeChat's voice is confused with APNs traffic generated by the text messages with lengths of 58 to 73. This led us to being unable to distinguish them correctly. For the sake of simplicity, the hash-based lookup table method mentioned in Section 4.5 is used directly instead of a computationally intensive machine learning algorithm.

In Figure 11, compared with the approach of Yan et al. [16], recognition evaluation indicators of actions of receiving the red packet and fund transfer are lower than in our method. This is because

the traffic patterns of the APNs have a more fixed mode compared to the patterns of traffic generated by WeChat.

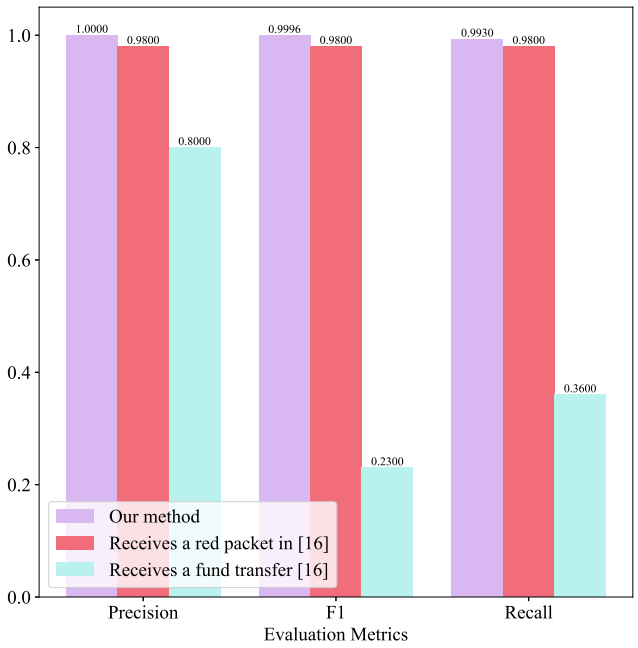

Figure 11. Performance comparison between proposed classifier and that of [16].

5.5. Efficiency

In the preceding section, we found that our proposed method performs better in F1, precision, and recall evaluation criteria than the method proposed by Yan et al. [16]. Now, we compare their efficiencies. As shown in Table 4, different from Yan et al. [16], in which a proportional division method was used, we used the 10-fold cross-validation method to divide our dataset. Owing to the absence of time consumed by time series transformation in Yan et al. [16], we considered the most ideal situation to be setting it to zero. Considering the aforementioned two situations, our overall efficiency performs better than Yan et al. [16].

Table 4. Computational performance.

Produces	Time (Seconds)	Time [16]
Time series transformation	1.48	/
Traffic segmentation (burst)	0.03	4.0
Feature extraction	1.70	2.06
10-fold cross validation (GNB)	0.04	/
Training and testing (RF)	/	0.22
Total	3.25	6.28

Time series transformation and feature extraction are the two most time-consuming stages. In the time series transformation stage, we used the dpkt tool (https://dpkt.readthedocs.io/en/latest/) to extract the length and timestamp of each packet. Meanwhile, we filtered packets with a payload size of zero. According to Section 5.3, we selected 0.6 s as the threshold of burst, and extracted only five features for each burst. Using fewer features helps reduce the computational complexity of the classifier. Additionally, choosing a smaller burst threshold allows the classifier to output results in a shorter amount of time. These advantages make our approach more efficient in real-time classification.

5.6. Countermeasures

In [17–20], authors proposed mutual authentication methods to prevent attackers from using fake identities to deceive devices, to obtain information. The traffic analysis method proposed in this study is to analyze network traffic on public channels. It is a passive listening method rather than an active attacking method. Therefore, there are two methods to deal with the network traffic analysis methods proposed in this study:

The first countermeasure is that Apple modifies the traffic patterns of APNs to make it more difficult to identify, so as to prevent eavesdroppers from analyzing APNs traffic. A simple way is to fill the contents of the APNs packets with arbitrary values or send redundant packets, so that the APNs packets cannot be identified using statistical features.

The second countermeasure is that WeChat fills the content of the message notification so that the length of each message notification is independent of the message types. Even if APNs traffic is recognized, eavesdroppers cannot know the message types of WeChat. A simple way is to use the maximum or arbitrary value padding.

6. Conclusions

In this study, we conducted an in-depth study of the APNs traffic of iOS. We successfully identified the message types of WeChat using APNs traffic. In detail, we first propose an approach with GNB to extract the traffic of the APNs from background traffic. Then, we use the hash-based lookup table method to identify the message types of each notification sent by the APNs. The experimental results show that our proposed approach using system traffic outperforms the previously state-of-the-art approaches using the traffic generated by the app. Furthermore, our method can analyze its message types when WeChat is closed or traffic is banned, which is the scenario that users are most likely to ignore. This opens a new way of identifying the message types of WeChat.

In the planned future studies, we intend to verify whether the APNs can be used to identify more message types and more applications. We can also use the methods presented in this paper to identify and analyze other remote message notification traffic generated by other OSs.

Author Contributions: Supervision, M.X. and N.Z.; Validation, Y.W.; Writing—original draft, Q.Z.; Writing—review & editing, T.Q.

Funding: This research was funded by the Cyberspace Security Major Program in the National Key Research and Development Plan of China under grant number 2016YFB0800201; the Natural Science Foundation of China under grant numbers 61702150 and 61803135; the State Key Program of Zhejiang Province Natural Science Foundation of China under grant number LZ15F020003; the Key Research and Development Plan Project of Zhejiang Province under grant number 2017C01065; and the Public Research Project of Zhejiang Province under grant number LGG19F020015. The APC was funded by the Cyberspace Security Major Program in the National Key Research and Development Plan of China under grant number 2016YFB0800201.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wang, Y.; Zheng, N.; Xu, M.; Qiao, T.; Zhang, Q.; Yan, F.; Xu, J. Hierarchical Identifier: Application to User Privacy Eavesdropping on Mobile Payment App. *Sensors* **2019**, *19*, 3052. [CrossRef] [PubMed]
2. Wang, Q.; Yahyavi, A.; Kemme, B.; He, W. I know what you did on your smartphone: Inferring app usage over encrypted data traffic. In Proceedings of the 2015 IEEE Conference on Communications and Network Security (CNS), Florence, Italy, 28–30 September 2015.
3. Park, K.; Kim, H. Encryption Is Not Enough: Inferring user activities on KakaoTalk with traffic analysis. In Proceedings of the International Workshop on Information Security Applications, Jeju Island, Korea, 20–22 August 2015; pp. 254–265.
4. Conti, M.; Mancini, L.V.; Spolaor, R.; Verde, N.V. Analyzing android encrypted network traffic to identify user actions. *IEEE Trans. Inf. Forensics Secur.* **2016**, *11*, 114–125. [CrossRef]

5. Fu, Y.; Hui, X.; Lu, X.; Jin, Y.; Chen, C. Service Usage Classification with Encrypted Internet Traffic in Mobile Messaging Apps. *IEEE Trans. Mob. Comput.* **2016**, *15*, 2851–2864. [CrossRef]
6. Shafiq, M.; Yu, X.; Laghari, A.A. WeChat Text Messages Service Flow Traffic Classification Using Machine Learning Technique. In Proceedings of the IEEE International Conference on IEEE International Conference on High-performance Computing & Communications, IEEE International Conference on Smart City, Sydney, NSW, Australia, 12–14 December 2016.
7. Coull, S.E.; Dyer, K.P. Traffic analysis of encrypted messaging services: Apple imessage and beyond. *ACM SIGCOMM Comput. Commun. Rev.* **2014**, *44*, 5–11. [CrossRef]
8. Conti, M.; Li, Q.Q.; Maragno, A.; Spolaor, R. The dark side (-channel) of mobile devices: A survey on network traffic analysis. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 2658–2713. [CrossRef]
9. Guo, W.; Liu, H. The analysis of push technology based on iphone operating system. In Proceedings of the 2013 2nd International Conference on Measurement, Information and Control, Harbin, China, 16–18 August 2013; Volume 1, pp. 570–574.
10. Wang, Y.; Ke, W.; Tao, X. A feature selection method for large-scale network traffic classification based on spark. *Information* **2016**, *7*, 6. [CrossRef]
11. Sultan, K.; Ali, H.; Ahmad, A.; Zhang, Z. Call Details Record Analysis: A Spatiotemporal Exploration toward Mobile Traffic Classification and Optimization. *Information* **2019**, *10*, 192. [CrossRef]
12. Gusgård, O. Application Development for the Apple Watch. Available online: https://www.theseus.fi/bitstream/handle/10024/147350/Gusgard_Thesis.pdf?sequence=1 (accessed on 25 December 2019).
13. Lee, D. Designing the multimedia push framework for mobile applications. *Int. J. Adv. Sci. Technol.* **2011**, *32*, 117–124.
14. Brüstel, J.; Preuss, T. A universal push service for mobile devices. In Proceedings of the 2012 Sixth International Conference on Complex, Intelligent, and Software Intensive Systems, Palermo, Italy, 4–6 July 2012; pp. 40–45.
15. Stöber, T.; Frank, M.; Schmitt, J.; Martinovic, I. Who do you sync you are? Smartphone fingerprinting via application behaviour. In Proceedings of the Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks, New York, NY, USA, 17–19 April 2013; pp. 7–12.
16. Yan, F.; Xu, M.; Qiao, T.; Wu, T.; Yang, X.; Zheng, N.; Choo, K.K.R. Identifying WeChat Red Packets and Fund Transfers Via Analyzing Encrypted Network Traffic. In Proceedings of the 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), New York, NY, USA, 1–3 August 2018; pp. 1426–1432.
17. Erdem, E.; Sandıkkaya, M.T. OTPaaS—One time password as a service. *IEEE Trans. Inf. Forensics Secur.* **2018**, *14*, 743–756. [CrossRef]
18. Wang, D.; Wang, P. Two birds with one stone: Two-factor authentication with security beyond conventional bound. *IEEE Trans. Dependable Secur. Comput.* **2016**, *15*, 708–722. [CrossRef]
19. Wang, D.; Cheng, H.; He, D.; Wang, P. On the challenges in designing identity-based privacy-preserving authentication schemes for mobile devices. *IEEE Syst. J.* **2016**, *12*, 916–925. [CrossRef]
20. Jiang, Q.; Qian, Y.; Ma, J.; Ma, X.; Cheng, Q.; Wei, F. User centric three-factor authentication protocol for cloud-assisted wearable devices. *Int. J. Commun. Syst.* **2019**, *32*, e3900. [CrossRef]