Feature Article

Traffic Analysis Attack for Identifying Users' Online Activities

Firdous Kausar
Sultan Qaboos University

Sarah Aljumah
Al Imam Mohammad Ibn Saud Islamic University

Shorouq Alzaydi
Al Imam Mohammad Ibn Saud Islamic University

Raghad Alroba
Al Imam Mohammad Ibn Saud Islamic University

***Abstract*—Traffic analysis is a serious threat over the network. An attacker can analyze network traffic patterns to infer packet's content, even though it is encrypted. This article demonstrates a traffic analysis attack that exploits vulnerabilities in encrypted smartphone communications to infer the web pages being visited by a user.**

This paper presents the traffic analysis problem and demonstrates one of its possible attacks to exploit vulnerabilities in the paradigms of network communications in smartphones.

Smartphones have become an essential go-to source to perform extensive activities, such as navigating, conducting transactions, socializing, browsing the internet, and reading/sending e-mail. While these smartphones offer internet connectivity, they carry security risks, and smartphones security has become increasingly important. A common approach to protect users' data is to encrypt the traffic. Although it helps in protecting users' data from being logged, it does not provide complete security.

Digital Object Identifier 10.1109/MITP.2018.2876988
Date of current version 27 March 2019.

The nature of web applications makes it vulnerable to attacks. Web applications are split into two parts: client side and server side. With this structure web, applications require the exchange of packets over the network. The flow of the exchanged packets is exposed on the network and can be intercepted via a sniffer.

When the network traffic is not encrypted, it is easy to identify and classify the traffic by simply reading the packet's content. However, when the traffic is encrypted, the task is not easy, but it is still feasible. An attacker can take advantage of side-channel leaks to find network traffic patterns and infer packet content despite that it is encrypted with the state-of-the-art cryptographic techniques.

The traffic features that contain information about the secret payload such as packet lengths, numbers, and timing are visible and can be

1520-9202 © 2018 IEEE

extracted by traffic monitoring. These features can give unique characteristics to the traffic.

Side channel leaks of web traffic have been known for a while. Chen *et al.*[1] demonstrate that a patient health record, a company's investment secrets, and a user search queries are leaked from the top of the line applications, even when the traffic is encrypted by hypertext transfer protocol secure (HTTPs) or wireless application protocol (WAP/WAP2). In practice, traffic analysis is used by network administrators to track malicious activities. Traffic analysis was also used in web traffic fingerprinting as shown by Cai *et al.*[2–5] to infer typed search queries[8,9] and infer phrases in voice over Internet Protocol calls.[10,11]

In the past, it was possible to reliably identify sites/mobile applications through destination IP addresses and hostname. Currently, this technique does not work for several reasons. For example, nowadays, sites and mobile application are built on the top of content delivery networks, and it is not reliable to identify mobile apps/sites using destination IPs. Several services could be directing their traffic to the same servers. Furthermore, many services utilize cloud, which makes use of load balancers, making it difficult to map backend servers IPs with the service.[6,7] In addition, techniques that rely on ports, deep packet inspection, IP address, and server name indication can be bypassed. Those fields can be easily altered to frustrate any attempt to identify the traffic. Last but not least, our goal is to identify specific web pages, and it is not possible to identify them by the use of the only IP/hostname.

CONTRIBUTIONS

In this article, we propose a traffic analysis attack to infer which web page the user visited on his smartphone, by only looking at the network traffic generated from the smartphone. We assume the traffic is encrypted and the adversary eavesdrops the packets exchanged between the user's device and the application server. The proposed attack makes use of only timing information since its effectiveness and its strong relation to web fetches were proven by Miskovic *et al.*[6,7]; we discuss this relation in the next section. We perform an *N*-fold cross validation to evaluate our method, which achieves an accuracy of 96%. We collected a dataset that contains 10 000 labeled webpages streams of 100 webpages. The streams were downloaded under real-world network conditions.

ORGANIZATION

This paper is organized as follows: "RELATED WORK" section discusses the state-of-the-art work in the area of traffic analysis techniques and machine learning algorithms. The "A TIMING ATTACK" section shows background detail of a timing attack, and it discusses the network and attack model with attack scenario. The "PROPOSED ATTACK METHOD" section presents the details of the attack. The "PERFORMANCE EVALUATION" section discusses the analysis and results of the proposed attack, and the "CONCLUSION" section concludes with a summary and some proposed future work.

RELATED WORK

In this article, we prove that traffic analysis techniques and machine learning algorithms can be used to reveal extremely private information. In the literature, several works have shown that encryptions are not sufficient to protect users' data.

Schaub *et al.*[9] focused on the autocomplete functionality in search engines to be able to retrieve a user search query using side-channel leaks in search engines such as Google and Being. They analyzed packet lengths associated with every character. A query, therefore, can be inferred by comparing the intercepted packets with a set of precomputed queries.

Subsequently, Miller *et al.*[13] present a traffic analysis attack against HTTPs to identify individual pages on the same website. They identify individual webpages with 90% accuracy. Liu *et al.*[14] consider website fingerprinting. They improve website fingerprinting by utilizing packet ordering information.

In encrypted communications, three parameters can be observed: packet size, directions of packets, and the times at which individual packets' depart and arrive.

The use of features depends on the type of the extracted information and the analyzed protocol. In most cases, the features like packet sizes are used to infer hidden information that is related to the application layer due to its strong relation to packet content. In an attack that uses packet sizes, an eavesdropper can infer the content of the plaintext using the relationship between the packet size and the size of the plaintext. In contrast, Transmission Control Protocol (TCP) timestamps can be used to reveal network-layer-related information such as source and destination.

To overcome traffic analysis methods, several techniques were used to reduce the correlation between features and content. For instance, padding packets to hide size information. Most of the previous works are mainly based on the packet size and the directions of the packets. Jabber *et al.* [15] prove that interpacket time is valuable information for internet traffic classification. They propose a solution to distinguish between the network delays and the application time. They show that the interpacket time increases application classification accuracy from 80% to 98%.

Also, Feghhi and Leith[16] demonstrate an attack against encrypted traffic using timing information alone on the uplink. They measure the distance between packet sequences. The difference between two packet streams associated with fetches of the same web page at different times is small; however, when streams are associated with different web pages, the difference is large. They have used a variant of dynamic time warping (DTW), which aims to be insensitive to time differences that are related to compressing of time. They achieved an accuracy exceeding 90%.

None of the above-mentioned works was designed for mobile devices. Recently, traffic analysis has been used to detect information leaks in mobile devices. Coull and Dyer's[17] analysis is focused on the Apple iMessage service. They show that it is possible to identify fine-grained users' actions in smartphones. They consider packets payload lengths.

Also, Conti *et al.*[18] propose a framework that identifies user actions on Android apps via traffic analysis. Framework analyzes the network communications, packet information in TCP/IP (like ports, IP addresses), together with packet lengths, timing, and directions (incoming and outgoing). They consider three applications for their study: Facebook, Gmail, and Twitter. Their approach is based on supervised machine learning.

Similarly, Salta Formaggio *et al.*[19] build a tool that detects a user activity within smartphone apps. This tool builds models for activities based on their traffic behavior (e.g., packet exchanges, transfer rates, and data movement). They use an unsupervised machine learning approach using k-means clustering. After learning about traffic behavior for each activity, this tool is able to infer user activities within the app. In our study, we take advantage of time feature, which is the first work that presents a passive attack in smartphones by only using time feature.

A TIMING ATTACK

In a timing attack, timing features are observed, and those features vary depending upon the events that occur in web fetches. Furthermore, events that occur in fetching a web page are triggered by many factors, for instance, number of objects in a page, and whether those objects are located on a different server resulting in a new connection between client and server. For each connection, traces of packets can have different timings, which allow a connection to be identified.

In case of a dynamic content, a number of dynamic pages that can be displayed per second are much less than the number of static pages. Moreover, dynamic pages execute code on the server. As a consequence, these events affect packet timing and form timing signatures that can identify web pages.[16]

Network Model

The proposed model shown in Figure 1 describes how the attacker would be able to monitor the network traffic. The attacker system must install and configure a server that assists in routing the victim's traffic. When the victim smartphone establishes a Wi-Fi connection to perform actions over the network, the server must be up and running. This server is used to route the traffic from the access point to the wide area network, and *vice versa*. The victim's traffic must be flowing through the server in order to perform network eavesdropping. Now, the attacker can capture the traffic using sniffing tools, and then detect and analyze the traffic.

Attack Model

This paper considers a side-channel attack model. This attack depends on the weakness in

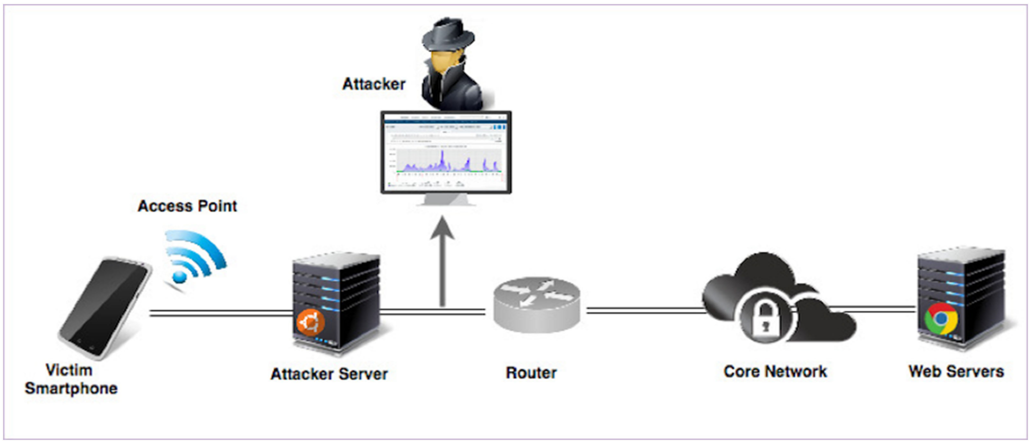

Figure 1. Attack network model; attacker resides between victim smartphone and the first router to monitor the encrypted network traffic.

the physical implementation rather than the theoretical weakness in a system. Attacker, in our case, would take advantage of leaks in encrypted network traffic caused by the application-specific implementation. Attack considers an attack model shown in Figure 2 that consists of a passive attacker running a sniffer to record the network traffic and all the network traffic is secured via the use of Secure Sockets Layer (SSL) and Transport Layer Security (TLS). The attacker has no previous knowledge about the activities of the target. Furthermore, the attacker is very weak and can only observe timestamps and transmission rate. The attacker intercepts the transmitted packets and uses the available information in the encrypted traffic to create a traffic signature. The attacker's goal is to infer the web pages being visited.

PROPOSED ATTACK METHOD

The attacker needs to prepare for the attack because the network conditions are not stable and change severely. It is not possible to rely on one trace for identification of a web page. The attacker needs to collect huge data to find the common features in all traces. The attacker gets this data by building crawlers (bots) that crawl data. The attacker is then ready to use the crawled data as training data that are fed to a classifier. The classifier will then be able to classify traces that have never been seen. The next sections explain in detail how each part is done.

Building a Web Pages Crawler

The first step in the preparation of the attack is to build the crawlers. A crawler is a bot that can perform actions in an automated manner. The bot can handle unexpected events such as pop-up windows or adds. The crawler can log in and enter passwords, record the analysis data, and capture the traffic. Most importantly, the crawlers can handle the attack's extensive use. The proposed attack performed heavy actions to log data.

Sniffing the Traffic

We propose the network model for the attacker to be able to sniff the traffic in a timing attack section. The attacker's server passes the traffic from the access point to the wide area network. The attacker's server also runs the crawlers, as well as the sniffer.

The sniffer is controlled by the crawler and they run in parallel. The crawler saves the traffic in packet capture file format (.pacp) and passes it to the next module.

Parsing Packets and Feature Extraction

In this stage, all the captured PCAP files are parsed. The purpose of parsing the PCAP files is to extract information that is of our interest. The extracted information includes packets' timestamps and packets' directions. The extracted information is then encoded into features. All packets are parsed and nothing is skipped. We extract all packets' timestamps and store them in a CSV file format and pass them to the next module.

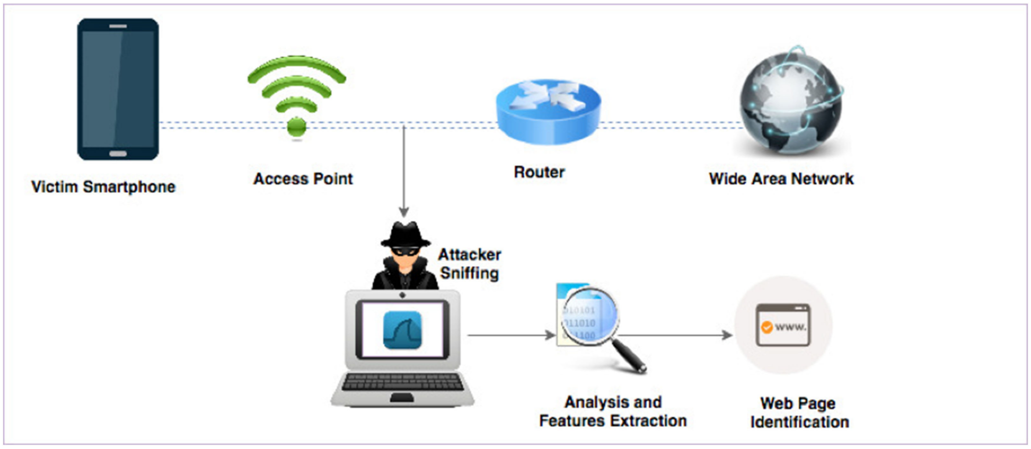

Figure 2. Attack scenario; attack will be performed when the attacker eavesdrops the encrypted network traffic and uses the available packet information to create traffic signature, which helps the attacker to identify the visit web pages.

Classification

In this section, we show the proposed methods for recognizing the encrypted network traces. This module gets features transferred from the previous feature extractor module. The classifier trains data for recognizing new traces that have never been seen before. Web pages are classified using a K-nearest neighbor algorithm (KNN). The KNN algorithm uses DTW to measure the distance between two traces. In the next sections, we explain the KNN approach, similarity measure by DTW, and how to break ties.

K-Nearest Neighbor Classifier The K-NN approach is used for the web pages' classification. The KNN depends on a similarity function to measure the distance between the two traces. The classifier stores all the considered cases. Then, it classifies new cases based on the similarity measure. A new case is assigned to the class the majority voted by its nearest *K* neighbors.

Dynamic Time Warping We use DTW for measuring similarities. DTW is a module that is used to measure the distance between two time traces. This module is insensitive to packet loss, retransmissions, compression/stretching of time caused by queuing delays, and other network conditions. The distance between two time traces of the same web page is measured as small. When two traces are from different web pages, distance is measured to be large.

Breaking Ties Ties occur when two traces give the minimum distance. When ties occur, we break them at random. However, this is a rare case since our DTW produces numbers with five digits after decimal number. Breaking ties at random is used only when performing cross validation. When performing the attack, all classes with the minimum distance are returned to the attacker.

PERFORMANCE EVALUATION

Data Acquisition

Appium framework is used to collect huge amount of data for the training purpose. Appium allows us to access mobile applications automatically and performs actions within mobile applications, e.g., opening a specific web page using a Google Chrome browser. When the crawler starts running, each web page is fetched 100 times. The crawler closes a web page once all elements are loaded.

Sniffing

Tshark is used to sniff the network traffic. Tshark sniffer is controlled by the crawler. The sniffer runs in parallel with the crawler to capture the exchanged packets. Since Tshark enables saving the captured packets, we capture 100 trace sample for each web page, then export these packets as packet capture extension (.pcap). As a result, we get 100 PCAP file for each web page: in total, 10 000 PCAP files.

Table 1. The experimental results.

Folds	Value of *K*		
	1	3	5
3	96%	94%	90%

Hardware and Network Configuration

We use HTC One smartphone, running Android operating system version 5.0.2 to prepare and perform the attack. We provide Wi-Fi access point connectivity to the smartphone using Hostapd. Also, we use the Dnsmasq server to provide network infrastructure that serves our network. Finally, we use a Lenovo E41-80 laptop with an Intel Core i5 processor and 4 GB RAM. It is running a Ubuntu Linux 16.04 operating system. The smartphone can be connected to any Wi-Fi connection. However, the Wi-Fi to which the smartphone is connected must be connected to the server that the attacker is monitoring. The server must be up and running to route the traffic from the access point to the Internet.

Diagnostics

We perform an *N*-fold cross validation, which is a technique used for accuracy estimation, to evaluate the effectiveness of the method. It portions the dataset into *N* subsets (fold), fits a model using $N-1$ folds, and predicts the performance using the fold that was left out. The cross-validation process is repeated on all possible folds. The results from the folds are averaged to estimate the accuracy.

Dataset

We collected a dataset of 10 000 encrypted network traffic. The streams are collected under real network conditions; web pages are randomly collected using Amazon, Alexa, and top Google hits. In this study, we decide to use the Chrome browser since it is a native application in the Android operating system, and it is one of the most widely used browsers.

EXPERIMENTAL RESULTS

Accuracy of the proposed method is measured as the number of times the classifier predicted traffic class i, and it was accurate. We first set *N* to be equal to three. In each test, we use a thousand traces for testing data and 9000 traces as training data. We show the experiment with respect to *K* in the KNN classifier. We first set *K* to be equal to one, three, and five. For each experiment, we get different results, as shown in Table 1.

Results Analysis and Discussions

The obtained results prove the effectiveness of the attack and prove that timing only attacks work on encrypted network traffic. The value of *K* indeed changes accuracy; if we look at the top ten votes, it shows that the correct answer always appear at the beginning, and when it does not, the answer does not show at all in the top ten, and if it ever shows, it is between indices 6–10 and not before that; therefore, the classifier will never vote for it no matter what is the value of *K*. To this point, we conclude that *K* set to 1 gives the best accuracy, without the consequence of losing the correct answer since it will never be voted for, even if we set *K* equal to 10. The achieved accuracy of the proposed method is 96%. What makes it such a high accuracy is that applications are implemented differently. These differences in implementation create application-specific traffic behavior.

CONCLUSION

This paper presents a traffic analysis attack to find users' online smartphone activities. We demonstrate an attack model that exploits SSL/TLS vulnerabilities to identify visited web pages. Our presented traffic analysis attack is successful against any encrypted tunnel. The attack is performed using only the visible features of encrypted packets exchanged over the network. Since web applications use client–server architecture, the exchanged packets through the network can be exposed. Passively, an attacker finds traffic patterns by intercepting the flow of the exchange packets. The attack uses timing feature to infer the visited web pages. The attacker makes use of the KNN classifier to classify a trace. A huge amount of data is used to train the classifier. The proposed method uses the *N*-fold cross-validation evaluation technique and achieves the accuracy of 96%.

■ REFERENCES

1. S. Chen, R. Wang, X. Wang, and K. Zhang, "Side-channel leaks in web application: A reality today, challenge tomorrow," in *Proc. IEEE Symp. Secur. Privacy*, Oakland, CA, USA, 2010, pp. 191–206.

2. X. Cai, X. C. Zhang, B. Joshi, and R. Johnson, "Touching from a distance: Website fingerprinting attacks and defenses," in *Proc. ACM Conf. Comput. Commun. Secur.*, New York, NY, USA, 2012, pp. 605–616.
3. J. Hayes and G. Danezis, "k-fingerprinting: A robust scalable website fingerprinting technique," *In USENIX Security Symposium. USENIX Association,* 2016.
4. X. Luo, P. Zhou, E. Chan, W. Lee, W. Chang, and R. Perdisci, "HTTPOS: Sealing information leaks with browser-side obfuscation of encrypted flows," in *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, San Diego, USA, Feb. 2011, pp. 1–20.
5. A. Hintz, "Fingerprinting websites using traffic analysis," in *Proc. 2nd Int. Conf. Privacy Enhancing Technol.*, 2002, pp. 171–178.
6. S. Miskovic, G. M. Lee, Y. Liao, and M. Baldi, "AppPrint: Automatic fingerprinting of mobile applications in network traffic," in *Proc. Int. Conf. Passive Active Netw. Meas.*, 2015, pp. 57–69.
7. V. Taylor, R. Spolaor, and M. Conti, "Robust smartphone app identification via encrypted network traffic analysis," *IEEE Transactions on Information Forensics and Security,* vol. 13, no 1, pp. 63–78, Jan. 2018.
8. S. Sharma and B. Menezes, "Implementing side-channel attacks on suggest boxes in web applications," in *Proc. 1st Int. Conf. Secur. Internet Things*, New York, NY, USA, 2012, pp. 57–62.
9. A. Schaub *et al.*, "Attacking suggest boxes in web applications over HTTPS using side-channel stochastic algorithms," in *Proc. Conf. Risks Secur. Internet Syst.*, Torento, Italy, 2015, pp. 116–130.
10. C. Wright, L. Ballard, S. Coulls, F. Monrose, and G. Masson, "Spot me if you can: Recovering spoken phrases in encrypted VoIP conversations," in *Proc. IEEE Symp. Secur. Privacy*, 2008, pp. 35–49.
11. A. White, A. Matthews, K. Snow, and F. Monrose, "Phonotactic reconstruction of encrypted VoIP conversations: Hookt on foniks," in *Proc. IEEE Symp. Secur. Privacy*, 2011, pp. 3–18.
12. A. Iacovazzi., A. Baiocchi, and L. Bettini, "What are you Googling? - Inferring search type information through a statistical classifier," in *Proc. IEEE Global Commun. Conf.*, 2003, pp. 747–753.
13. B. Miller, L. Huang, A. Joseph, and J. Tygar, "I know why you went to the clinic: Risks and realization of HTTPS traffic analysis," in *Proc. 14th Int. Symp. Privacy Enhancing Technol.*, 2014, pp. 143–163.
14. F. Zhang, W. He, X Liu, and P. Bridges, "Inferring users' online activities through traffic analysis" in *Proc. 4th ACM Conf. Wireless Netw. Secur.*, Hamburg, Germany, 2011, pp. 59–70.
15. M. Jaber, R. G. Cascella, and C. Barakat, "Can we trust the inter-packet time for traffic classification?" in *Proc. IEEE Int. Conf. Commun.*, Jun. 2011, pp. 1–5.
16. S. Feghhi and D. Leith, "A first-hop traffic analysis attack against femtocell," in *Proc. 16th IEEE Annu. Consum. Commun. Netw. Conf.*, 2016, pp. 1060–1065.
17. S. Coull and K. Dyer, "Traffic analysis of encrypted messaging services: Apple imessage and beyond," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 5, pp. 5–11, 2014.
18. M. Conti, L. Mancini, R. Spolaor, and N. Verde, "Can't you hear me knocking: Identification of user actions on android apps via traffic analysis," in *Proc. 5th ACM Conf. Data Appl. Secur. Privacy*, San Antonio, TX, USA, 2015, pp. 297–304.
19. B. Saltaformaggio *et al.*, "Eavesdropping on fine-grained user activities within smartphone apps over encrypted network traffic," in *Proc. 10th USENIX Workshop Offensive Technol.*, Austin, TX, USA, 2016, pp. 69–78.

Firdous Kausar is an Assistant Professor with the Electrical and Computer Engineering Department, College of Engineering, Sultan Qaboos University, Muscat, Oman. She has a number of publications in refereed international conferences and journals. Her research interests include cryptography, cryptanalysis, information security management, ubiquitous computing, network security, digital forensics, and Internet of Things. She has co-organized several international workshops and conferences, served on many technical program committees, and reviewed papers for several journals, conferences, and workshops. She received the Ph.D. degree in information security from the National University of Sciences and Technology, Islamabad, Pakistan, in 2009. Contact her at firdous.imam@gmail.com.

Sarah Aljumah is a Software Engineer with Al Elm Information Security Company. Her current research interests include big data analytics and machine learning. She received the Bachelor's degree in computer science from Al Imam Mohammad Ibn Saud Islamic University, in 2017. Contact her at sarahaljuma@gmail.com.

Shorouq Alzaydi received the Bachelor's degree in computer science from Al Imam Mohammad Ibn Saud Islamic University, Department of Computer Science. Her research interest focuses on information security. Contact her at shorouq.alzaydi@gmail.com.

Raghad Alroba received the Bachelor's degree in computer science from Al Imam Mohammad Ibn Saud Islamic University. Her research interests include encrypted networks and traffic analysis. Contact her at raghadroba@hotmail.com.