2016 Fourth International Symposium on Computing and Networking

Service Identification by Packet Inspection based on N-grams in Multiple Connections

Masaki Hara, Shinnosuke Nirasawa, Saneyasu Yamaguchi
Electrical Engineering and Electronics, Kogakuin University Graduate School
Tokyo, Japan

Masato Oguchi
Department of Information Sciences
Ochanomizu University
Tokyo, Japan

Akihiro Nakao,Shu Yamamoto
Interfaculty Initiative in Information Studies
Graduate School of Interdisciplinary Information Studies
The University of Tokyo
Tokyo, Japan

***Abstract*—Identifying the service of traffic by given IP network flows is essential for various purposes, such as management of QoS and avoiding security issues. Typical methods for this are identification based on its IP addresses and port numbers. However, the achieved accuracies of these method are not sufficient, then improving these methods is required. Deep Packet Inspection (DPI) is one of the most effective methods for improving accuracy of identification. In this paper, we explore a method for identifying the service of flow. We propose an identifying method based on DPI which covers multiple connections in a service. Then, we present performance evaluation and demonstrate that our method can suitably identify service from given network flows.**

***Keywords*—Service indetification; DPI (deep packet inspection); n-gram; HTTPS; DPN**

I. Introduction

In the current Internet, various types of services, such as web search and video streaming, are provided. Service identification from Internet flows is important for various reasons, such as management of QoS and avoiding security issues. Identifications using IP addresses and port numbers are the basic methods [1]. However, these basic methods cannot provide enough accuracy in current Internet. In some cases, many services are provided by a site with the same IP address. In some cases, dynamic port control [2] may be utilized. Thus, identification based on only this information has limitation. Inspecting packet payloads is one of the most promising methods for increasing accuracy. The work [3] have demonstrated that Deep Packet Inspection (DPI) have increased accuracy of identification. Because many of recent communications are encrypted, analyzing encrypted packets is required. The method [3] inspects encrypted packets using machine learning and achieved sufficient accuracy, but the method needs five days for learning. Thus, developing a method which can identify services in short time is important.

In this paper, we focus on service identification by inspecting packets in encrypted communication using HTTPS, and propose a method which can immediately identify services without using IP address. We define *service* as a function provided by a site. It is almost the same meaning with *service* of *application service provider* (ASP). For example, Gmail is a service. Gmail, Google Web search, and YouTube are different services independent to providing hosts, as described in Section V. Our method analyzes packets for establishing a TLS session, and creates *n*-gram frequency database. Then, the method groups sessions according to *n*-gram frequencies. It identifies the service based on the number of groups contained in the flows. We present our evaluation and demonstrate that our method can identify services with high accuracy without using information on IP addresses and port numbers.

The rest of this paper is structured as follows. Section II introduces related works on service and application identification, DPI, DPN, and *n*-gram. Section III explains establishment of TLS session. Section IV proposes our method. Section V evaluated our method. Section VI presents discussion. Finally, Section VII concludes this work.

II. RELATED WORK

A. Service and Application Identification

In many cases, service type and port number have strong relation [1]. Identification using port number is the most basic and effective way for identification. However, the achieved accuracy is not enough, and it can be improved [3]. In addition, in some cases, such as using NAT or reverse proxy, this method does not work well [2]. In some cases, a site provides several services with the same IP address, and its IP address changes dynamically. In many cases, many services are provided via Web, and all the port numbers are 80 or 443, and a service cannot be identified from its port number.

The works extracting information from encrypted traffic are kindly introduced in a survey by Velan et al. [4]. They argued that almost all network protocols ensure secure data transfer by means of encryption containing an unencrypted initialization phase and these data can be easily extracted and used for monitoring network traffic. HTTP Client Fingerprinting Using SSL Handshake Analysis by SSL LABS proposed to fingerprint client based on the SSL/TLS initial handshake [5]. p0f [6] extracted information of

2379-1896/16 $31.00 © 2016 IEEE
DOI 10.1109/CANDAR.2016.83

systems of peers, such as the operating systems user agents, from traffic by monitoring. Holz et al. presented a comprehensive analysis of X.509 certificates in traffic [7]. Their analyses revealed passive monitoring certificates could identify communication peers. Moreover, they argued that X.509 certification infrastructure was in a sorry state. Husák et al. proposed real-time lightweight client identification based on network monitoring and TLS fingerprinting [8]. Unlike our work, these works do not discuss identification based multiple connection. Thus, we expect that these methods cannot suitably identify a service containing commonly used certification sessions among several services. In addition, these works are mainly for identification of clients and do not identify services from traffic. The objectives of these works and ours are different.

Iwai et al. proposed a method for identifying using machine learning [9]. The method has achieved 82 % accuracy. However, it requires five days for learning, and reducing preparing time is an important issue. In work [10], a fast identifying method without machine learning was proposed. It analyzes packet payloads and create *n*-gram database. However, the former [9] and latter [10] methods identify only application and connecting sites, respectively. These cannot identify its service.

In addition, these studies do not take account of multiple connections in a service access but discusses only a single connection. Thus, the accuracy of identification is limited.

B. DPI and DPN

DPI is a method for filtering packets. The method analyses the payload of a packet for detecting spam, malicious software or other attacks. Then, the network element doing DPI determines whether the packet passes, and collects statistical information.

In a case of a packet in Internet, a packet has Ethernet header and trailer, IP header, TCP header, and payload. It optionally has an application protocol header, such as HTTP header, in the payload. Common IP routers and OpenFlow switches monitor IP headers and L4 headers, TCP and UDP headers, respectively. DPI usually means analyzing payloads.

Deeply Programmable Network (DPN) [11] is a network with which data planes can be programed. It enables to implement packet inspecting function in a switch. FLARE [11] is a network architecture for DPN. Click modular router [12] is a programming language for programing data plane.

C. N-gram

N-gram is continuous *n* items in a document. This is usually used in the text search field. *N*-grams of size one, two, and three are called unigram, bigram, and trigram, respectively. Fig. 1 shows an example of 2-gram. In the figure, the document “`This is a pen.`” includes “`Th`” once. It also includes “`hi`” and “`is`” once and twice, respectively. The continuous two letters, such as “`is`”, is bigram, and its frequency in the document is two.

Typically, *n*-gram is applied for text data. It can be adopted also for binary data.

```
This_is_a_pen.
Th : 1 time, at 0
hi : 1 time, at 1
is : 2 times, at 2,5
s_ : 2 times, at 3,6
_i : 1 time, at 4
:
```

Fig. 1. *n*-gram

Wei-Jen Li et al. proposed a method for identifying file type based on n-gram analysis [13]. This work adopts binary analysis based on n-gram. The method can identify only file type and the objective of this work is different from ours.

D. Application Identification using n-gram

Iwai et al. improved their work [9] using *n*-gram and achieved 90 % accuracy [3]. Their improved method is based on machine learning. Their work can present high accuracy, but their method requires five days for learning. Hence, identifying in short time is an important issue.

Wang et al. proposed a method for identification based on standard deviation and Mahalanobis distance based on *n*-gram [14]. However, this work does not provide deep investigation on encrypted communication. In addition, the work is only for detecting intrusion. The objectives of this and our work are different.

III. TLS Session Establishment

TLS session is established as shown in Fig. 2. Firstly, available TLS versions and cipher suites are sent from a client to a server. Cipher suites define key exchange algorithm and cipher specification. The server chooses TLS protocol version and cipher suite. Secondly, a common key is generated and shared as follows. The server sends a server’s certification and a public key to the client. The client generates a common key, then encrypts and sends the common key using the public key. The server receives and decrypts the common key using the private key. Finally, the client sends a message for change of cipher specification and complete of TLS session establishment.

In above processes, the processes from communication for decision of TLS protocol and cipher suites to sending of the server’s certification and the public key are performed in plain text. Thus, payload analyses using DPI can be applied. On the contrary, sending the common key and subsequent processes are encrypted. Thus, analyzing and extracting features are hard.

IV. PROPOSED METHOD

In this section, we introduce our model of service flows and propose a method for identifying service by a given HTTPS flow using DPI.

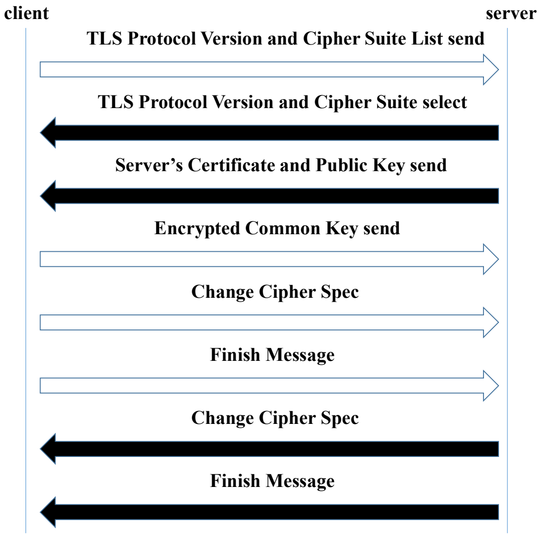
client
server
TLS Protocol Version and Cipher Suite List send
TLS Protocol Version and Cipher Suite select
Server's Certificate and Public Key send
Encrypted Common Key send
Change Cipher Spec
Finish Message
Change Cipher Spec
Finish Message

Fig. 2. TLS session establishment

A. *A system overview*

Fig. 3 and Fig. 4 illustrates the overview of our proposed method and model of service flow, respectively. As shown in the Fig. 4, multiple TLS sessions are established in one access of a service. For example, about ten TLS connections are establishes when a user opens a web search engine web page www.google.com. We call opening a web page via a browser "one access". The forwarding function in Fig. 3 is implemented with Click modular router and our identifying method is also implemented in this function.

B. *Traffic Model*

Traffic of an access to a service can be modeled as Fig. 4. After a user connected to a service web site via a browser, multiple TCP connections are established. In each TCP connection, a TLS connection is established. These sessions are clustered into several groups with similarity.

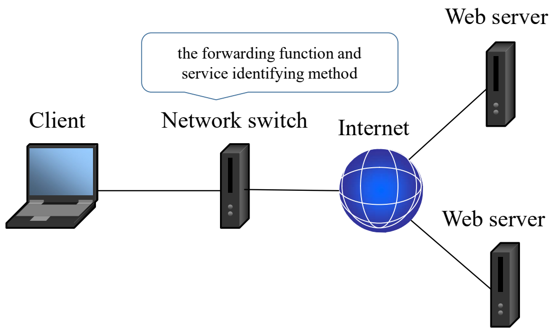

Fig. 3. System overview

C. *Identifying Method*

In this subsection, we propose a method for identifying service by given HTTPS network flows using DPI. The method is composed of the preliminary investigation phase and the identifying phase.

In the preliminary investigation phase, the following steps are performed.

1) Accesses to each service are performed, and traffic is captured.

2) The unencrypted parts in TLS session establishment are analyzed and *n-gram frequency database* is created. We call this frequency *n-gram frequency*.

3) The sessions are clustered with correlation coefficient of *n*-gram frequency.

2) and 3) are repeated until the number of sessions in every group exceeds *m*. *M* is a tuning parameter.

4) For every service, the number of sessions of each group is counted. We call this number *group frequency*.

In the case of Fig. 4, an access to a service is composed of several TLS sessions. For example, the access to the email service in the figure is composed of six sessions. All the sessions in all the services are analyzed and clustered by correlation coefficient of *n*-gram frequencies. For every service, group frequencies are counted. We call this *group frequency database*. In the case of the email service in the figure, four Group A sessions, two Group B sessions, and no Group C sessions are included. Focusing *m*, we can expect better accuracy and faster identification with larger and smaller *m*, respectively. As described in section III, transmitted data after sending the common key are encrypted, and then extracting features by inspecting them is almost impossible. Thus, our method analyses communication between choosing a TLS protocol and sending server's certification. We focus on packets sent from a server to a client because we expect that data from clients to servers contains information on the client and it may be noise for identification.

In the identification phase, the following steps are executed.

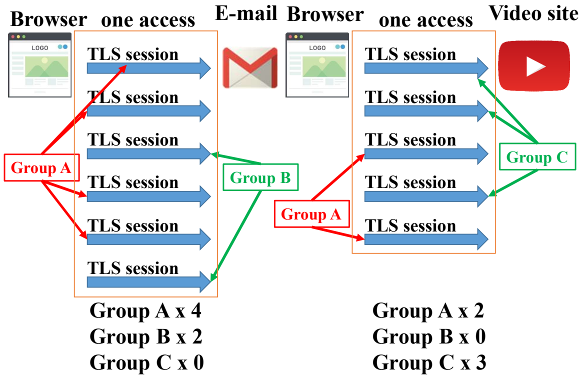

Fig. 4. Traffic model of an access to a service

1) The traffic for identifying is captured.

2) In every TLS session, *n*-gram frequency is calculated and every session is clustered with similarity. Every session is classified into the group which has the largest average correlation coefficient with *m* sessions in database.

3) Based on the previous step, the number of sessions of every group is counted.

4) The distance between the group frequency of the traffic for identifying and these of services in group frequency database are calculated. Then, the service with the smallest distance is determined as the identification result. The distance is calculated using the modified Manhattan distance, which we optimized. The original Manhattan distance and our modified Manhattan distance are as follows.

$$\text{original Manhattan distance(A, B)} = \sum_{i=1}^{G} d_{org}\,(a_i, b_i)$$

$$d_{org}\,(a_i, b_i) = |a_i - b_i|$$

$$\text{modified Manhattan distance(A, B)} = \sum_{i=1}^{G} d(a_i, b_i)$$

$$d(a_i, b_i) = \begin{cases} LD \;\; if\; a_i = 0 \; xor \;\; b_i = 0 \\ \quad |a_i - \; b_i| \;\; else \end{cases}$$

where *LD* is a number large enough.

The modified Manhattan distance implies that zero and non-zero has large difference. We apply this policy because of the following reason. Our preliminary investigation reveals that group frequencies are not constant for the same service. However, zero or non-zero is always kept. Thus, we expect that difference of existence, zero or non-zero, is a remarkable indicator of non-matching.

Fig. 5 shows an example of identification using modified Manhattan distance with *LD* = 10. In the figure, *freq*(Gr*k*, Sv*A*) is the frequency of Gr*k* in Sv*A* in group frequency database. *X* is a traffic for identification. Sv*A* and Sv*B* are services in group frequency database. Gr*a* to Gr*e* are the names of groups. The numbers in the tables are group frequencies of Services. In the case of *X* and Sv*A*, the original Manhattan distance and the modified Manhattan distance are two. In the case of *X* and Sv*B*, the modified Manhattan distance is 22, while the original Manhattan distance is 6. That is *d*(*freq*(Gr*k*, Sv*B*),*freq*(Gr*k*, *X*)) are 2, 0, and 0 with Gr*a*, Gr*b*, and Gr*c*, respectively. With Gr*d* and Gr*e*, *d*(*freq*(Gr*k*, Sv*B*), *freq*(Gr*k*, *X*)) are 10 and 10, respectively.

V. EVALUATION

In this section, we evaluate accuracy of identification. Experimental setups are as follows. *M* is ten. *N*-gram is 2-gram, i.e. frequencies of 65536 types of data are counted and correlation coefficient between two 65536 –vectors are calculated. The number of groups is ten.

	Gr*a*	Gr*b*	Gr*c*	Gr*d*	Gr*e*	
Traffic *X* for identification	3	2	0	3	2	
Sv*A*	3	3	0	2	2	
***d*(*freq*(grk, Sv*A*),*freq*(grk,*X*))**	0	1	0	1	0	→ 2

	Gr*a*	Gr*b*	Gr*c*	Gr*d*	Gr*e*	
Traffic *X* for identification	3	2	1	3	0	
Sv*B*	1	2	1	0	1	
***d*(*freq*(grk, Sv*B*),*freq*(grk,*X*))**	2	0	0	10	10	→ 22

Fig. 5. Example of identification using modified Manhattan distance

Ten Google services are used for evaluation. They are Google Web Search, YouTube, Google Play, Gmail, Google Drive, Google Calendar, Google Scholar, Google Translate, Google Plus, and Google News.

The modified Manhattan distances between the group frequency of the flows for identification and services in database are depicted in Fig. 6 and Fig. 7. In every figure, a label in the horizontal axis shows a flow for identifying. The vertical axis shows the average of the obtained modified Manhattan distance between the flows for identifying, written in the horizontal axis, and services in the group frequency database.

These figures demonstrate that identifications are correctly performed, with all the services in our experiment using ten Google service. That is, the modified Manhattan distance with the correct one is the least in every identification. In both figures, we can see large difference between the distance with the correct one and the least one of non-correct services. In the case of Google Web Search for example, the modified Manhattan distance with the correct one is 0.5. On the other hand, the least distance in non-correct ones is 11.1.

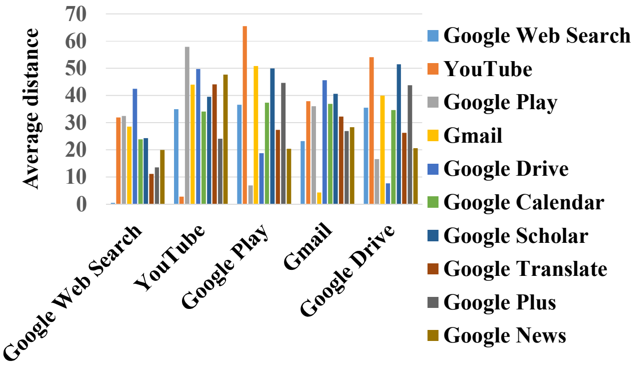

Fig. 6. The average modified Manhattan distance between the flows for identifying and service in database (1)

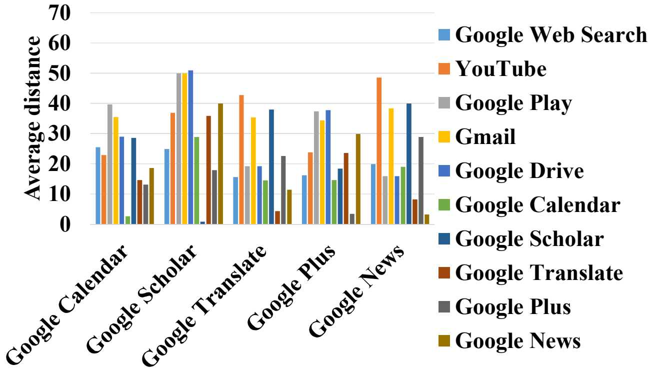

Fig. 7. The average modified Manhattan distance between the flows for identifying and service in database (2)

VI. DISCUSSION

This section discusses database updating, identification from more services, and *LD*.

We assume that software updates of server application results in change of *n*-gram and group frequencies. Thus, we can expect that updating *n*-gram and group frequency database is effective for keeping high accuracy. Updating database requires communication with the service and analyses of the traffic. Time to communicate with a service, which is time to open the web site, depends on network speeds, but it is several seconds at most in usual cases. The average time to update frequencies of a service in database is 6.59 seconds in our system. Updating database every day, for example, can be assumed very small load.

Next, we present discussion on identification with more services. Fig. 6 and Fig. 7 show large difference between correct and wrong identifications. Therefore, we can expect that our method works suitably with more challenging identification.

Last, we discuss effect of *LD*. We have evaluated our method only with *LD*=10. *LD* may have direct impact on accuracy. If a correct identification also includes a case of *LD*, i.e. a_i=0 *xor* b_i=0, too large *LD* may have negative impact. We think also that *LD* which is not large enough is not suitable because detection of a_i=0 *xor* b_i=0 is an important indicator of inaccurate identification. Optimization of *LD* can be explored by exhaustive investigation with many instances.

VII. CONCLUSION

In this paper, we proposed a method for identifying services from IP flows without depending IP addresses and port numbers. Unlike the existing identifying methods, the proposed method observes multiple TLS sessions. Our evaluation demonstrates that the method can identify services with high accuracy, 100 % in the case of our evaluation.

For future works, we plan to evaluate our method with various services of other companies and discuss a method for identifying based on Protocol Data Unit (PDU) analyses of TLS session.

ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant Numbers 24300034, 25280022, 26730040, 15H02696.

This work was supported by CREST, JST.

REFERENCES

[1] Service Name and Transport Protocol Port Number Registry, “http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml”

[2] A.W. Moore and K. Papagiannaki, "Toward the accurate identification of network applications," Passive and Active Network Measurement, pp.41-54, Springer, 2005.

[3] Takamitsu IWAI and Akihiro NAKAO, "Identification of Mobile Applications via In-Network Machine Learning Using N-gram for Application-Specific Traffic Control," IEICE Tech. Rep., vol. 115, no. 209, NS2015-78, pp. 41-46, Sept. 2015. (in Japanese)

[4] Petr Velan, Milan Čermák, Pavel Čeleda, and Martin Drašar, “A survey of methods for encrypted traffic classification and analysis,” Int. Journal of Network Management, 25, 5, pp. 355-374, September 2015.

[5] Qualys, Inc.. HTTP Client Fingerprinting Using SSL Handshake Analysis. Web page, 2016. Available from: "https://www.ssllabs.com/projects/client-fingerprinting/" {23 September 2016}.

[6] P0f,”http://lcamtuf.coredump.cx/p0f3/” {23 September 2016}.

[7] Ralph Holz, Lothar Braun, Nils Kammenhuber, and Georg Carle, “The SSL landscape: a thorough analysis of the x.509 PKI using active and passive measurements,” In Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference (IMC '11), pp. 427-444, 2011.

[8] Martin Husák, Milan Čermák, Tomáš Jirsík, Pavel Čeleda, “HTTPS traffic analysis and client identification using passive SSL/TLS fingerprinting,” EURASIP Journal on Information Security, Volume 2016 Issue 1, Article No. 30, December 2016.

[9] Takamitsu IWAI and Akihiro NAKAO, "Identification of Mobile Applications via In-Network Machine Learning for Application Specific QoS Traffic Control ", IEICE Tech. Rep., vol. 114, no. 477, NS2014-260, pp. 487-492, March, 2015.

[10] Masaki Hara, Shinnosuke Nirasawa, Akihiro Nakao, Masato Oguchi, Shu Yamamoto and Saneyasu Yamaguchi, "Fast Application Identification Based on DPI N-gram", 2016 IEEE 17th International Conference on High Performance Switching and Routing Workshop Program, June, 2016.

[11] A.Nakao, "FLARE: Open Deeply Programmable Node Architecture", Stanford Univ. Networking Seminar, Oct 2012

[12] Eddie Kohler et al., "Click Modular Router", ACM Trans. On Computer Systems, Aug. 2000

[13] Wei-Jen Li, Ke Wang, Salvatore J. Stolfo, “Fileprints: Identifying File Types by n-gram Analysis”, Proceedings from the Sixth Annual IEEE SMC Information Assurance Workshop, pp.64-71, 15-17 June 2005.

[14] K. Wang and S.J. Stolfo, "Anomalous payload-based network intrusion detection", Recent Advances in Intrusion Detection, pp.203-222 2004.