TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 07/12 pp239–254
DOI: 10.26599/TST.2019.9010008
Volume 25, Number 2, April 2020

DTA-HOC: Online HTTPS Traffic Service Identification Using DNS in Large-Scale Networks

Xuemei Zeng, Xingshu Chen*, Guolin Shao, Tao He, and Lina Wang

Abstract: An increasing number of websites are making use of HTTPS encryption to enhance security and privacy for their users. However, HTTPS encryption makes it very difficult to identify the service over HTTPS flows, which poses challenges to network security management. In this paper we present DTA-HOC, a novel DNS-based two-level association HTTPS traffic online service identification method for large-scale networks, which correlates HTTPS flows with DNS flows using big data stream processing and association technologies to label the service in an HTTPS flow with a specific associated domain name. DTA-HOC has been specifically designed to address three practical challenges in the service identification process: domain name ambiguity, domain name query invisibility, and data association time window size contradictions. Several experiments on datasets collected from a 10-Gbps campus network are conducted alongside offline and online testing. Results show that DTA-HOC can achieve an average online association rate on HTTPS traffic of 83% and a generic accuracy of 86.16%. Its processing time for one minute of data is less than 20 seconds. These results indicate that DTA-HOC is an efficient method for online identification of services in HTTPS flows for large-scale networks. Moreover, our proposed method can contribute to the identification of other applications which make a Domain Name System (DNS) communication before establishing a connection.

Key words: HTTPS; Domain Name System (DNS); service identification; big data; flow association

1 Introduction

Network traffic service identification is an important basis for monitoring and understanding the composition of network traffic[1]. It is therefore significant for network supervision and network service quality guarantees. With the enhancement of user privacy protection and network security awareness, encrypted traffic represented by HTTPS has continued to grow significantly over recent years. More than half of the world's web traffic is currently transmitted using HTTPS[2,3]. HTTPS encryption makes it very difficult to identify the service over HTTPS flows using traditional methods, such as packet decryption and Deep Packet Inspection (DPI). Due to the major challenge that this poses for network supervision, HTTPS traffic service identification issues are receiving the attention of many researchers[4,5]. A variety of web fingerprint-based service identification methods have been proposed. However, since these methods can only identify web services present in a fingerprint library defined in advance, they have poor flexibility and scalability. Studies have shown that Domain Name System (DNS) traffic can be used to classify HTTPS

• Xuemei Zeng and Xingshu Chen are with the Cybersecurity Research Institute, Sichuan University, Chengdu 610065, China. E-mail: zengxm @scu.edu.cn; chenxsh@scu.edu.cn.
• Xingshu Chen, Tao He, and Lina Wang are with the College of Cybersecurity, Sichuan University, Chengdu 610065, China. E-mail: chenxsh@scu.edu.cn; 641593700@qq.com; wlnlnw1992@163.com.
• Guolin Shao is with the College of Computer Science, Sichuan University, Chengdu 610065, China. E-mail: sgllearn @163.com.
∗ To whom correspondence should be addressed.
Manuscript received: 2018-10-07; revised: 2019-01-28; accepted: 2019-03-04

traffic by assuming that a web application will resolve the IP address of the HTTPS server via a DNS query before requesting an HTTPS flow[6]. By monitoring DNS queries, HTTPS flows can be labelled by the domain name associated with the IP address of the HTTPS server. Thus, DNS-based HTTPS classification methods do not require prior knowledge of a webpage fingerprint and are therefore flexible and scalable. Some existing methods first extract from DNS traffic the client IP addresses, server IP addresses, domain names, and the relationships among them, and save them in a database or a customized data structure; an HTTPS flow is then labelled by retrieving the domain name from the database or the data structure using a pair of client-server IP addresses (hereafter abbreviated as (c, s)) as a key[7]. However, due to the widespread use of various DNS caching mechanisms, the DNS queries corresponding to some HTTPS flows occur much earlier than the related HTTPS flow. Therefore, these methods make use of a warm-up phase to cache the DNS queries in advance so that DNS queries can be continuously updated and cached in the system running process. With an increase of system uptime, the accumulation of DNS data causes a system performance degradation. Some mechanisms have been designed to alleviate the problem of the excessive amount of DNS data (e.g., data expiration removal and data structure capacity limitations), but due to the fact that the pair (c, s) is used as the search key, these make DNS queries invisible for some HTTPS flows in the identification process and eventually result in a low flow association rate. As the network size and bandwidth increases, the number of nodes and simultaneous online users in a network grows rapidly, and the number of DNS requests and flows also rises. The problems identified in these methods therefore become more serious over time, rendering them unsuitable for large-scale network environments.

In view of the above problems, we propose a DNS-based two-level association HTTPS traffic online service identification method called DTA-HOC, which is suitable for large-scale network environments. In DTA-HOC, big data stream processing and association technologies are applied to realize the processing and association of massive DNS and HTTPS flows. First, we propose a near-real-time stream processing mode that uses an adjustable sliding time window to read DNS and HTTPS flows, which can simultaneously meet the two main requirements of online service identification, namely, timeliness and dependence on long-term DNS data. We then propose a two-level association mechanism based on two different associated keys, which can address the problem that some HTTPS flows cannot be associated due to *domain name query invisibility* (which will be detailed in Section 3.2) in the sliding time window. Furthermore, we designed an HTTPS flow label determination method based on the domain name request behaviours of clients for the two-level association mechanism when handling *domain name ambiguity* (which will be detailed in Section 3.1). Finally, we conducted several experiments using datasets from a 10-Gbps campus network to verify DTA-HOC. The experimental results show that the association rate of DTA-HOC can reach 91.18% (or 99.84%, when the impact of DNS data collection points on the association rate is ignored), the specific accuracy for HTTPS traffic service identification can reach 61.64%, and the generic accuracy can reach 86.16%. Moreover, the generic accuracy can reach 96.04% when multiple candidates are considered. When DTA-HOC was deployed in the 10-Gbps campus network, the average online association rate of HTTPS traffic in the actual campus network reached 83% and the data processing time per minute is less than 20 seconds. These experimental results indicate that DTA-HOC is an efficient method for the online identification of services in HTTPS traffic for large-scale networks.

The main contributions of this paper are as follows:

(1) We propose a two-level online association mechanism based on two different associated keys. The two-level association mechanism makes it possible to preserve the accuracy of the method with (c, s) as the key and address the problem of domain name query invisibility without caching DNS data for a long period of time. Moreover, the mechanism solves the problem that existing methods have of continuously accumulating and updating DNS data.

(2) We propose a multi-candidate domain name determination method for a two-level association mechanism based on the domain name request behaviours of clients. The method can address the problem of domain name ambiguity and enables DTA-HOC to achieve a higher service identification accuracy. It also provides the capability of sorting all candidate domain names. Thus, it is possible for one or more candidate domain names to be assigned to the associated HTTPS flows, better reflecting the service over the HTTPS flow.

The remainder of this paper is organized as follows. In Section 2, related work regarding HTTPS traffic service identification using domain names is briefly presented. In Section 3, we describe three problems in DNS-based HTTPS traffic online service identification in large-scale networks. We then highlight our proposed HTTPS traffic service identification method DTA-HOC and provide details of the implementation of its three components in Section 4. In Section 5, we show the experimental results and analyze the reasons for cases of lack of association and failed service identification. Finally, we conclude the work in Section 6.

2 Related Work

According to various sources of domain name acquisition, the methods of identifying HTTPS traffic service via domain names can be divided into two main types: those based on the Server Name Indication (SNI) field of Transport Layer Security (TLS) and those based on DNS traffic.

SNI-based methods classify TLS applications or HTTPS websites by extracting the domain name of the SNI extension from the "client hello" packet in the TLS handshake protocol. These methods aim to identify the "client hello" packet and extract the SNI information using DPI technology. SNI-based traffic monitoring methods have been integrated into many firewall solutions, such as Sphirewall[8] and IPFire[9]. However, SNI is not a mandatory requirement of the protocol, and SNI values can be easily forged to escape the filtering of security mechanisms[10]. Moreover, SNI is currently used alongside other information to classify HTTPS traffic. For example, along with the Common Name and other information in the HTTPS communication certificate, SNI information is used to create an HTTPS web page object fingerprint to identify HTTPS web pages[11]. Additionally, SNI has been used for service identification of HTTPS traffic after its accuracy was verified via DNS data[12]. However, since SNI-based methods must extract packet-level data and process application-layer information, they pose difficulties in large-scale network environments. Therefore, these methods are not suitable for large-scale networks.

DNS-based methods exploit the large number of applications that use DNS to obtain the server IP address and the rich information present in the clear text transmission of DNS traffic. They extract the relationship between the domain name and the server IP address from DNS traffic and save it in a database or custom data structure. When an HTTPS flow arrives, the specified domain name is assigned to the flow by retrieving the record from the database or the data structure, and then the service is identified from the encrypted network traffic. At present, there is a few researches based on this method; to the best of the our knowledge, the following works are all that are relevant. Plonka and Barford[6] were among the first to propose DNS-based web traffic classification method. In this method, each client's DNS collection point (i.e., Rendezvous) state information is collected and stored in a tree-like data structure; then, a classification label would be applied to each client-related web flow. Trevisan et al.[7] discussed the role of server IP addresses and hostnames to classify traffic on the web and found that collisions of names and addresses are common among popular services. However, they only provided a first look into traffic classification for modern web services and did not build a useable web traffic classifier. Bermudez et al.[13] proposed a web traffic classification system called DN-Hunter, which uses a First-In First-Out (FIFO) Cycle List (CList) of a given length to store a Fully Qualified Domain Name (FQDN). DN-Hunter retrieves the FQDN of the most recent entry from the CList with (c, s) as a key and assigns the FQDN to the web flow as the label. It has been applied in some real networks, but it has two problems: (1) HTTP/HTTPS flows with an invisible related DNS query cannot be correlated and labelled due to the absence of the corresponding DNS query stemming from the limitation of the CList's max length; and (2) the system can only return one FQDN as the flow label for a given client-server pair, and so multiple candidate FQDNs cannot be provided. Foremski et al.[14] proposed a traffic classification system called DNS-Class. DNS-Class saves the client IP addresses, server IP addresses, and domain names acquired from DNS flows in a database called Resolver and sets (c, s) as an index key. Its flow tagging module then labels each arriving flow by querying the domain name in the Resolver. Since the mapping relationships between the client IP addresses, server IP addresses, and domain names need to be continuously inserted and stored in the Resolver, and in the absence of a data expiration deletion mechanism, the Resolver becomes increasingly massive as the system runs. DNS-Class therefore has obvious inefficiencies since DNS data must be continuously written into the Resolver, which is also queried frequently. Mori et al.[15] proposed an encrypted web flow service inference framework called SFMap, which is based on a Domain Name Graph

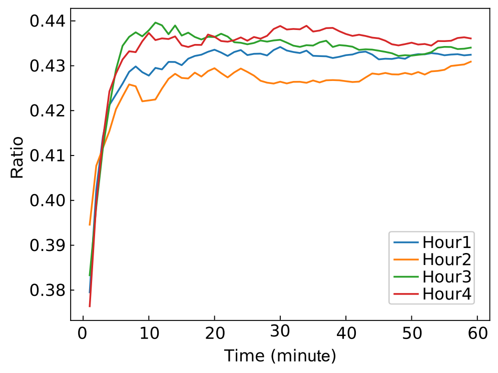
(DNG). SFMap stores the dynamic relationships among client IP address, server IP address, domain name, and canonical domain name (CNAME) in the DNG. The hostname of an HTTPS flow can be obtained by searching the domain name in the DNG with a key (c, s). Additionally, maximum likelihood probability is used to address the hostname selection problem when multiple hostnames are retrieved. SFMap obtains a good flow tagging effect on datasets from local area networks when prepared 10 hours in advance. If applied in a large-scale network, the DNG would become very large because all canonical domain names need to be stored. Although DNGs have a dynamic deletion and update mechanism based on the Time To Live (TTL) of a domain name, the node deletion of the DNG is invalid in some scenarios such as the UE-NTE. Therefore, despite its high accuracy, SFMap is unsuitable for large-scale networks.

3 Problem Analysis

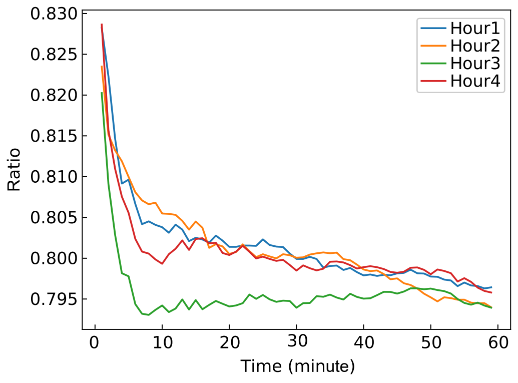
In large-scale network environments, online DNS-based HTTPS traffic service identification faces three problems when associating an HTTPS flow with the preceding DNS query: domain name ambiguity, domain name query invisibility, and data association time window size contradiction. In this section, we discuss each of these problems in turn.

3.1 Domain name ambiguity

Increasing numbers of web services are now being served from shared infrastructure such as Content Delivery Networks (CDN) and cloud computing. These web services often make use of the modern DNS ecosystem to optimize their resources, which makes many-to-many relationships between the server IP address and the domain name very common in popular services[16]. Approximately 35% of IP addresses can be associated with multiple domain names[6]. This phenomenon of an IP address being associated with multiple domain names is called *domain name ambiguity* in this paper. We analyzed domain name ambiguity using DNS data collected from multiple time periods (one hour per period) on the campus network. As seen from Fig. 1a, as time passes, the proportion of the total IP addresses that are mapped to multiple domain names increases from 38% to 44%, until after 15 minutes, when the proportion gradually stabilizes and remains in the range 42.5% to 44%. Figure 1b shows a case in which an IP address is mapped to one



(a)



(b)

Fig. 1 Proportion of the total IP addresses that are mapped to multiple domain names over time (in minutes). (a) Proportion of the total IP addresses that are mapped to multiple domain names over time; (b) Proportion of the total IP addresses that are mapped to multiple domain names with a public suffix over time.

domain name or multiple domain names with the public suffix. The public suffix here refers to the combination of the secondary domain and the root domain of the FQDN. For example, the public suffix shared between *xxx.example.com* and *yyy.example.com* is *example.com*, where "*example*" is the secondary domain and "*com*" is the root domain. As seen from Fig. 1b, the proportion reduces from 82% to 80% over time, until after 15 minutes, when it gradually stabilizes, maintaining a proportion between 79.5% and 80.5%.

Clearly there is a high probability of domain name ambiguity in actual networks. Therefore, we need to face the challenge of domain name ambiguity and find a method to choose the correct domain name from multiple options when labelling the service of an HTTPS flow. But it is worth mentioning that the existence of a public suffix can be used to label HTTPS flows and mitigate the challenge at some degree, since most IP addresses are either mapped to a single domain name or to multiple domain names with a public suffix.

3.2 Domain name query invisibility

The primary task of implementing DNS and HTTPS associations is to determine a key. The existing method is mainly based on the key (c, s); however, in an actual network, the DNS system uses a variety of caching mechanisms to cache DNS records, such as local DNS resolvers, the DNS cache within an operating system, and the DNS cache in applications such as a web browser, which results in domain names not being re-requested until their TLLs have expired[17]. These caching mechanisms make domain name queries invisible for some HTTPS flows during the data observation period, which leads to the absence of the corresponding DNS queries for some HTTPS flows.

3.3 Data association time window size contradiction

In online traffic service identification, both a DNS flow and an HTTPS flow arrive to the data processing platform in real time; however, the DNS query appears before the HTTPS flow. Thus, it is difficult to establish the association between them according to the one-to-one mechanism of streaming data processing. To associate the two flows effectively, a batch of flows arriving at a time must be obtained. The timing relationship between an HTTPS flow and the corresponding DNS flow in a real-time network environment is shown in Fig. 2. For the current moment t, a flow Flow_i arrives in the HTTPS flows at the current batch time window at t_{f_i} ($t_{f_i} \leqslant t$). Assuming that the duration of Flow_i is Duration_i, the start time of Flow_i is $t_{\text{StartTime}_i}$, and the TTL of the corresponding request domain DR_i is TTL_n, the occurrence/arrival time of the DNS response is then $t_{d_i} = t_{f_i} - \text{Duration}_i - \varepsilon$, where ε is the time difference between the DNS response arrival time and the Flow_i's start time $t_{\text{StartTime}_i}$, and $0 < \varepsilon < \text{TTL}_n$; when Flow_i is the first HTTPS flow after the DNS request, $\varepsilon \approx 0$. Thus, Flow_i and DR_i can be associated only when the data association time window size w has satisfied the condition $w \geqslant t - t_{d_i}$. In a real-world network scenario, various domain cache mechanisms mean that $\varepsilon \gg 0$ for some HTTPS flows, and Duration_i will be larger when Flow_i is a long-time connection. Only under the condition of $w \geqslant \text{Duration}_i + \varepsilon$ can an HTTPS flow and its corresponding DNS flow be associated successfully.

However, as the DNS time window size increases, the proportion of IP addresses mapped to multiple domain names also increases, and domain name ambiguity will affect the service identification accuracy. Moreover, in a large-scale network environment, when the associated time window increases, a large number of flows arriving at a high speed would increase both the amount of data that need to be processed and the calculation processing delay in each time slice, which will reduce the timeliness of the association. Therefore, the service identification method faces the problem of data association time window size contradiction.

4 Proposed Approach

4.1 Approach overview

DTA-HOC is designed based on a distributed big data real-time stream processing platform; the framework structure is shown in Fig. 3. The framework of DTA-HOC consists of three components: a stream data extractor, a DNS flow processor, and a two-level associator. The stream data extractor, which includes a DNS flow extractor and an HTTPS flow extractor, is responsible for reading the DNS and HTTPS flows required by association from the specified network streaming data buffers according to the given sliding time window parameters. The DNS flow data processor

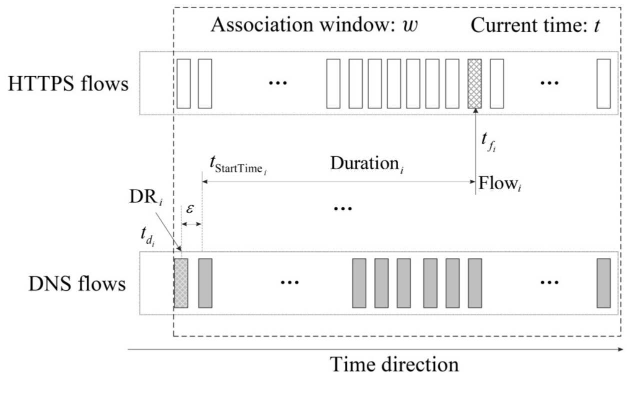


Fig. 2 Time relationship between an HTTPS flow and the corresponding DNS flow in a real-time network.

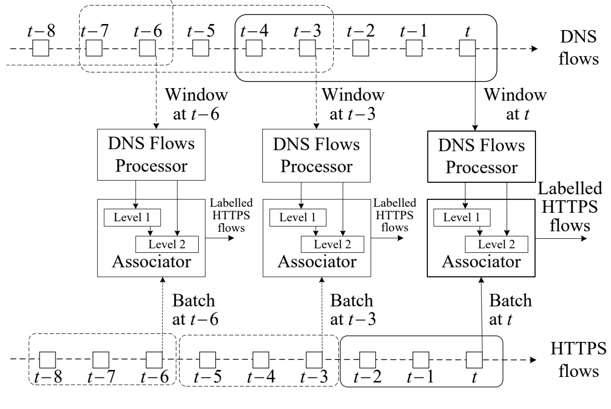


Fig. 3 Online association method DTA-HOC structure overview.

is responsible for parsing and processing the DNS flows obtained by the DNS flow extractor and forming DNS mapping relationship datasets G_{level1} and G_{level2} for the two-level association based on the (c, s) key and the (s) key, respectively. The two-level associator is responsible for hierarchically associating the HTTPS flows obtained by the HTTPS stream extractor with G_{level1} and G_{level2} and identifying the HTTPS flow using the associated domain name(s), thereby implementing HTTPS flow service identification.

4.2 Stream data extractor

4.2.1 Streaming data reading and processing modes

The micro-batch mode in big data stream processing is used to buffer the incoming data into a container according to the time window size and aggregate the data dropped in the window to perform calculations. It is suitable for the association analysis of DNS and HTTPS flows in an online environment. In this paper, the stream data extractor uses a sliding-window-based micro-batch mode to read real-time DNS and HTTPS flows. Let the DNS stream extractor process DNS flows with the sliding window size w each time with a forward step size x, and let the HTTPS stream extractor process HTTPS flows with the sliding window size b each time with a forward step size y. Because an HTTPS flow only needs to be processed once, there is no intersection between each batch of HTTPS flows (i.e., $y = b$). After the flows are divided by each slice of the time window, for the i-th time slice, the interval of the DNS time window w_i is $(t_i - w, t_i]$, and the time interval of the HTTPS time window b_i is $(t_i - b, t_i]$, where t_i is the right edge of the i-th time window. To ensure synchronization of the DNS and HTTPS time windows for each time slice, the right edge of the windows is set to be the same, and the sliding step size is $x = y$.

4.2.2 Data window size determination

In most scenarios, the same sliding time window size is adopted to process the data from different sources. However, since the DNS request arrives before the HTTPS flow, most HTTPS flows in the beginning of the time window will miss the corresponding DNS records in the associated time window when $w = b$. Increasing the size of the sliding window can alleviate the effect of domain name query invisibility. However, after the HTTPS sliding window size b is increased, the data association needs to wait until the right boundary condition of the sliding window is satisfied before starting, which will delay the output of the associated result and affect the real-time nature of the association. Therefore, to meet the real-time requirements of service identification and make the DNS data of more HTTPS flows fall into the associated time window, it is necessary to make b as small as possible (to improve real-time performance) and w as large as possible (to increase the association rate). However, according to the previous analysis, increasing w will increase the data processing time and affect the service identification accuracy. Therefore, various sliding window sizes for the DNS and HTTPS flows are set in DTA-HOC. Figure 3 shows the stream data extractor extracting DNS and HTTPS flows based on sliding windows with various parameter values. In DTA-HOC, b is determined according to the real-time requirement of service identification and the delay in data association processing in the actual network, and w is determined according to the distribution of the TTL of the requested domain names, the distribution of the duration of the HTTPS flows in the actual network, and the data processing delay of the data processing platform. According to the analysis presented in Section 3.3, the conditions $w > b$ and $w > \text{TTL}_{\text{most}} + \text{Duration}_{\text{most}}$ need to be satisfied, where TTL_{most} refers to the TTL value of most of the requested domain names in the actual network, and $\text{Duration}_{\text{most}}$ refers to the duration time of most of the HTTPS flows. Therefore, we set w to be equal to a value near $\text{TTL}_{\text{most}} + \text{Duration}_{\text{most}}$ when considering the performance of the actual data processing platform.

4.3 DNS flow processor

The DNS flow processor processes the output of the DNS flow extractor in each time slice to form the DNS mapping datasets G_{level1} and G_{level2} that are required by the two-level associator. Dataset G_{level1} uses (c, s) as a key to store the mapping from a client-server IP address pair to a request domain name. Dataset G_{level2} uses (s) as a key to store the mapping from a server IP address to a request domain name.

4.3.1 Two-level association DNS mapping datasets

Let C represent the set of all client IP addresses that have DNS request behaviour in the sliding time window. For a specific client IP c_i with $c_i \in C$, the server IP address set S_{c_i} is obtained from the DNS request/response in the time slice. According to domain name ambiguity, for a specific server IP address s_j with $s_j \in S_{c_i}$, there are one or more domain names pointing to it. Let the domain name set $N_{c_i,s_j} = \{n^m_{c_i,s_j} | 1 \leqslant m \leqslant \mathrm{Nnum}_{c_i,s_j}\}$ refer to all domain names pointing to s_j requested by client c_i, where $n^m_{c_i,s_j}$ refers to one of the domain names pointing to s_j in the time slice, and Nnum_{c_i,s_j} refers to the numbers of the distinct domain names in the time slice. Based on the idea of MapReduce in big data, we use the structure [Key, Value] to express the mapping relationship between two objects. Therefore, the mapping relationship between the pair (c_i, s_j) and the requested domain name set N_{c_i,s_j} can be expressed as $r_{c_i,s_j} = [(c_i, s_j), N_{c_i,s_j}]$, where Key $= (c_i, s_j)$ and Value $= N_{c_i,s_j} = (n^1_{c_i,s_j}, n^2_{c_i,s_j}, \ldots, n^{\mathrm{Nnum}_{c_i,s_j}}_{c_i,s_j})$. When $\mathrm{Nnum}_{c_i,s_j} = 1$, the (c_i, s_j) pair only maps to one domain name n_{c_i,s_j}, and we can use n_{c_i,s_j} as the label of the HTTPS flows associated in the current time slice. Additionally, when $\mathrm{Nnum}_{c_i,s_j} > 1$, we select one or multiple domain names as the label of the associated HTTPS flows according to the candidate domain name determination method (detailed in Section 4.3.3). We set a score ω for each domain name $n^m_{c_i,s_j}$ in the set N_{c_i,s_j} according the behaviour of the client c_i, and sort all domain names in N_{c_i,s_j} by the value of ω. Accordingly, the final expression of r_{c_i,s_j} is shown as Eq. (1):

$$r_{c_i,s_j} = [(c_i, s_j), ((n^1_{c_i,s_j}, \omega^1_{c_i,s_j}), (n^2_{c_i,s_j}, \omega^2_{c_i,s_j}), \ldots, (n^{\mathrm{Nnum}_{c_i,s_j}}_{c_i,s_j}, \omega^{\mathrm{Nnum}_{c_i,s_j}}_{c_i,s_j}))] \quad (1)$$

where $\omega^1_{c_i,s_j} \geqslant \omega^2_{c_i,s_j} \geqslant \cdots \geqslant \omega^{\mathrm{Nnum}_{c_i,s_j}}_{c_i,s_j}$. Thus, the mapping relationship dataset G_{level1}, which includes all the mapping relationships r_{c_i,s_j} in the time slice, can be expressed as $G_{\mathrm{level1}} = \{ r_{c_i,s_j} | c_i \in C, s_j \in S_{c_i}\}$.

Mapping relationship r'_{s_j} will be formed when ignoring the constraint of the client on the mapping relationship r_{c_i,s_j}, and its expression is shown as Eq. (2):

$$r'_{s_j} = [s_j, ((n^1_{s_j}, \omega^1_{s_j}), (n^2_{s_j}, \omega^2_{s_j}), \ldots, (n^{\mathrm{Nnum}_{s_j}}_{s_j}, \omega^{\mathrm{Nnum}_{s_j}}_{s_j}))] \quad (2)$$

Therefore, the mapping relationship dataset G_{level2}, which includes all mapping relationships r_{s_j} in the time slice, can be expressed as $G_{\mathrm{level2}} = \{r_{s_j} | s_j \in S\}$, where S refers to the set of all server IP addresses in the time window.

Only the mapping relationship from the pair of client IP addresses and the requested domain name to the resolved server IP address can be directly acquired from the DNS response. To generate datasets G_{level1} and G_{level2}, it is necessary to process the DNS data, reverse the transformation mapping relationship between domain names and server IP addresses, and calculate the score ω of each candidate domain name. Following is details of the process and calculation method.

4.3.2 Mapping relationship reverse transformation

The mapping relationship reverse transformation process is made up of two stages. The first stage is to parse each DNS response record in the DNS data and extract the client IP address, the requested domain name, and the (one or more) server IP addresses pointed to by the domain name, and then form a mapping from the pair of client IP addresses and the domain name to the server IP address(es). The second stage is to reverse the mapping relationship between the domain name and the server IP address to form multiple mappings from the client-server pair to the domain name.

Let Resp_i refer to the i-th DNS response record in the sliding time window. Thus, the mapping relationship between the pair (c, s) and the requested domain name acquired by resolving Resp_i can be expressed as $q_i = [(c, n), (s_1, s_2, \ldots, s_k)]$, where c refers to the client IP address, n refers to the requested domain names, s_k refers to one of the resolved server IP addresses, and k refers to the number of resolved server IP addresses with $k \geqslant 1$. Thus, we can obtain mapping relationship dataset Q by resolving all DNS response records within the sliding time window and $Q = \{q_i | 1 \leqslant i \leqslant \mathrm{Rnum}\}$, where Rnum refers to the number of all resolved server IP addresses in the current sliding time window.

The reverse transformation process of the requested domain name and the server IP address is shown in Eq. (3). First, the one-to-many mapping relationship of each q_i in Q is decomposed into multiple one-to-one mappings, as shown in (i) of Eq. (3). Then, the reverse transform is taken for each resolved mapping relationship from $[(c, n), s]$ to $[(c, s), n]$, as shown in (ii) of Eq. (3). Thus, the new mapping relationship dataset $\mathbb{Q}$ is made available, composed of all mapping relationships such as $[(c, s), n]$ generated by processing each element of Q through the two processes (i) and (ii) in Eq. (3).

$$q_i = [(c, n), (s_1, s_2, \ldots, s_k)] \overset{\text{(i)}}{\Rightarrow} \left\{ \begin{matrix} [(c, n), s_1] \\ [(c, n), s_2] \\ \cdots \\ [(c, n), s_k] \end{matrix} \right\} \overset{\text{(ii)}}{\Rightarrow} \left\{ \begin{matrix} [(c, s_1), n] \\ [(c, s_2), n] \\ \cdots \\ [(c, s_k), n] \end{matrix} \right\} \quad (3)$$

4.3.3 Candidate domain name determination

As a matter of experience, we believe that for a given time window, when the server IP address s obtained by parsing client c corresponds to multiple domain names, the most relevant domain name is that with the highest frequency of requests. Therefore, based on the client's domain name request behaviour, the requested frequency of the domain name is used to calculate the score of each candidate domain name in G_{level1}. Let $F_{c,s}(n)$ refers to the frequency with which domain name n is queried by client c and resolved server IP address s is obtained, then

$$F_{c,s}(n)=\frac{W([((c,s),n)]}{W([((c,s),*)]} \tag{4}$$

where $W(B)$ refers to the number of B occurrences in set $\mathbb{Q}$, and "$*$" refers to any of the domain names. Thus, the preferred service label of the HTTPS flow associated with G_{level1} is $\hat{n}$ subjected to $r_{c,s}$ and maximizing $F_{c,s}(n)$; that is

$$\hat{n} = \underset{r_{c,s}}{\arg\max}\, F_{c,s}(n) \tag{5}$$

Therefore, in Eq. (1), the first candidate domain name $n^1_{c_i,s_j}$ is $\hat{n}_{c_i,s_j}$ and $\omega^1_{c_i,s_j} = F_{c_i,s_j}(\hat{n}_{c_i,s_j})$, and the score of the l-th candidate domain name $n^l_{c_i,s_j}$ can be calculated by $\omega^l_{c_i,s_j} = F_{c_i,s_j}(n^l_{c_i,s_j})$.

We determine the score of each candidate domain name in G_{level2}, based on the domain name request behaviour of all clients in the slide time window. Therefore, the score of a domain name is calculated by the domain name request frequency of all clients. Let $F_s(n)$ refer to the frequency with which domain name n is requested by all clients and resolve IP address s is obtained in the slide time window, then

$$F_s(n)=\frac{W([(s,n)]}{W([(s,*)]} \tag{6}$$

Thus, the preferred service label of the HTTPS flow associated with G_{level2} is the candidate domain name $\hat{n}$, and $\hat{n}$ satisfies the mapping relationship r'_s and maximizes $F_s(n)$; that is

$$\hat{n} = \underset{r'_s}{\arg\max}\, F_s(n) \tag{7}$$

Therefore, in Eq. (2), the first candidate domain name $n^1_{s_j}$ is $\hat{n}_{s_j}$, $\omega^1_{s_j} = F_{s_j}(\hat{n}_{s_j})$, and the score of the l-th candidate domain name $n^l_{s_j}$ can be calculated by $\omega^l_{s_j} = F_{s_j}(n^l_{s_j})$.

4.4 Two-level associator

The DTA-HOC two-level associator processes the HTTPS flows read by the stream extractor and labels the HTTPS flows by associating them with the DNS stream processor outputs G_{level1} and G_{level2}. The first level is associated based on the (c,s) key. After association, HTTPS flows that were successfully associated are marked and output, while HTTPS flows that were not successfully associated in the first-level associator will be fed to the second-level associator. The second-level associator associates the HTTPS flows based on the (s) key. The two-level association mechanism makes full use of the advantage of (c,s)-based association (in terms of accuracy) and (s)-based association (in terms of association rate). It not only resolves the problem of domain name query invisibility but also increases the association rate to the same level as the (s)-based association method in reducing the size of the DNS time window used for association.

5 Experiments and Results Analysis

5.1 Experimental platform and datasets

In this paper, we performed experiments based on the big data analysis platform NTCI-BDP established by our laboratory. The platform is built on Hadoop and Spark and has one master node and 14 data nodes. The communication bandwidth between nodes is 10 Gbps. To facilitate the evaluation and verification of the DTA-HOC method, the experiments are performed both offline and online. The data used in the experiments was from the real environment of the authors' campus network. Offline data includes DNS and HTTP flow data. The main reasons for conducting offline experiments using HTTP data instead of HTTPS data are as follows: (1) HTTPS tag data sets and DNS associated data sets for experimentation are lacking, and manual data tagging is too costly; (2) the request header of HTTP data contains the hostname information requested by the client, which can be used as a label for identifying the flow; and (3) although HTTP and HTTPS differ in the distribution of hostnames, the fundamental mechanism of resolving domain names before starting HTTP/HTTPS communication should be consistent[15]. Therefore, the accuracy of our method can be evaluated by the method of associating DNS data with HTTP data. The HTTP data in the experiment comes from the HTTP flow records collected from the campus network border router. The fields of each HTTP flow include source IP address, destination IP address, source port number, destination port number, number of bytes, number of packets, flow start time, flow duration, and HTTP request hostname. The campus network has an export bandwidth of 10 Gbps, a traffic peak value that can reach 14 Gbps, and an online local user peak value that can reach 35 000 per second. HTTP traffic accounts for 35%–53% of the total traffic, and HTTPS traffic accounts for 14%–22% of the total traffic. The DNS data are collected from the campus network server area. Three DNS servers are deployed in the area to provide authoritative DNS mapping services for the data centre business systems and domain name resolution services for campus network users. The peak DNS response rate of DNS traffic data is approximately 300 000 per min, and the amount of different requested domain names is approximately 21 000 per min. Offline data are stored in Parquet format in HDFS. Real-time data includes the DNS flows of the server area and the HTTPS flows of the campus network border. The data are provided by the streaming data platform built by Apache Kafka. The experimental environment's logical topology is shown in Fig. 4. The experiment assumes that campus network users normally use the local DNS server on campus to access the Internet.

To simulate online data correlation scenarios to conduct experiments, we used offline HTTP and DNS data from four time periods on June 6, 2017 stored on the big data platform NTCI-BDP. The data details are shown in Tables 1 and 2. For ease of analysis, the HTTP dataset filters the flows for which the request hostname is empty or an IP address.

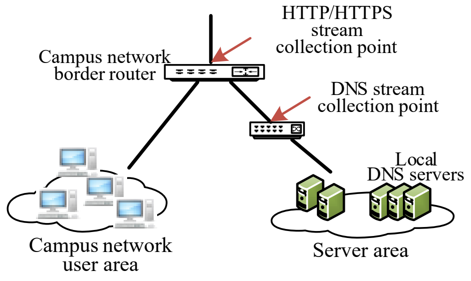

HTTP/HTTPS stream collection point
Campus network border router
DNS stream collection point
Local DNS servers
Campus network user area
Server area

Fig. 4 Logical network topology of the experimental environment.

5.2 Evaluation indices

DTA-HOC offers a refined classification of HTTPS traffic. However, at present, encrypted traffic identification is mainly evaluated by accuracy-related indicators, which are relatively coarse. To evaluate the effectiveness of the DTA-HOC method, we designed nine indices to measure the degree of association and the service identification accuracy of DTA-HOC. Accuracy is split into two types: specific and generic. The first indicates that the domain name assigned to an HTTPS flow is the same as its requested hostname, and the second indicates that the domain name assigned to an HTTPS flow shares the public suffix as its requested hostname. For a more fine-grained measure, each accuracy index has two sub-indices, which are represented by adding a 1 or 2 at the end of an indicator symbol.

(1) *Association rate* (R). This refers to the ratio of HTTPS flows associated with the DNS data to the total number of flows. Let Z be the total number of HTTPS flows in a given time window and M be the total number of HTTPS flows associated with the DNS, then $R = M/Z$. R is an index that reflects the degree of association between DNS flows and HTTPS flows.

(2) *Specific accuracy* (A). Let C be the number of HTTPS flows labelled correctly using the first candidate domain name. The specific accuracy indices can be expressed as $A1 = C/M$ and $A2 = C/Z$ and are used to measure the service identification accuracy of HTTPS flows by hostname.

(3) *Generic accuracy* (PSA). Let PS be the number of HTTS flows labelled correctly based on the public suffix. The generic accuracy indices can be expressed as $\text{PSA1} = \text{PS}/M$ and $\text{PSA2} = \text{PS}/Z$ and are used to measure the service identification accuracy of HTTPS flows. It is meaningful to use PSA to measure the

Table 1 Basic statistics of the HTTP dataset.

Time duration	Number of HTTP flows	Number of servers	Number of clients	Number of hostnames
10:00–11:00	658 638	5221	11 261	6075
16:00–17:00	591 096	5325	11 492	6148
20:00–21:00	517 143	4594	9837	5309
3:00–4:00	85 644	1345	1089	1307

Table 2 Basic statistics of the DNS dataset.

Time duration	Number of DNS responses	Number of distinct domain requests	Number of clients	Number of resolved distinct server IPs
10:00–11:00	12 629 110	214 918	33 334	93 275
16:00–17:00	13 197 486	224 025	34 483	95 700
20:00–21:00	11 358 331	194 534	28 647	82 902
3:00–4:00	2 589 382	69 324	10 979	28 802

service identification method because most multiple domain names that point to the same IP address have the public domain name suffix according to the analysis in Section 3.1.

(4) *Specific accuracy based on the first two* (P2A). Let $P2$ be the number of HTTPS flows labelled correctly using the first two candidate domain names. The specific accuracy based on the first two indices can be expressed as $\text{P2A1} = P2/M$ and $\text{P2A2} = P2/Z$ and used to measure the service identification accuracy of HTTPS flows.

(5) *Generic accuracy based on the first two* (P2SA). Let P2S be the number of HTTPS flows labelled properly using the first two candidate domain names based on the public suffix. The generic accuracy based on the first two indices can be expressed as $\text{P2SA1} = \text{P2S}/M$ and $\text{P2SA2} = \text{P2S}/Z$ and are used to measure the service identification accuracy of HTTPS flows.

(6) *Specific accuracy based on the top three* (P3A). This index is basically the same as P2A, except that it uses the first three candidate domain names. Let $P3$ be the number of HTTPS flows labelled correctly; then, $\text{P3SA1} = \text{P3S}/M$ and $\text{P2SA2} = \text{P3S}/Z$.

(7) *Generic accuracy based on the top three* (P3SA). This index is basically the same as P2SA, except that it uses the first three candidate domain names. Let P3S be the number of HTTPS flows labelled properly; then, $\text{P3SA1} = \text{P3S}/M$ and $\text{P3SA2} = \text{P3S}/Z$.

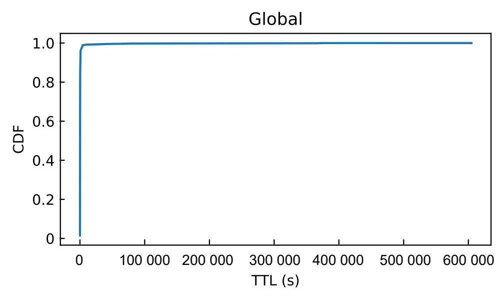
(8) *Specific accuracy based on the highest requested frequency* (FA). When calculating the score of a candidate domain name according to Eq. (4) or Eq. (6), there are some cases in which the probability of multiple domain names is the same and is at the maximum value. Let F be the number of HTTPS flows labelled correctly using the candidate domains names with the highest requested frequency; the specific accuracy based on the highest requested frequency indices can then be expressed as $\text{FA1} = F/M$ and $\text{FA2} = F/Z$.

(9) *Generic accuracy based on the highest requested frequency* (FSA). Let FS be the number of HTTPS flows labelled correctly using the candidate domains names with highest requested frequency based on the public suffix. The generic accuracy based on the highest requested frequency indices can be expressed as $\text{FSA1} = \text{FS}/M$ and $\text{FSA2} = \text{FS}/Z$.

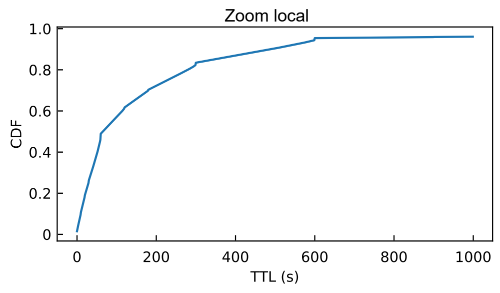
5.3 Estimation accuracy and discussion

5.3.1 Experimental method and parametric determination

According to the analysis in Section 3, the time when we obtain an HTTP/HTTPS flow record is the end time of the flow in an online environment. However, an HTTP/HTTPS flow record only stores the flow's start time and duration, but not the flow's end (or arrival) time. Therefore, to accurately simulate the online data scenario in an offline environment, an HTTP flow record for association is extracted from the offline datasets according to the condition that the calculated value of the flow end time falls within the time window. The formula for the flow end time is $\text{end\_time} = \text{start\_time} + \text{duration}$. The parameters in DTA-HOC include the DNS sliding time window size w, sliding step size x, HTTP sliding time window size b, and sliding step size y. The smaller the value of the parameter b, the better the real-time performance; however, in a large-scale network environment, near-real-time data service identification is sufficient to meet the actual requirements. Therefore, we set $b = 1$ as the HTTP sliding time window parameter value. To determine parameter w, we firstly analyzed the distribution of TTL of domain names and the distribution of duration of HTTP flows of the experimental datasets. Figure 5 reports the Cumulative Distribution Function (CDF) of the value of TTL of the requested domain names and Fig. 6 shows the CDF of the value of the durations of HTTP flows. As shown in Figs. 5 and 6, more than 95% of requested domain names have a TTL value of less than 800 seconds and more than 95% of HTTP flows have a duration



(a)



(b)

Fig. 5 Distribution of TTL of requested domain names. (a) Overall view; (b) Local zoom at the beginning.

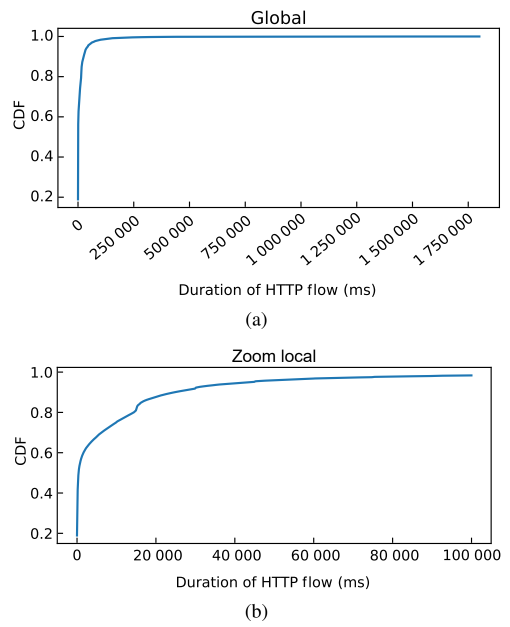


Fig. 6 Distribution of duration of HTTP flows. (a) Overall view; (b) Local zoom at the beginning.

value less than 80 000 ms, while $\text{TTL}_{\text{most}}+\text{Duration}_{\text{most}}$ is equal to 880 seconds (14.67 min). Therefore, in accordance with Section 4.2.2, we can set $w = 15$ as the DNS sliding time window parameter value for DTA-HOC in the current network environment. Additionally, to maintain synchronization of the two data windows, we set $x = y = b$.

To investigate the effect of data window size on the association rate and accuracy, w is set to 1, 2, 3, 5, 10, 15, 20, 25, and 30 minutes and b is set to a fixed value of 1 minute. The experimental results for different values of w on four time periods of the experimental datasets are shown in Fig. 7. The experimental results show that correlation rate R and accuracy indicators $A1$, $A2$, PSA1, and PSA2 increase as the time window increases. When $w < 15$, the growth rate increases, and then it tends to stabilize; when $w > 15$, the service identification ability barely improves, and a few indicators, such as $A1$, PSA1, and PSA2, decreased slightly. Therefore, it is appropriate to set $w = 15$ for the networks before taking account of the data processing delay. The data processing time on NTCI-BDP is described in Section 5.4.

5.3.2 Experimental results

To verify our proposed method, we compare it with the method based on the (c, s) key association (hereafter referred to as BCS), the method based on the (s) key association (hereafter referred to as BS), and the DN-Hunter method[13] using the evaluation indicators R, $A1$, and $A2$. Note that BCS uses a client's domain

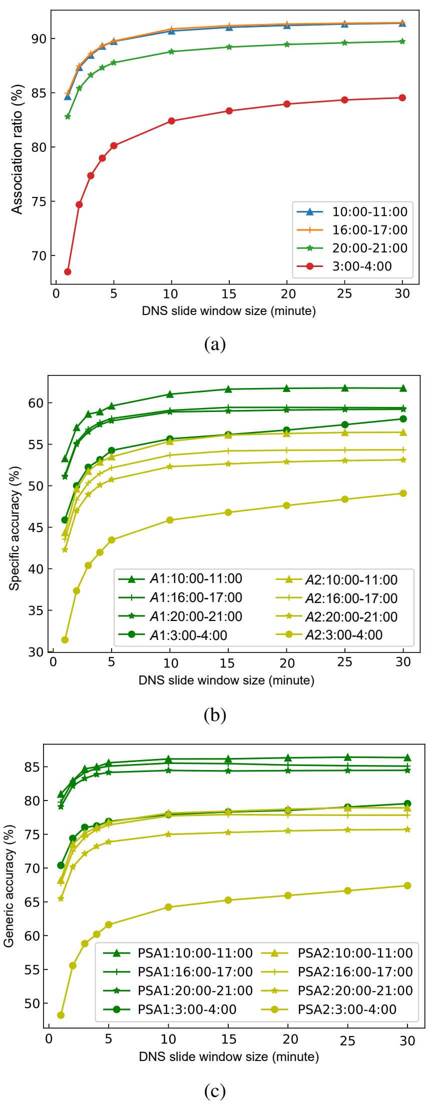


Fig. 7 Trends in evaluation indicators under varied DNS sliding time window parameter. (a) Line graph of R versus DNS time window size; (b) line graph of $A1$, $A2$ versus DNS time window size; and (c) line graph of PSA1, PSA2 versus DNS time window size.

name request behaviour to select the candidate domain name for the associated HTTP flow, and BS uses all client domain name request behaviours to perform this task. Additionally, from the literature we choose DN-Hunter as the comparison method because currently it is, to the best of our knowledge, the only method that can be run in a large-scale network environment. Since the existing related studies verified their methods on datasets collected from specific non-public network environments, we implemented BCS, BS, and DN-Hunter and ran them on the four datasets of this paper for a comparative experiment. In accordance with the discussion in the previous section, the parameters of DTA-HOC, BCS, and BS in the comparison experiment are set as follows: $w = 15$, $x = 1$, $b = 1$, and $y = 1$. The comparative experimental results data are shown in Fig. 8.

The following conclusions can be drawn from Fig. 8:

(1) BCS attains a better service identification accuracy than BS, but BS has a significant advantage regarding the association rate. BCS gives a 19.04%–35.93% lower association rate than BS, but BCS gives a 10.9%–15.03% more accurate $A1$ than BS and a 0.1%–10.88% more accurate $A2$. Since BCS requires that the associated DNS request and HTTPS flow come from the same client, its ability to accurately label the associated data is obviously stronger than that of BS (i.e., BCS is significantly better than BS with regard to the $A1$ indicator). However, HTTPS flows with invisible DNS requests in the time window are not associated by BCS, which affects the association rate (i.e., the association rate R of BCS is significantly lower than that of BS). BS does not consider the client and can associate and mark HTTPS flows with missing DNS requests using the related DNS requests of other clients; however, this will reduce the accuracy because the domain name requested preference determined from all clients does not reflect the domain name requested behaviour of a particular client very well. The experimental data shown in Fig. 8 verifies this conclusion.

(2) DTA-HOC takes full advantage of BCS and BS. The service identification effect of DTA-HOC is better than that of those two methods in general. For association rate R, DTA-HOC performs as well as BS and better than BCS; for specific accuracy, DTA-HOC performs 10%–16% better than BS and 0.3%–0.6% worse than BCS; for $A2$, DTA-HOC performs significantly better (11%–16%) than the other two methods.

(3) Compared with DN-Hunter, DTA-HOC performs better for R, $A1$, and $A2$ in general. The association rate R of DTA-HOC is 13%–19% higher than that of DN-Hunter, $A1$ is 2%–3% higher, and $A2$ is 11%–13% higher.

Specifically, on dataset 3:00 to 4:00, the $A1$ of DTA-HOC is 9% lower than that of DN-Hunter and 8.6% than that of BCS. The main reason is that the ratio of HTTP flows associated in the second level of DTA-HOC on the dataset 3:00–4:00 is much higher than on the other datasets. The ratio on the dataset 3:00–4:00 comes up to 42%, but it is 23% on the other dataset. The increase in the ratio of second-level associations reduces the specific accuracy of DTA-HOC. The reason for the first-level association rate decrease will be discussed in the next subsection. Actually, $A2$ of DTA-HOC is 13.4% higher than that of DN-Hunter. That is to say, DTA-HOC has a superior capacity to accurately identify more HTTP flows than DN-Hunter.

Therefore, DTA-HOC is superior to the other

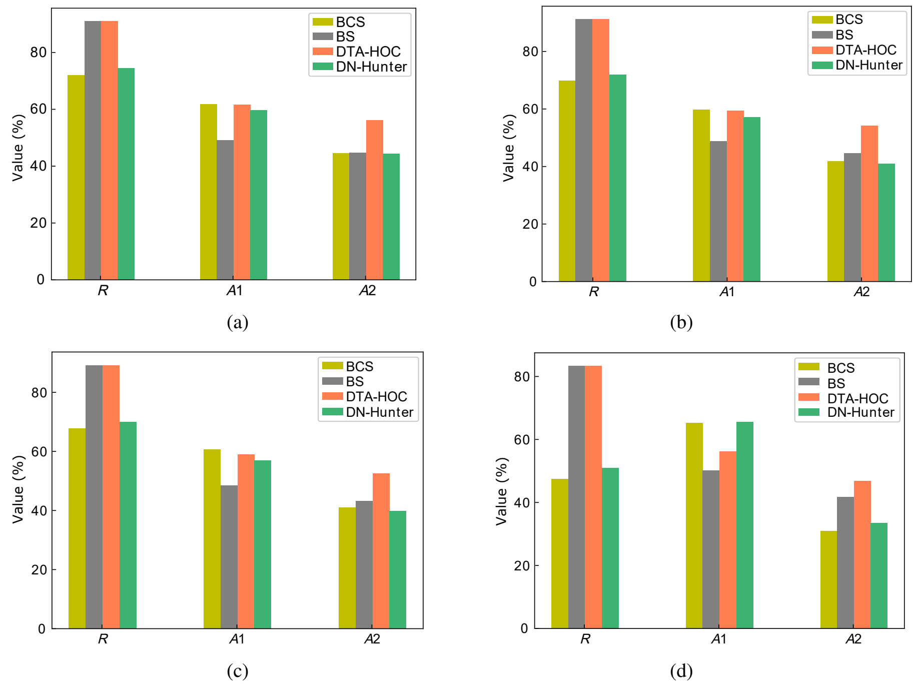


Fig. 8 Performance comparison between BCS, BS, DTA-HOC, and DN-Hunter. (a) Comparison results on dataset 10:00–11:00; (b) comparison results on dataset 16:00–17:00; (c) comparison results on dataset 20:00–21:00; and (d) comparison results on dataset 3:00–4:00.

methods at HTTP traffic service identification. Moreover, based on the previous analysis of the relationship between HTTPS and HTTP, DTA-HOC also has better service identification capabilities on HTTPS traffic.

We also analyzed the various evaluation indicators of DTA-HOC on the offline datasets under the specified parameters. As seen in Table 3, DTA-HOC can associate approximately 83.33%–91.18% of HTTP flows with DNS. Approximately 46.79%–56.11% of HTTP flows had the service accurately labelled, and approximately 65.25%–78% of HTTP flows can have the service marked by the public suffix. Moreover, the specific accuracy of HTTP flow service identification based on multiple candidate domain names can reach 90%, and the generic accuracy can reach 96.04%. These experimental results further verify that DTA-HOC is effective at distinguishing the service of HTTP flows.

5.3.3 Discussion

Here we discuss why some HTTP flows cannot be associated or accurately classified in the HTTP experiments.

(1) Non-association analysis

In our experiments, it was assumed that campus network users use the campus network local DNS service to access the Internet. However, some users do not use the local DNS service; therefore, their DNS request/response data will not pass through the DNS data collection point. The HTTP traffic generated by this group of users at the campus network border has no associated DNS requests. To analyze the impact of this phenomenon on the experimental association rate, we extract the server IP address(es) and the request hostname(s) from the unrelated HTTP flows, and then retrieve them in the DNS response data over the past one day to determine whether the pair of server IP address and request hostname appears simultaneously in a DNS response record. We then estimate the percentage of unallocated HTTP traffic caused by not using the local DNS. The experimental results show that 92.51% of non-associated server IP addresses do not appear in DNS response data in the period from 3:00 to 4:00, and the proportion in the other three time periods is 98%, 98.2%, and 99.32%, respectively. Therefore, the main reason for non-associated occurrences is that some users in the campus network do not use the local DNS service. This problem can be solved by adding DNS traffic at the same collection point as HTTP traffic. If this factor is excluded, the association rate of DTA-HOC can reach 99.84%, and almost all HTTP flows can be associated.

In addition, since the time window size limits the amount of DNS flows that can be obtained in each time slice, domain name invisibility would result in DNS flows that correspond to partial HTTP flows to be absent in the time slice. According the previous experiment, 7.59% of the non-associated HTTP flows in the 3:00–4:00 period occur because the corresponding DNS requests are missing in the time slice. Further, we counted the average TTL value of the requested domain names and a proportion η over the four experimental time periods, where η is the proportion of HTTP flows whose duration is greater than DNS sliding time window size w in all HTTPS flows. The experimental results show that the average value of TTL during the period from 3:00 to 4:00 is 3584 seconds, which is 2- to 3-fold that of the other three periods (1504, 1112, and 1043 seconds). Additionally, proportion η during the period from 3:00 to 4:00 is 0.1%, which is 2- to 3-fold that of the other three periods (0.03%, 0.04%, and 0.05%). The results further verify that domain name query invisibility is more prone to appear during the 3:00–4:00 period. This analysis is consistent with the results in Fig. 8; i.e., the value of indicator R in the 3:00–4:00 period is significantly lower than that of the other three periods, and the first level association rate of

Table 3 Classification results based on DTA-HOC for HTTP ($w = 15, x = y = b = 1$).

(%)

Time duration	R	$A1$	PSA1	P2A1	P2SA1	P3A1	P3SA1	FA1	FSA1
10:00–11:00	91.03	61.64	86.16	72.99	92.65	79.28	94.47	90	96.04
16:00–17:00	91.18	59.44	85.46	70.38	92.24	76.68	94.39	88.3	95.99
20:00–21:00	89.21	59.01	84.37	71.01	90.97	77.17	93.41	87.94	95.01
3:00–4:00	83.33	56.16	78.31	68.52	85.43	73.5	86.89	80.36	88.13
Time duration	R	$A2$	PSA2	P2A2	P2SA2	P3A2	P3SA2	FA2	FSA2
10:00–11:00	91.03	56.11	78.42	66.44	84.33	72.16	85.99	81.92	87.42
16:00–17:00	91.18	54.2	77.92	64.17	84.1	69.92	86.06	80.51	87.52
20:00–21:00	89.21	52.64	75.27	63.35	81.16	68.85	83.33	78.45	84.76
3:00–4:00	83.33	46.79	65.25	57.09	71.18	61.25	72.4	66.96	73.44

DTA-HOC is also much lower.

(2) Inaccuracy analysis

Through the analysis of inaccurate data, we found that HTTP requested hostnames, such as *wx?.sinaimg.cn*, *dldir?.qq.com*, *img?.imgtn.bdimg.com*, etc. (where "?" indicates the number 1, 2, 3, etc.), have a high service identification error rate. Further analysis found this problem is due to the HTTP1.1 specification. The specification defines that the client should not establish more than two HTTP links with the server; if this limit is exceeded, the new connection will be blocked. Web browsers such as IE strictly abide by this rule[18]; for example, when a web page displays multiple images, the IE user's image download speed will be affected. To solve the bottleneck caused by displaying multiple images on the same webpage, many large websites, such as *Baidu* and *Sina*, use the same group of image servers and multiple second-level domain names to increase the number of simultaneous image download connections. Therefore, in a webpage request, there are multiple different domain names that point to the same server IP address, and the query times and access times of these domain names are both basically the same. That is, in a very small time slice, multiple domain name requests parsed to the same IP address are interleaved with the corresponding HTTP communication. Existing DNS-based web traffic service identification schemes (including our method) are currently unable to distinguish this situation. Further analysis shows that the assigned domain names of some HTTPS flows that were not accurately marked have the public suffix as their actual hostname. Although the designated flow identifier by DAT-HOC is not the same as the actual hostname, this does not affect the service identification accuracy of the HTTP flow service.

Additionally, we find that there are many mismatches caused by certain CDN domain names. These CDN domain names from DNS flows have different public suffixes than the requested hostname of the HTTP flow, but they have a certain degree of similarity in text characteristics to the actual hostname requests. For example, *p1.music.126.net* and *p1.music.126.net.wscdns.com* have the same service domain name as the third-level sub-domain of CDN; for *c.hiphotos.baidu.com* and *hiphotos.jomodns.com*, the two domain names have different public suffixes, but the three-level subdomains have high similarity. Therefore, we use the Levenshtein distance and the Jaro distance algorithms to measure the textual similarity between the candidate domain name and the requested hostname on the second-level sub-domain and the third-level sub-domain. From these algorithms we find that there is a similarity between the first candidate domain name and the requested hostname for 48% of HTTP flows. That is, the selected candidate domain names in the above scenarios can identify the service of flows effectively although they do not match the actual hostnames of the flows.

5.4 Online performance on HTTPS

DTA-HOC is deployed to the online environment of the campus network, and real-time DNS flows and HTTPS flows are read from the streaming data platform with the same time window parameters as used for the offline environment. The data input rate and the actual data processing time on the big data platform are shown in Fig. 9. The processing time of a minute of data takes only 20 seconds or less, which meets the real-time processing requirements of large-scale network traffic.

The association results are further visualized in Fig. 10, which shows the total number of HTTPS flows, the number of associated HTTPS flows, and the change of the association rate over time. As seen in Fig. 10, the change of the associated flow number curve is in good agreement with the change of the total flow number curve. The association rate ranges from 55% to 89%, and the average association rate based on time accumulation is approximately 83%. The association rate fluctuates over time. Similar to the

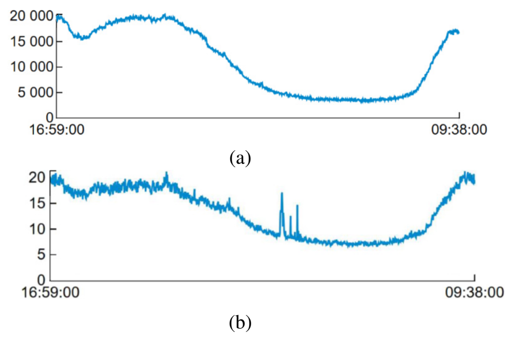


Fig. 9 Data input rate and total processing time in the online environment. (a) Data input rate, where *x*-coordinate represents time (minute) and *y*-coordinate represents the average number of input records per second in one minute. (b) Total processing time, where *x*-coordinate represents time (minute) and *y*-coordinate represents time spent processing data in one minute (the diagrams obtained from the Job Streaming monitoring webpage provided by the big data platform).

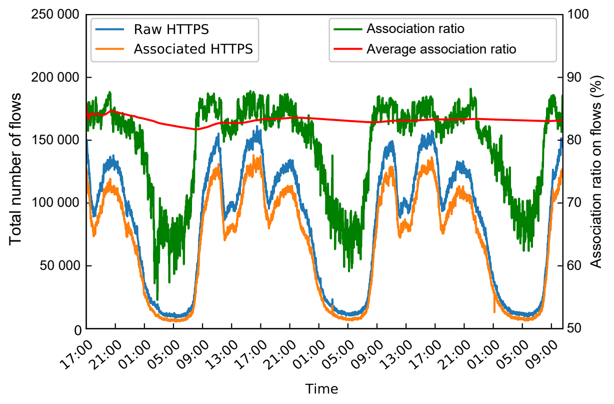


Fig. 10 Number of associated HTTPS flows online and the change of the association rate over time (per minute).

HTTP experiments, the association rate in the early morning hours was significantly lower than that in the later morning hours.

Based on the analysis of HTTP offline experiments, we consider the following three reasons for why some HTTPS flows are not associated. The first reason is the domain name query invisibility caused by the position of the DNS data collection point. If the impact of DNS collection points on the correlation results is ignored, the average association rate will increase to at least 90%. The second reason is the DNS query invisibility caused by the sliding time window mechanism. Similar to the HTTP experimental scenario, the DNS caching mechanism leads the corresponding DNS queries of some HTTPS flows to not fall into the current time window, a phenomenon that is more prominent in the early morning hours on the campus network. We find that the above two reasons are the main causes for the association rate of HTTPS flows in the early morning hours being significantly lower than in other time periods. The third reason is that some HTTPS applications hardcode IP addresses directly into URLs and do not send DNS requests. DTA-HOC assumes the existence of the corresponding DNS query before HTTPS communication and is invalid for applications that only use IP addresses to provide services.

6 Conclusion

In this paper, we presented a DNS-based two-level association HTTPS traffic service identification method called DTA-HOC. By addressing three practical challenges, domain name ambiguity, domain name query invisibility, and data association time window size contradictions, DTA-HOC is able to perform online identification of the web service over an HTTPS flow in a large-scale network environment. Additionally, DTA-HOC does not depend on prior knowledge of the identified webpage. Moreover, our proposed method might contribute to the identification of other applications that perform DNS communication before establishing a connection.

However, DTA-HOC is insufficient in two aspects. First, the method is still affected by the domain name ambiguity problem caused by webpage optimization against the limited number of permitted web browser connections. Second, the effectiveness of the method depends on the visibility of DNS traffic in the network; therefore, the method will be invalid if there are no DNS queries related to the HTTPS flow from the client or if applications and services do not rely on DNS and work with IP addresses only. We aim to propose a further method to solve the above problems in future work to achieve an in-depth analysis of encrypted traffic.

Acknowledgment

This work was partially funded by the National Natural Science Foundation of China (No. 61802270), National Entrepreneurship & Innovation Demonstration Base of China (No. C700011), Key Research & Development Project of Sichuan Province of China (No. 2018GZ0100), and Fundamental Research Business Fee Basic Research Project of Central Universities (No. 2017SCU11065).

References

[1] W. B. Pan, G. Cheng, X. J. Guo, and S. X. Huang, Review and perspective on encrypted traffic identification research, (in Chinese), *J. Commun.*, vol. 37, no. 9, pp. 154–167, 2016.

[2] G. Gebhart, We're halfway to encrypting the entire web, https://www.eff.org/deeplinks/2017/02/were-halfway-encrypting-entire-web, 2017.

[3] Google, HTTPS encryption on the web, https://transparencyreport.google.com/https/overview, 2018.

[4] P. Velan, M. Čermák, P. Čeleda, and M. Drašar, A survey of methods for encrypted traffic classification and analysis, *Int. J. Netw. Manage.*, vol. 25, no. 5, pp. 355–374, 2015.

[5] Z. G. Cao, G. Xiong, Y. Zhao, Z. Z. Li, and L. Guo, A survey on encrypted traffic classification, in *Proc. 5^{th} Int. Conf. Applications and Techniques in Information Security*, Melbourne, Australia, 2014, pp. 73–81.

[6] D. Plonka and P. Barford, Flexible traffic and host profiling via DNS rendezvous, in *Proc. Workshop on Securing and Trusting Internet Names*, Cambridge, UK, 2011, pp. 1–8.

[7] M. Trevisan, I. Drago, M. Mellia, and M. M. Munafò, Towards web service classification using addresses and DNS, in *Proc. 12^{th} Int. Wireless Communications and Mobile Computing Conf.*, Paphos, Cyprus, 2016, pp. 38–43.

[8] Sphirewall, http://www.sphirewall.net/, 2018.

[9] IPFire, http://www.ipfire.org/, 2018.

[10] W. M. Shbair, T. Cholez, A. Goichot, and I. Chrisment, Efficiently bypassing SNI-based HTTPS filtering, in *Proc. 2015 IFIP/IEEE Int. Symp. Integrated Network Management,* Ottawa, Canada, 2015, pp. 990–995.

[11] N. Kang, Research on fingerprint extrantion and identification of HTTPS web traffic, (in Chinese), Master dissertation, Harbin Institute of Technology, Harbin, China, 2017.

[12] W. M. Shbair, T. Cholez, J. François, and I. Chrisment, Improving SNI-based HTTPS security monitoring, in *Proc. 36th Int. Conf. Distributed Computing Systems*, Nara, Japan, 2016, pp. 72–77.

[13] I. N. Bermudez, M. Mellia, M. M. Munafò, R. Keralapura, and A. Nucci, DNS to the rescue: Discerning content and services in a tangled web, in *Proc. 2012 Internet Measurement Conf.*, Boston, MA, USA, 2012, pp. 413–426.

[14] P. Foremski, C. Callegari, and M. Pagano, DNS-Class: Immediate classification of IP flows using DNS, *Int. J. Netw. Manage.*, vol. 24, no. 4, pp. 272–288, 2014.

[15] T. Mori, T. Inoue, A. Shimoda, K. Sato, K. Ishibashi, and S. Goto, SFMap: Inferring services over encrypted web flows using dynamical domain name graphs, in *Proc. 7th Int. Workshop on Traffic Monitoring and Analysis*, Barcelona, Spain, 2015, pp. 126–139.

[16] V. Gehlen, A. Finamore, M. Mellia, and M. M. Munafò, Uncovering the big players of the web, in *Proc. 4th Int. Workshop on Traffic Monitoring and Analysis*, Vienna, Austria, 2012, pp. 15–28.

[17] T. Callahan, M. Allman, and M. Rabinovich, On modern DNS behavior and properties, *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 3, pp. 7–15, 2013.

[18] IEBlog, Internet explorer and connection limits, https://blogs.msdn.microsoft.com/ie/2005/04/11/internet-explorer-and-connection-limits/, 2005.

Xuemei Zeng is an engineer at Cybersecurity Research Institute of Sichuan University, China. She received the MS degree from Sichuan University, China, in 2004, where she is currently pursuing the PhD degree. Her current research interests include network behavior analysis and big data analysis for security.

Xingshu Chen is a full professor, doctoral supervisor at the College of Cybersecurity from Sichuan University, Chengdu, China. She received the MS and PhD degrees from Sichuan University in 1999 and 2004, respectively. Her main research focuses on cloud computing, big data analysis, and network security.

Guolin Shao is a PhD candidate in computer science at Sichuan University, China. His research interests include deep learning for cyber security and network behavior analysis. He has published more than 16 academic papers so far.

Tao He is a master student at Sichuan University, China. He received the BS degree from Sichuan University in 2017. His research interests include big data analysis for security.

Lina Wang received the BS degree from Yanshan University, China, in 2013, and the MS degree from Southwestern University of Finance and Economics, China, in 2016. She is currently pursuing the PhD degree with the School of Cyber Security, Sichuan University, China. Her current research interests include deep learning and cyber security.