Website Fingerprinting on Early QUIC Traffic

Pengwei Zhan[a,b], Liming Wang[a], Yi Tang[c,*]

[a]*Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China*
[b]*School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China*
[c]*School of Mathematic and Information Science, Guangzhou University, Guangzhou, China*

Abstract

Cryptographic protocols have been widely used to protect the user's privacy and avoid exposing private information. QUIC (Quick UDP Internet Connections), as an alternative to traditional HTTP, demonstrates its unique transmission characteristics: based on UDP for encrypted resource transmission, accelerating web page rendering. However, existing encrypted transmission schemes based on TCP are vulnerable to website fingerprinting (WFP) attacks, allowing adversaries to infer the users' visited websites by eavesdropping on the transmission channel. Whether QUIC protocol can effectively resisting to such attacks is worth investigating. In this work, we demonstrated the extreme vulnerability of QUIC under WFP attacks by comparing attack results under well-designed conditions. We also study the transferability of features, which enable the adversary to use proven effective features on a special protocol attacking a new protocol. This study shows that QUIC is more vulnerable to WFP attacks than HTTPS in the early traffic scenario but is similar in the normal scenario. The maximum attack accuracy on QUIC is 56.8% and 73% higher than on HTTPS utilizing Simple features and Transfer features. The insecurity characteristic of QUIC explains the dramatic gap. We also find that features are transferable between protocols, and the feature importance is partially inherited on normal traffic due to the relatively fixed browser rendering sequence and the similar request-response model of protocols. However, the transferability is inefficient when on early traffic, as QUIC and HTTPS show significantly different vulnerability when

*Corresponding author
Email addresses: zhanpengwei@iie.ac.cn (Pengwei Zhan), wangliming@iie.ac.cn (Liming Wang), ytang@gzhu.edu.cn (Yi Tang)

considering early traffic. We also show that attack accuracy on QUIC could reach 95.4% with only 40 packets and just using simple features, whereas only 60.7% when on HTTPS.

Keywords: website fingerprinting, QUIC, encrypted traffic

1. Introduction

TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) are two standard transport-layer protocols in the TCP/IP protocol suite. TCP provides reliable transmission, so packets are transmitted without error, loss, and duplication. Therefore, applications that require reliable connections, e.g., accessing web resources (with HTTP), are implemented base on TCP. However, with the increase of bandwidth and complexity of the network environment increases, problems such as Head-of-line blocking (HOL blocking) and re-transmission ambiguity significantly affect TCP's transmission performance. QUIC is a UDP-based cryptographic protocol with built-in TLS function and optimized multiplexing, flow control, and congestion control mechanism, which solves TCP transmission performance shortcomings. QUIC, which equal to HTTP+TLS+UDP, can achieve the same or better transmission efficiency as HTTPS (equal to HTTP + TLS + TCP) in most network conditions[1] and provides security equivalent to TLS[2].

Meanwhile, confidentiality has become the dominant trend of the Internet. As of January 2021, all of the top 100 non-Google websites support cryptographic protocol, and 97 of which use encrypted protocol by default[3]. Encrypted communication directly causing the traditional methods such as deep packet inspection fails to sniff the payload of the packets and better protect users' privacy. However, the extent to which privacy can be protected is worth exploring.

Existing encryption protocols, including QUIC and HTTPS, do not wholly overturn the application layer's working principle while implementing encrypted communication. Instead, a function suite that can perform encryption is added to the existing application layer. Encryption does not significantly change the size of web resources, and the request-response HTTP traffic triggered by browser rendering of web pages is highly correlated with the structure of website, making WFP attack based on traffic analysis possible. An adversary can analyze the website fingerprint to infer the website that the user is visiting. Website fingerprint refers to a set of metrics (e.g., num-

ber of packets, packets inter-arrival time) extracted from the traffic between client and server when the user is accessing a website. Website fingerprinting (WFP) attacks try to infer the websites that a user is visiting by learning the correspondence between websites and website fingerprints, using pattern recognition and machine learning techniques. Besides, an attack will escalate to an early WFP attack if implemented using incomplete traffic, which will bring higher security risks than a normal WFP attack.

Although QUIC is based on UDP instead of TCP, it does not change the HTTP request-response model structure. Moreover, browsers all roughly follow a rendering paradigm, and the browser's rendering causes the traffic generated by the request-response model. This will result in that the transporting sequence of resources remains roughly in same when transmitted via different protocols and may introduce associations between traffics in different transporting protocols. Moreover, these undesired associations may extend the vulnerabilities present in one protocol to another.

In this paper, we develop a study of website identification under QUIC traffic from the perspective of traffic analysis. We first define some general traffic features, collect early traffic of QUIC and HTTPS, and study website identification efficiency on the early traffic. We then explore the feature transferability between QUIC traffic and ordinary HTTPS traffic. We constructed a comprehensive testbed to collect qualified traffic data. Two sets of features are designed for control purposes, and five machine learning algorithms are selected to comprehensively evaluate the vulnerability of QUIC and HTTPS under WFP attacks. It is proved that QUIC is much more vulnerable to WFP attacks than HTTPS on early traffic scenario, and features transferring is flexible between protocols under normal traffic scenario.

The main contributions of this paper are as follows:

- We implement WFP attack on QUIC and HTTPS, finding that QUIC is more vulnerable to WFP attacks than HTTPS in the early traffic scenario but is similar in the normal scenario. Even computationally cheap and simple features are highly effective when attacked QUIC.

- We quantitatively analyzed the latent feature representation space on QUIC and HTTPS, finding that features can represent inter-class and intra-class data more efficiently on QUIC than HTTPS, intuitively showing why QUIC is more vulnerable than HTTPS.

- We performed transfer studies between QUIC and HTTPS, showing

that, when on normal traffic, features are transferable between protocols, and the feature importance is inherited; however, it is inefficient when on early traffic due to the different magnitudes of variation in the traffic distribution of protocols.

- We implement upgrade WFP attacks on QUIC and HTTPS, exposing the highly insecure of QUIC in the real deployment: with only 40 packets, attack accuracy on QUIC reaches 95.4%, whereas 60.7% when on HTTPS.

The rest of this paper is organized as follows. In Section 2, we discuss related work in the paper's context. We introduce QUIC, website fingerprint, and WFP attacks in Section 3. In Section 4, we detail our testbed and how we collect data. We explain the data processing operations and define our designed features in Section 5. In Section 6, we present experiment metrics and discuss experimental results on the vulnerability of protocols and thetransferability of the features. In Section 7, We discuss the limitations of the experiment and future work. In Section 8, we conclude this paper.

2. Related Works

In this section, we provide an overview of related work on WFP attacks. Related work was divided into WFP on unencrypted traffic and WFP on encrypted traffic based on the protocol studied. Besides, some important work on QUIC was discussed.

Website fingerprinting on unencrypted traffic. Karagiannis et al.[4] propose the BLINC algorithm that analyzes host behavior and successfully identify eDonkey, MSN, IRC, NNTP, and SSH traffic. Bar-Yanai et al.[5] identify the traffic of Skype, eDonkey, and BitTorrent through a hybrid algorithm of k-means and KNN, which analyze the flow-based features of different applications and achieved approximately 99.1% accuracy. Lakhina et al.[6] adopt an entropy-based method and clustering method to distinguish regular traffic from abnormal traffic. Gu et al.[7] divided packets into 2348 categories according to functions and destination port numbers, and then performed maximum entropy analysis on the packet classes distribution. Furthermore, a behavior-based model to detect traffic anomalies was established, reaching F_1 above 0.93.

Website fingerprinting on encrypted traffic. As early as 1996, Cheng et al.[8] have researched encrypted traffic analysis attacks. They used

the HTML file size and requested traffic size to identify websites. Liberatore and Levine[9] use NaiveBayes and Jaccard's coefficient to attack SSH on a dataset containing 2,000 web pages with a total of 480000 samples. Bernaille et al.[10] adopt a method based on GMM clustering, using the packet size to identify SSL connection. Khakpour et al.[11] use entropy vector estimation, Decision Tree, SVM to identify encrypted traffic. Zhang et al.[12] use an improved k-means algorithm to distinguish between SSH, SSL, and non-encrypted applications. Panchenko et al.[13] propose the CUMUL method, which utilizes 104 features to conduct WFP attacks on the Tor and has achieved more than 91% accuracy on several datasets, including ALEXA100. Dusi et al.[14] use GMM and SVM to identify applications under SSH based on packet size and direction. Tong et al.[15] use Random Forest and convolutional neural networks to identify applications under QUIC, considering Netflow-based and packet-based features, including the size and direction of packets. Five types of Google app services are distinguished with 99.24% accuracy. Hayes and Danezis[16] propose a Random Forset based WFP model called k-fingerprinting. They used features such as the number of packets statistics, incoming and outgoing packet ratios, and packet ordering statistics to identify 30 monitored services on 100,000 web pages, achieved 85% true positive rate, and 0.02% false positive rate.

QUIC. Biswal and Gnawali[17] show that QUIC performs better than HTTP/2 in weak network environments, if the website does not contain many small-sized objects. Megyesi et al.[1] compare the performance of HTTP, SPDY, and QUIC in different network situations, and found that QUIC has better performance under the network with slight packet loss and large RTT. They conclude a Decision Tree to reach the best perform protocol in different network conditions. Cook et al.[18] find that QUIC outperforms HTTP/S in unstable networks (such as wireless mobile networks), but perform similarly in stable and reliable networks. These studies demonstrate the importance of QUIC in improving transmission efficiency.

3. Preliminaries

3.1. QUIC protocol

QUIC is very young, as Google started the development in 2012, and only released the first version in August 2013. QUIC has made huge changes in design concept[2], packet loss recovery and congestion control mechanisms[19],

encryption details[20], and flow control mechanisms[21] to address the performance bottlenecks of TCP. QUIC only requires userspace support, making it possible to deployed easily without changing the Internet middleware[2]. About 5.1% of websites support QUIC, including Youtube and Google Translate[22]. When users use a browser that supports QUIC, such as Chromium and Chrome, QUIC will be tried first. Moreover, it will elegantly downgrade to HTTPS if the QUIC handshake fails.

3.2. Website fingerprinting

A website is a specified webpage W which can be viewed as a Web resource set $R = \{r_i\}$ of Web resources. To render this page, a user agent, in general a browser, will fetch those resources and generate web traffic T, $T = Traffic(R) = \{(t_j, d_j)\}$, where t_j is a j-th packet and d_j is the direction of client-to-server or server-to-client. A feature F is an m-dimension vector, $(f_1, f_2, ..., f_m)$, that characterizes the traffic T, and is denoted by $F = Feature(T)$. Ideally, the traffic fingerprinting of website W can be represented by F in a singe visit because of the visit traffic T coming from W. However, dynamic network communications and parallel resource downloads makes the same website generate different traffic in different visits and the feature from a a single visit cannot characterize the webpage traffic features. It needs an algorithm to extract a more general feature from traffics in n different visits. When no ambiguity is possible, we reuse the notation F and have $F = Feature(T_1, T_2, ..., T_n)$. We call this F as the fingerprinting of website W.

Define $Early(T, k)$ as a function that fetch the first k packets from T, i.e., $Early(T, k) = \cup_{j=1}^{k}\{(t_j, d_j)\}$. It is obviously that $Early(T, k)$ is in fact the early traffic of website W. Following the notation in previous paragraph, we have a feature $Early(F, k)$ which is also an m-dimension vector that characterizes the early traffic $Early(T, k)$ and also an early fingerprinting of website W, $Early(F, k)$, comes from early traffics in n different visits.

3.3. Website fingerprinting attacks

Suppose an adversary wants to infer a specified website W visiting by users. The adversary may reside in intermediate nodes such as the routers or the switches to eavesdrop the network traffic. Since the page W is transported in encrypted over QUIC or HTTPS, the adversary cannot identify what contents are hidden in traffics because of the limited decryption power.

The widely used tunnels, proxies, and CDNs make the IP addresses in traffics may not be the real addresses of the visiting websites. It seems that the privacy of users' visiting behaviors could be well-preserved when visiting over encryption protocols.

However, QUIC and HTTPS, the primary application secure protocol for website visiting, are based on the request-response model. It implies that it is possible to distinguish traffics from different web resources because of the relatively fixed browser rendering sequence. Furthermore, the encryption of web resources is on block ciphers, making the sizes between encrypted resources and plain ones are not changed significantly. And thus, WFP attacks could be launched.

Suppose there are N websites the user may visit, the adversary needs to continually maintain a fingerprint series $S = \cup_{i=1}^{N}\{F_i\}$ or $S = \cup_{i=1}^{N}\{Early(F_i, k)\}$ depended on normal or early traffic scenario on the eavesdropping process, the purpose of the adversary is to learn the parameters λ of the attack model M from S. Then a normal WFP attack and a WFP attack on early traffic can each be represented as:

$$M(F, \lambda) \longrightarrow W \tag{1}$$

$$M(Early(F, k), \lambda) \longrightarrow W \tag{2}$$

According to the definition we have given, the specific WFP attack process is divided into two stages, as shown in Figure 1.

(I) **Training**. Adversary passively eavesdrops on the user's exit-side network traffic, from which website fingerprints are generated and feed to the attack model (Figure 1(a)).

(II) **Attacking**. Adversary generates website fingerprint from specific traffic and predicts the related website by asking the well-trained attack model(Figure 1(b)).

Even if the information that can help identifying the website cannot be sniffed directly from the traffic, the attack model and website fingerprint will expose the website, causing the privacy benefited from encryption damaged.

3.4. Website fingerprinting attack on early traffic

The browser engine determines the page's display, and the same page may be displayed differently on different browsers. However, although modern browsers may use different kernels (e.g., Google Chrome: Blink; Mozilla

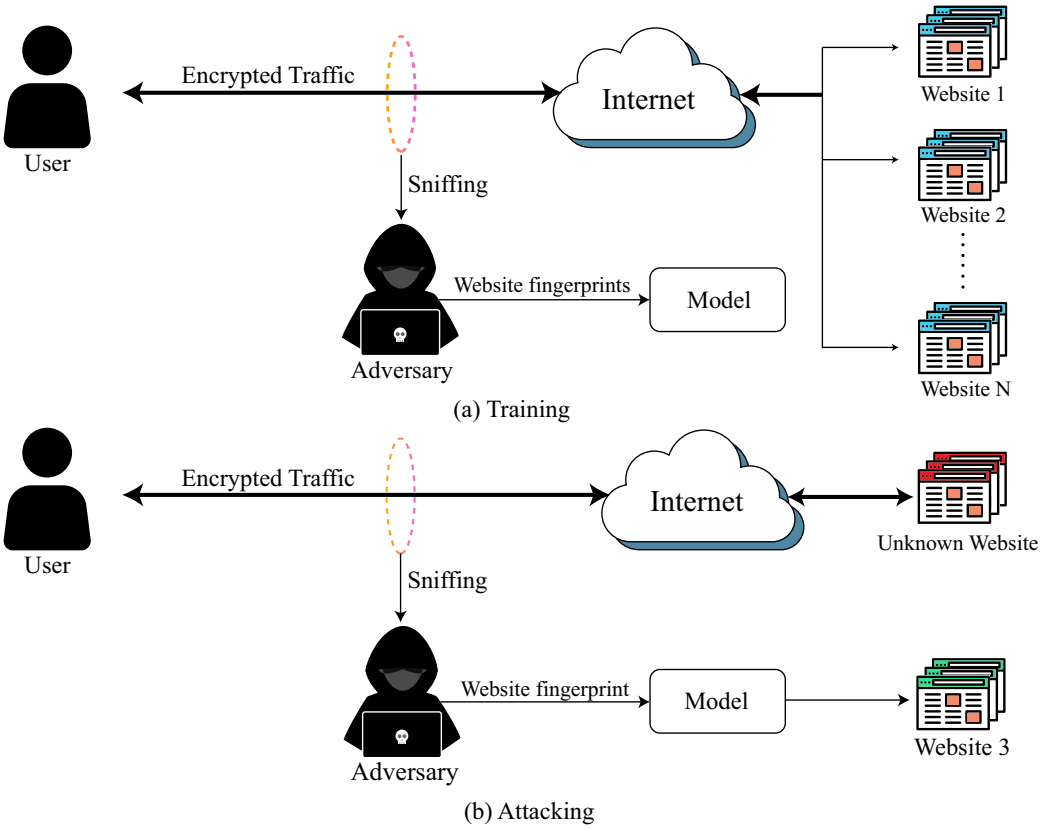

Figure 1: Process of website fingerprinting attack

Firefox: Gecko; Safari: WebKit; Internet Explorer: Trident), they all follow a similar paradigm when requesting resources and rendering web pages, and a complete page rendering process consists of the following steps:

1. Obtain the HTML file from the server.
2. Parse CSS tags and JavaScript tags in HTML files
3. Obtain the CSS file and JavaScript file from the server according to the HTML file's parsing result.
4. Parse the CSS file and JavaScript file and then combine the HTML file.
5. Parse the image and video tags in the HTML file.
6. Obtain media resources from the server and render the web page.

Differences in traffic between websites are caused from step (1) onwards, and different HTML file structures can directly cause significant differences

in subsequent steps. The difference in traffic increases when considering the packets generated in more steps. A regular WFP attack uses the total traffic from step (1) through step (6) to build the website fingerprint (from start to end of entire traffic), but an attack on early traffic, instead, use traffic from the first few steps (from start to k-th packet of traffic).

The effect of WFP attack on early traffic is getting closer to normal WFP attack as the parameter k increases. Intuitively, an attack on early traffic can be performed more quickly than a normal attack, since it does not need to wait for the total traffic to be transfer and computation is cheaper as fewer packets are considered. A WFP attack should complete when the website visited by the user is just finished initializing, enable the adversary to implement further attacks such as RST attack in TCP, making connection between client and server blocked without the user's awareness when the targeted website is visited. Therefore, a WFP attack on early traffic is more threatening than a normal WFP attack.

3.5. Assumptions

We refine the assumptions made in the previous section, which ensure a fair comparison of QUIC and HTTPS and guarantee that extraneous factors do not confound our conclusions on vulnerability and feature transferability. The assumptions have been divided into three areas: User, Adversary, and Website.

- **User.**
  - **Closed-world**: The page a user may visit is limited to N pages. This assumption greatly reduces the size of S that the adversary needs to maintain, ensuring that every T generated by the user can be collected to update the maintained S, with no additional filtering required by the adversary. Conversely, users can access pages other than the N target pages in the open-world scenario.
  - **Sequentially visiting**: The user will only access websites sequentially. Only one page is accessed at a time, and the next page is accessed only when the previous page is closed, i.e., the last packet completed transmission. This may differ from the average user's browsing behavior, but it ensures that the website fingerprints are pure.
- **Adversary**.

  - **Traffic parsing ability**: An adversary can eavesdrop on the user to collect traffic on the user's exit-side and detect the start position and end position of traffic. However, the end position of traffic is not needed to identify on early traffic scenario but the k-th packet position.
  - **Normal computing power**: An adversary can complete the construction of the website fingerprint within an acceptable time delay, and cannot decrypt or modify the packet.

- **Website**.
  - **Same-origin resource limit**: All target pages only contain same-origin resources, including CSS scripts, JavaScript scripts, images and video resources, etc.

4. Data Collection

To address QUIC and HTTPS's vulnerability under the WFP attack and the transferability of features, we required a dataset consisting of encrypted traffic that can eliminate the impact of distribution differences in data, and simulate the condition that a real user is visiting web servers. The dataset must (i) large enough to reflect the pattern in a real-world (dynamic network communications and parallel resource downloads happens), (ii)contains aligned paired encrypted QUIC and HTTPS traffic, which originated from the same group of resources, and (iii) be collected like what an actual adversary would attempt. In this section, we present the details of our testbed and the processes we collect data.

4.1. Testbed setup

To collect qualified traffic data, we establish a comprehensive testbed. The overall architecture of the testbed is shown in Figure 2.

The entire testbed is established under a controlled environment, and we do not sniff traffic data from the Internet directly. Instead, we sniff from the local area network for (i) it can prevent individual functional packets (e.g., RST packets in TCP and RST_STREAM frames in QUIC) from interfacing the regular traffic, and ultimately ensures the reliability of experimental results and (ii) only a few websites support both QUIC and HTTPS, and complete details of these sites are difficult to obtain. We expect to collect

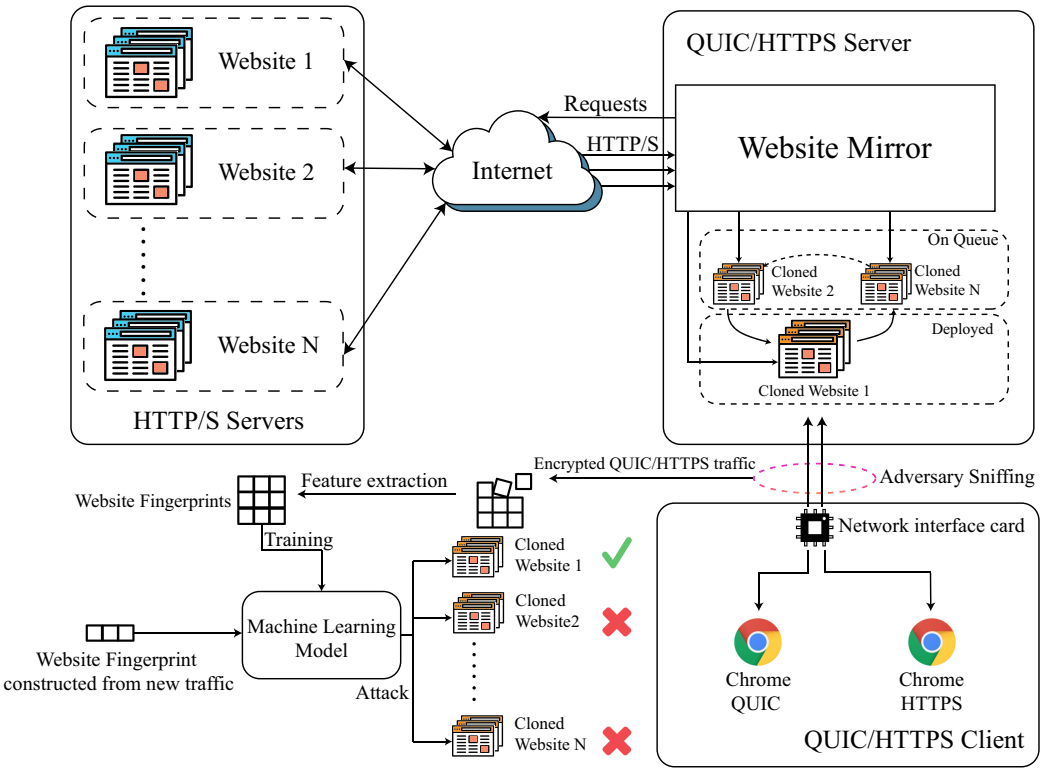

Figure 2: Architecture of established testbed

aligned paired QUIC and HTTPS traffic originating from the same group of resources, avoiding the distribution differences of two protocols traffic.

We select the official landing page of the top 100 schools in the 2019 TIMES World University Rankings[23] as our closed-world. We first use HTTrack Website Copier[1] to clone the same-origin resources of the target website (e.g., A.com/figure.jpg is the same-origin resource of A.com, while img.A.com/figure.jpg is not.) with depth set as 2 or 3 (ensuring the traffic length of a single visit is long enough) to the local QUIC Server, a machine with Ubuntu 16.04 installed. Cloned websites resources include HTML files, CSS files, JavaScript files, pictures, and other media files involved in the entire process of accessing the landing page.

[1] https://www.httrack.com/

QUIC/HTTPS Server is a host running Caddy Server[2] that supports both QUIC and HTTPS communication. We use scripts to deploy all websites in rotation, each time only a single website is deployed, and the rest are in the queue, this can ensure that users obey sequentially visiting rule. Another host in the same local area network is taken as the QUIC/HTTPS client, on which we utilize Selenium[3] to drive Chrome automatically to imitate actual user behavior when visiting a website. Expressly, Chrome is set to disable cache mode to prevent interference from previous access and enable QUIC mode is set when website transfer via QUIC. Specifically, a complete visit process has the following steps:

1. Selenium starts a new Chrome process;
2. Chrome access the website currently being deployed;
3. Chrome wait for the landing page to load completely;
4. Chrome wait 5 seconds after completion, and then close the current process;

We define 100 consecutive visits as a sniffing cycle. We visit each website via QUIC and HTTPS, respectively, for 100 times, and retaining utilizing Wireshark to sniff the traffic on the exit-side of the client's network interface card to simulate the adversary behavior in WFP attack (Figure 1). Collected traffic data is saved as pcap files, and each file contains all traffic obtained in a sniffing cycle.

4.2. Collected Data

7.07G websites are cloned to localhost with 92 websites are available after deleting invalid and error pages; therefore, N in our experiments is 92. Table 1 shows basic information about the three smallest and largest websites.

The traffic of each website obtained ranges from 4.5M to 1.86G. The final traffic data size is 45.19G, of which 22.51G is from QUIC, and 22.68G is from HTTPS. HTTPS consumes more traffic than QUIC when accessing the same set of resources. 736000 samples were extracted in total under 40 different k for two protocols.

[2] https://caddyserver.com/
[3] https://chromedriver.chromium.org/

Table 1: Information of collected websites

Name	URL	Size	Number of files
USC	www.usc.edu	1.9 M	88
UW-Madison	www.wisc.edu	2.1 M	62
UniMelb	www.unimelb.edu.au	2.5 M	97
...	...	...	...
TUM	www.tum.de	458.2 M	2502
Monash	www.monash.edu	1592 M	1661
SKKU	www.skku.edu	2536 M	9977

5. Feature Extraction

The effectiveness of the WFP attack is directly related to extracted features. Simple and powerful features help to improve attack efficiency and effectiveness. However, the differences between protocols introduce uncertainty in the transferring of features between protocols, and the knowledge discovered on HTTPS may be incapable of improving the attacks on QUIC. In this paper, we designed two groups of features, including Simple features and Transfer features, to explore the vulnerability of QUIC and HTTPS and the transferability of features under both early and normal scenario.

5.1. Traffic tailoring

Chrome drove by Selenium in "disable cache" mode will start a new process on each visit, making the client and server always establish a connection as the first-time connection. The handshake process of QUIC differs from HTTPS in the first-time connection. QUIC requires 1-RTT to complete the QUIC handshake. The client will first generate the handshake parameters with a new Connection ID (CID) and send them to the server in a *Client Hello* packet, which represents the beginning of a QUIC Handshake. At the end of the QUIC handshake, the client sends the second *Client Hello* packet. The first-time handshake process of HTTPS is divided into TCP Handshake and SSL Handshake, and the whole process consumes at least 3-RTT. A *Change Cipher Spec* packet from the client indicates the end of the SSL handshake and the complete HTTPS handshake. To avoid influence caused by the different handshake process, we tailor traffic from the first packet after the handshake to the last packet to close the current connection (in HTTPS, the RST packet close the current connection; in QUIC, a new

CID indicates the end of the previous connection). Traffic used to generate website fingerprint is limited to the encrypted data traffic part (Figure 3).

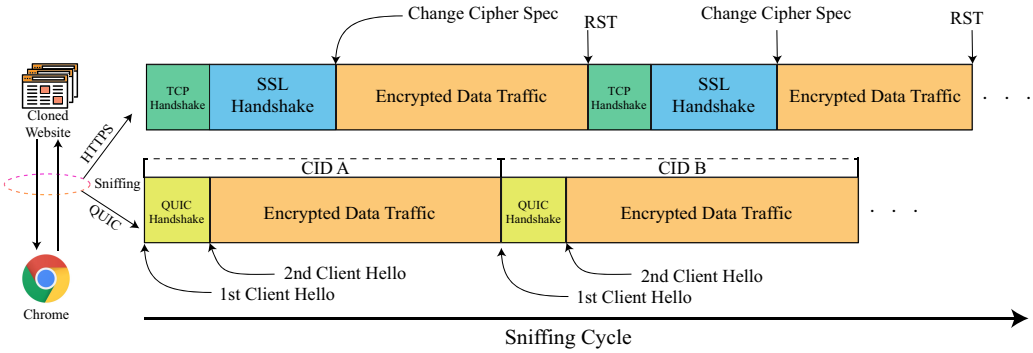

Figure 3: Illustration of traffic tailoring

5.2. Simple features

The WFP attacks on early traffic can infer the website that the user is visiting much faster than regular WFP attacks. However, the WFP attack's efficiency, especially in the early traffic scenario, will decrease if the attack features are complicated and computationally expensive. We design a group of simple but informative features, which are only related to the packet's direction and size, defined as Simple features. In this section, we detail the Simple features.

First, we divide packets into two categories based on their direction d:

- Positive: packets from the client to the server.
- Negative: packets from the server to the client.

Second, we divide the packets into four categories based on their size:

- Tiny: packets size less than 79 bytes.
- Small: packets size range from 80 – 159 bytes.
- Medium: packets size range from 160 – 1279 bytes.
- Large: packets size over 1280 bytes.

We divide all packets into eight categories, by combining direction features with size features. They are *Positive Tiny, Negative Tiny, Positive Small, Negative Small, Positive Medium, Negative Medium, Positive Large*, and *Negative Large*. The number of packet in each category form the Simple features. Specifically, T can be characterized with $F = Feature(T) = (n_{pt}, n_{nt}, n_{ps}, n_{ns}, n_{pm}, n_{nm}, n_{pl}, n_{nl})$, where n represents the number of different kind of packet in the eight categories.

These features do not require much calculation: when a packet is collected, it can be classified into the proper categories immediately by just reading the packet header's information. Moreover, the calculation all focus on packet counting, which meets our expected low-cost computing characteristics. In experiments, we also found that although this set of features is simple, it contains a wealth of information in the WFP attacks on QUIC traffic. This set of features is also a set of controls relative to the intricate features presented subsequently to illustrate that the vulnerability of the protocol is not caused by the defects of specific features.

5.3. Transfer features

In previous studies, many features have been proven effective in traffic identification and WFP attack on TCP-based protocols, such as unique packet size[9], packet size count[24], and packet order[25]. Features used for HTTPS can be divided into five levels: Packet-level, Burst-level, TCP-level, Port-level, and IP-level[26]. Some effective features are selected among Packet-level and Burst-level, since TCP-level, Port-level, and IP-level features are excluded for QUIC and HTTPS respectively based on TCP and UDP, and our experiment focuses on a single connection between two hosts each time. Selected proven effective features form the Transfer features, which are specifically defined as:

- **unique packet size**: this feature statistics weather t in length l, ignoring d, is occur in T. Specifically, define $Length(t)$ as a function that calculate the length of packet t, if $l \in \{Length(t_i)|t_i \in T\}$, l-th dimension of this feature is set to 1, otherwise, is set to 0. This feature is a 1460-dimension vector (packet length of QUIC and HTTPS range from 54 to 1514).

- **packet size count**: this feature statistics the number of t in length l, ignoring d, in T. Specifically, if $l \in \{Length(t_i)|t_i \in T\}$, l-th dimension

of this feature is set to $Card(\{t_i|t_i \in T, Length(t_i) = l\})$. This feature is a 1460-dimension vector.

- **packet order**: this features record the packets length in order of packet position. Specifically, the i-th dimension of this feature is set to $Length(t_i)$, where t_i is the i-th packet in T. This feature is a k-dimension vector.

- **inter-arrival time**: this feature statistics arrival interval of adjacent packets in order of packet position. Specifically, define $Time(t)$ as a function that fetch the arrival time of t, let t_0 be the 2-nd *Client Hello* packet for QUIC, and the *Change Cipher Spec* packet for HTTPS, then l-th dimension of this feature is set to $(Time(t_i) - Time(t_{i-1}))$, where $t_i, t_{i-1} \in T$. This feature is a k-dimension vector.

- **negative packets**: this feature statistics the number of packet t with d is negative in T. This 1-dimension feature is set to $Card(\{(t_j, d_j)|d_j = negative\})$.

- **cumulative size**: this feature statistics the cumulative size of packets in T. This 1-dimension feature is set to $\sum\{T_p, T_n\}$, where $T_p = \{Length((t_i, d_i))|t_i \in T, d_i = positive\}, T_n = \{Length((t_i, d_i))|t_i \in T, d_i = negative\}$.

- **cumulative size with direction**: this feature statistics the cumulative size of packets in T, but the impact of packet direction d is considered. This 1-dimension feature is set to $\sum\{T_p, T_n\}$, where $T_p = \{Length((t_i, d_i))|t_i \in T, d_i = positive\}, T_n = \{-Length((t_i, d_i))|t_i \in T, d_i = negative\}$.

- **bursts numbers/maximal length/mean length**: burst is define as the consecutive packets between two packets sent in the opposite direction[27]. Bursts numbers, bursts maximal length, bursts mean length is the statistical features based on burst.

- **total transmission time**: this features statistical the total transmission time of T. This 1-dimension feature is set to $\sum\{Time(t_i) - Time(t_{i-1})|t_i \in T, i > 1\}$.

This group of features is defined as Transfer features, proven originally effective in the attack on HTTPS. This set of features is not conducive to handling high-speed network flows but can adequately expose the vulnerability of the protocol, as they are informative but also computationally expensive.

6. Evaluation

In this section, we first introduce the metrics and the basic settings used in the experiment. Then, We conduct fair comparisons between QUIC and HTTPS and tested the vulnerability of the protocol under different scenarios under WFP attacks. Furthermore, we analyze the transferability of the features. Finally, we illustrate the high risk exposed by QUIC in real deployments with an escalating attack.

6.1. Evaluation metrics

The purpose of the WFP attack is to infer the website that the user is visiting, which can be seen as a multi-classification task, an aggregation of multiple binary classification tasks. In our balanced dataset, accuracy is the most intuitive measurement. Higher accuracy means a better attack performance and a more vulnerable protocol. Precisely, accuracy is calculated as follows:

$$\text{Accuracy} = \frac{n_c}{n_t} \tag{3}$$

Where, n_c is the total number of correct predictions, and n_t is the total number of instances.

6.2. Experimental setting

In this paper, we do not focus on the improvement of attack algorithms, and five standard machine learning algorithms are used as the baseline model of WFP attacks. They are *Random Forest (RF), Extra Trees (ET), K-Nearest Neighbors (KNN), Naive Bayes (NB)* and *Support Vector Machine (SVM)*. The algorithms are implemented with scikit-learn[4], using the default parameters. We took 40 different k values from 5 to 200 in steps of 5 to simulate different levels of attack scenario (early $\rightarrow$ normal). We specifically define early traffic scenario as conditions that $k \leq 40$, and normal scenario

[4] https://scikit-learn.org

when $k > 40$. The given experimental results are obtained through 10-fold cross-validation.

6.3. Vulnerability of protocols

Information that can be represented by the features is not saturated in early traffic scenario, and the main factor affecting the attack effectiveness becomes the information richness of the features as k increasing. The attack accuracy increases with parameter k when Simple features are used in the attack (Figure 4(a), Figure 4(b)), as a larger k capture more significant differences between websites. When k becomes large enough, the attack accuracy will not continue to increase but will fluctuate steadily over a range, which is also the performance upper bound of a normal WFP attack.

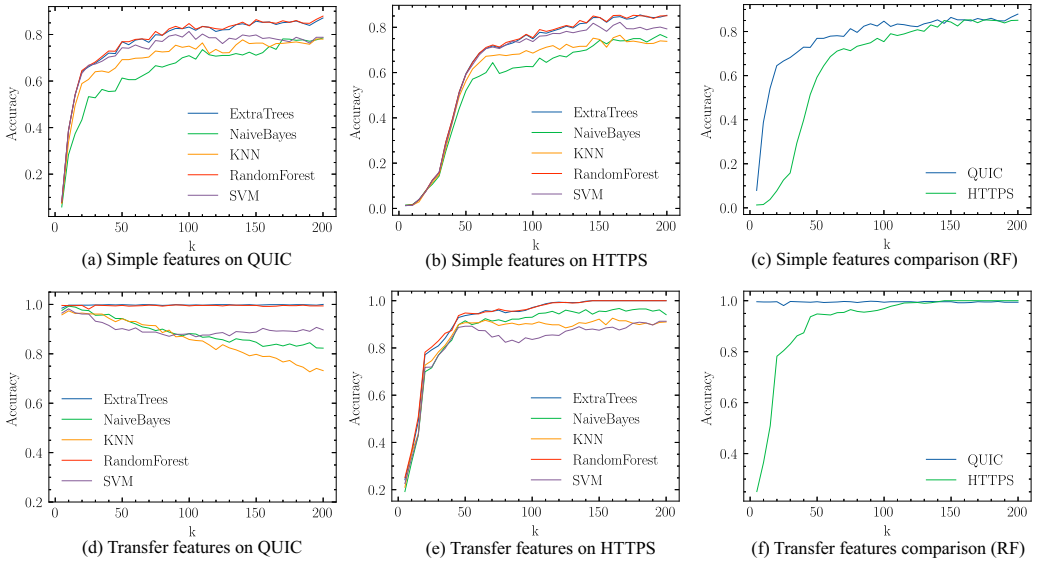

Figure 4: Attack accuracy of website fingerprinting attacks

Tree-based models like RF and ET outperformed the five selected attack algorithms on both QUIC and HTTPS. When the WFP attacks implemented on QUIC, utilizing Simple features, RF and ET respectively reach an average accuracy of 77.3% and 76.8% (Figure 4(a)), and, when on HTTPS, 65.0% and 64.7% (Figure 4(b)). When using Transfer features, tree-based models also show significant advantages, and their accuracy increases with k as the accuracy of other algorithms slightly decreases when attack on QUIC. Especially for the attack on QUIC, the accuracy of tree-based models is maintained at

extremely high levels (RF: 99.49%; ET: 99.78%, average, Figure 4(d)), and RF and ET reach an average accuracy of 91.60% and 91.30% (Figure 4(e)) when attacks on HTTPS.

QUIC is more vulnerable to WFP attacks than HTTPS in the early traffic scenario but is similar in the normal scenario, and the gap manifests itself when using both Simple features and Transfer features. We find that the accuracy of the attack on two protocols is similar when k is large (e.g., Simple/Transfer, $k = 200$, 87.9% / 99.4% on QUIC, 85.1% / 100% on HTTPS, 2.8% / 0.6% difference, Figure 4(c)). However, the attack performance shows a large difference when the k is small (e.g., Simple / Transfer, $k = 20$, 64.5% / 99.6% on QUIC, 7.7% / 78.2% on HTTPS, 56.8% / 21.4% difference, Figure 4(c)). The small k indicate early traffic scenario, and the gap introduce the high security risks of QUIC when even only few packets are available. By comparing the result on both simple and complex features, we show that the vulnerable in early traffic scenario is not caused by defects of features, but the attributes of the protocol itself. Focus on the performance of the attack on QUIC, and we find that attack can still achieve good performance when only a few packets are considered, even if the features are sample and computationally cheap (Simple feature, $accuracy \leq 72.9\%$ when $k \leq 40$).

Feature importance on HTTPS fluctuates dramatically when k is small, indicating that HTTPS is k-sensitive, while QUIC is not (Figure 5). We define k-sensitive as: if a protocol is k-sensitive, it will show more different vulnerabilities when the number of packets considered (k) is different. Feature importance fluctuates dramatically with increasing k when k is small. i.e., the features considered by the attack algorithm change considerably, indicating that the traffics distribution of the protocol changes considerably, which directly leads to the difficulty of attacking the protocol changes when k is different. The feature importance is always more stable when the attack implements on QUIC than on HTTPS (Simple features: QUIC, $\sigma^2 = 1.275E-04$; HTTPS, $\sigma^2 = 1.736E-03$. Transfer feature: QUIC, $\sigma^2 = 9.940E-05$; HTTPS, $\sigma^2 = 1.081E-03$), and fluctuates dramatically at small k and only stabilizes when the k is large enough when the attack implement on HTTPS. In the early traffic scenario, the average attack accuracy of RF on QUIC is 55.34% and is 13.98% on HTTPS; In the normal scenario, is 82.74% on QUIC and 77.82% on HTTPS. As scenario change, the maximum difference in attack accuracy is 27.40% on QUIC and up to 63.84% on HTTPS. This phenomenon is consistent with the difference in the degree of fluctuation of feature importance.

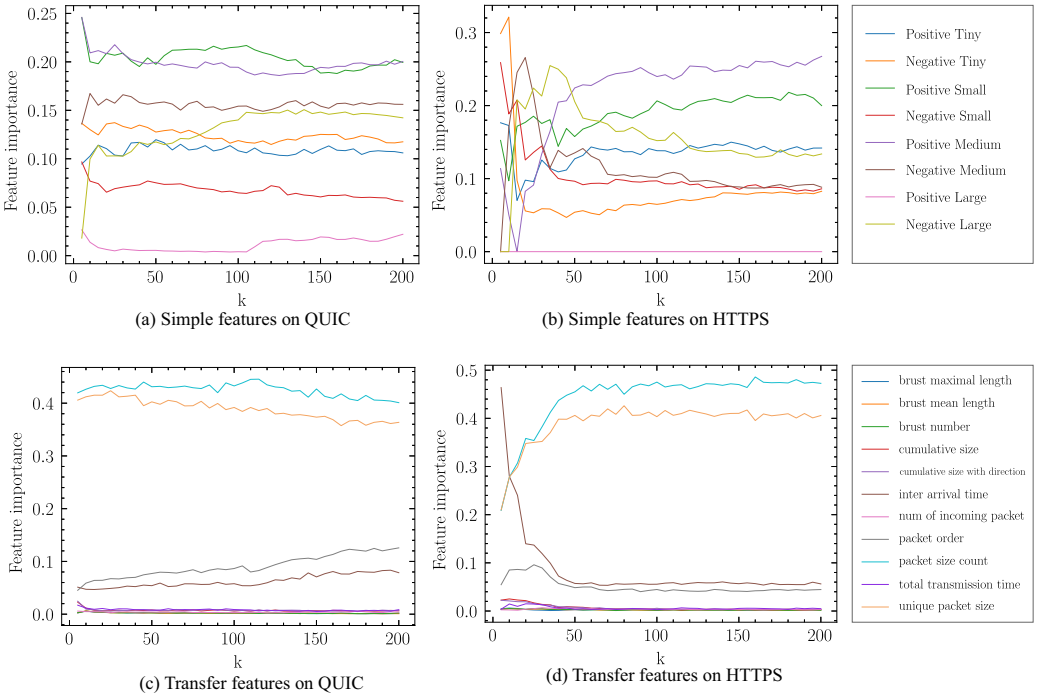

Figure 5: Feature importance of Simple features and Transfer features

6.4. Qualitative Analysis of Latent Representation

In this section, we will qualitatively analyze the vulnerability differences between the two protocols in terms of the latent representation of features on QUIC and HTTPS. The qualitative analysis, combined with the quantitative analysis in the previous section, will provide a more comprehensive description of the nature of the protocol vulnerability. To display the representation's distribution pattern, we randomly selected 10 out of 92 classed of websites and analyzed only the sampled data. Because the degree of latent representation variation of the feature over the two protocols will be more considerable when k is small and will gradually stabilize as k increases, we selected five k values with unequal differences to better represent the process of variation. We will use the Transfer features as the original representation for the protocol because it exposes the protocol's fragile characteristic more completely. T-distributed stochastic neighbor embedding (T-SNE) was used to extract latent representations of the protocol on the sampled data at fixed k values (Figure 6).

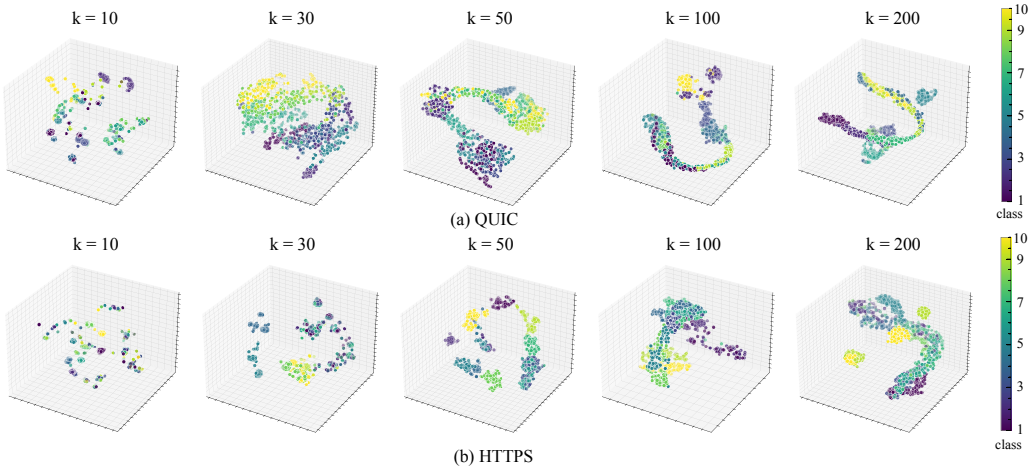

Figure 6: Visualization of the latent representation of Transfer features on two protocols

When $k = 10$, i.e., in the extremely early traffic scenario, the latent representations of both QUIC and HTTPS have many features collapsed together, which indicates that different classes of data are difficult to be represented clearly, thus illustrating the difficulty of conducting WFP attacks on early traffic. However, even in this case, the representations of QUIC is more of aggregationes of same classed, while the representations of HTTPS is more of aggregationes of different classes. This indicates that QUIC, in the extremely early scenario, although the intra-class gap cannot be fully learned yet, the inter-class gap's adequate representation has laid the groundwork for the attack's high success rate. When $k = 30$, as the increased number of packets considered, the Transfer feature can better express the inter-class disparity. There is a clear tendency of dispersion in the QUIC representation, and all features are more spreading, which indicates that the latent representation space of QUIC can distinguish the inter-class data better, making the protocol more vulnerable to attacks. The representation on HTTPS also shows the same trend, but the magnitude of change is relatively small, and most of the features are still too aggregated to be represented, which illustrates the defensive of HTTPS against WFP attacks. When $k = 50$, the representation space of QUIC has almost convergence, and when k continues to increase, the relative positions between points in the space are very similar, which mirrors the results obtained when the Transfer feature and RF are used to attack QUIC when $k > 50$. It should be noted that there is still a large mixture of

data of different classes in the HTTPS representation space. When $k > 100$, the representations spaces of QUIC and HTTPS show similar characteristics, and the similarities and differences between intra-class data and inter-class data are well represented, indicating that the vulnerabilities of QUIC and HTTPS are similar under the normal scenario, which is consistent with the results of the quantitative analysis in previous.

6.5. Transferability of features

We have explored the vulnerability of QUIC and HTTPS to WFP attacks. In this section, we will further explore the transferability of features in the view of attack accuracy and feature importance, also we expect a new perspective to explain the vulnerability gap between the two protocols under attack. As we have excluded the distribution differences of data in our experiments (aligned traffic of different protocols come from the same set of resources), we are able to compare the effectiveness of features on different protocols accurately.

The information carried by the features determines the effectiveness of the features, and this effectiveness is similar across protocols when on the normal scenario but is different when on the early traffic scenario(Figure 7). In normal scenarios, features proven effective in the attack on one protocol are also effective on others due to the relatively fixed browser rendering sequence and the similar request-response model. However, as QUIC and HTTPS show different sensitivity to k, transferability is inefficient when on early traffic (Figure 4,5,6). Effective features expose the vulnerability of the protocol more completely, as evidenced by (i) an increase in the upper bound of the attack accuracy, i.e., the accuracy in normal scenarios (Simple feature, QUIC: 87.9%, HTTPS: 85.3%; Transfer feature, QUIC: 99.8%, HTTPS: 100%). (ii) fewer packets are needed to achieve high accuracy ($k = 5$; QUIC: Simple feature: 7.9%, Transfer feature: 99.6%; HTTPS: Simple feature: 1.3%, Transfer feature: 25.1%).

Feature importance is inherited partly across protocols when on the normal scenario, and there are no subversive changes in feature importance when features are transferred between protocols, i.e., the most crucial feature to one protocol will not become the last to another. This indicates that the resulting change in classification accuracy while the feature set remains is not the result of the combination of features but the unique feature itself. The effect of inheritance is ineffective in the early traffic scenario, as the distribution of HTTPS traffic is much more variable than QUIC. We rank

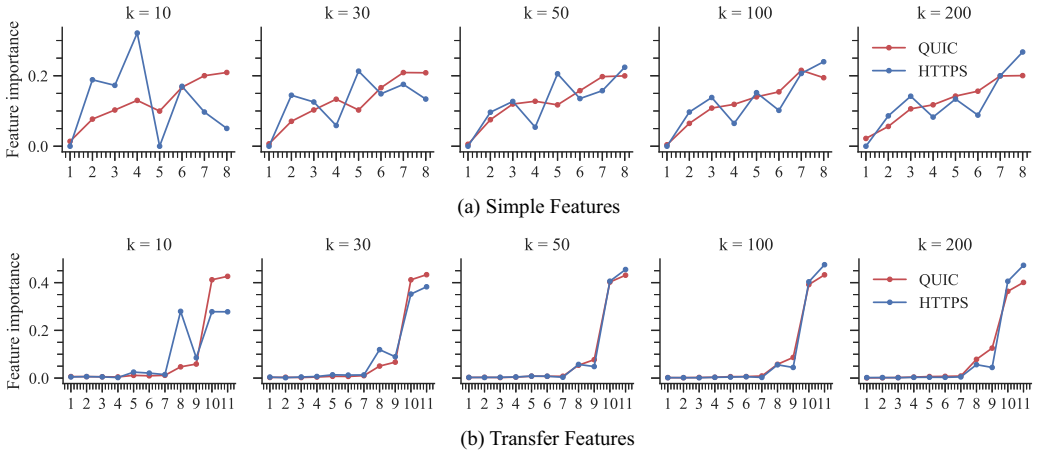

Figure 7: Inheritance of feature importance between two protocols

the features on QUIC by feature importance at $k = 200$ and fix the order of the features. The curves of feature importance on QUIC and HTTPS are plotted in the obtained feature order for different k, as shown in Figure 7. It can be found that the feature importance changes proportionally over the two protocols when $k \geq 30$. However, because HTTPS is $k-$sensitive when k is small, the difference in features importance on the two protocols is more extensive than when the value of k is large. Specifically, when on normal scenario, *Positive Small* and *Positive Medium* in Simple features rank top 2 in the attack on both QUIC and HTTPS, and *Positive Large* always contribute a little (Figure 5(a)(b)). Similarly, *packets size count, unique packets size, inter-arrival time*, and *packet order* are significant in both attacks, ranking top 4 among Transfer features while the importance of other features are close to zero (Figure 5(c)(d)). However, *Negative Tiny* (Figure 7(a), feature 4) and *inter-arrival time* (Figure 7(b), feature 8) shift when on early traffic scenario.

6.6. Website Fingerprinting Top-a Attacks

In this section, we expect to illustrate the high risk that QUIC may have in case of a real attack. Top-a attack simulates real-world adversary behavior, and the adversary could perform further attacks if the first attack fails to infer the target website, i.e., recall is important than precision for adversary. Top-a attacks mean the adversary will predict a websites with the highest probability as the target website. Specifically, Top-a attack is defined as:

$$M(F, \lambda) \longrightarrow A \tag{4}$$

$$M(Early(F, k), \lambda) \longrightarrow A \tag{5}$$

Where, $A = \cup_{i=1}^{a}\{\{W_i | i \in \{1, 2, ..., N\}, P(W_1) > P(W_2) > ... > P(W_N)\}\}$, $P(W_i)$ is the vising probability of W_i given by the model. A Top-a attack is counted as success when the website accessing by the user belongs to the prediction set, i.e., $W \in A$. It should be noted that upgrading from a WFP attack to a Top-a attack does not require additional calculations, ensuring the efficiency does not drop, even dealing with a high-speed stream.

The adversary should prefer to use low computational features during the attack as a way to reduce the latency of the attack, thus saving more time to prepare for further attacks. And since the attack's accuracy on QUIC fail to reflect the differences between protocols under Top-a attack when using Transfer features and RF (accuracy are high enough due to the complexity of features, i.e., always close to 100% when on QUIC), we will experiment with simple and effective Simple features while using the most effective RF as the attack algorithm.

When the attack is upgraded to a Top-a attack, the weaknesses of QUIC are further exposed in two ways (Table 2). First, Top-a attacks benefit more when implemented over QUIC than over HTTPS. As the parameter a increases ($1 \rightarrow 5$), the attack accuracy improves more on QUIC (average, QUIC: 27.0%; HTTPS: 14.7%), and at the most significant improvement, the improvement on QUIC is almost two times as the improvement on HTTPS (QUIC, 41.2%; HTTPS: 22.9%). This attack effectiveness improvement is independent of baseline's attack accuracy, i.e., accuracy when $a = 1$, but is determined by the characteristic of QUIC itself. Even attacks has similar accuracy on both QUIC and HTTPS when $a = 1$, the Top-a attack on QUIC always achieves a higher accuracy improvement (e.g., QUIC: k=10, accuracy=38.7%, improve: 41.2% v.s. HTTPS: k=40, accuracy=39.7%, improve: 21.0%). This vulnerability of QUIC makes it especially dangerous in real-world deployment, where an adversary can always achieve a significant accuracy improvement within an acceptable tolerance even when the attack accuracy is originally low. Second, Top-a attacks can achieve high accuracy on QUIC with a tiny number of packets, which introduces a critical risk to the actual deployment of QUIC. The accuracy gap reaches largest when using only 10 packets, and the Top-5 attack can identify the website currently

Table 2: Results of Top-a attacks with Random Forest

Protocol	k	a					improve
		1	2	3	4	5	$(1 \rightarrow 5)$
QUIC	5	0.079	0.14	0.183	0.22	0.260	0.181
	10	0.387	0.583	0.682	0.757	<u>0.799</u>	**0.412**
	15	0.542	0.702	0.786	0.839	0.872	0.330
	20	0.645	0.788	0.852	0.887	0.911	0.266
	25	0.666	0.810	0.875	0.912	0.93	0.264
	30	0.682	0.806	0.871	0.906	0.927	0.245
	35	0.705	0.838	0.888	0.925	0.941	0.236
	40	0.729	0.852	0.910	0.943	**0.954**	0.225
HTTPS	5	0.013	0.024	0.039	0.053	0.068	0.055
	10	0.015	0.031	0.043	0.058	<u>0.072</u>	0.057
	15	0.038	0.074	0.096	0.121	0.145	0.107
	20	0.077	0.112	0.149	0.182	0.219	0.142
	25	0.126	0.186	0.224	0.259	0.289	0.163
	30	0.158	0.229	0.296	0.342	0.387	**0.229**
	35	0.294	0.375	0.434	0.477	0.507	0.213
	40	0.397	0.501	0.558	0.581	**0.607**	0.210

being transmitted over QUIC with an accuracy of 79.9%, while the accuracy of the attack over HTTPS is only 7.2% under the same conditions. Moreover, when the attack reaches the best performance, using only 40 packets, attack accuracy on QUIC could reach 95.4%, whereas only 60.7% when on HTTPS.

7. Discussion

The superior transmission performance of QUIC protocol brings opportunities for speeding up the Internet, but its security risks bring uncertainties. We are the first to study the vulnerability of QUIC to WPF attacks on early traffic. Previous research[15, 28] explored the website fingerprinting of QUIC, but their work was on a normal scenario, while we focus more on the risk exposed by QUIC on early traffic. The vulnerability of QUIC on early traffic poses a significant challenge to the privacy and confidentiality guaranteed. We also find that features can transfer between QUIC and HTTPS, but the transfer is ineffective when on early traffic.

One limitation of our work is that we have made restrictions to make the traffic pure enough, whereas pure traffic may not be available in the open world. Our experiments are performed under closed-world, sequentially visiting, same-origin resource limitations, and traffic is tailored, all of which assumption and operation ensure that the traffic is not affected by network condition, background traffic or special functional packets. However, those limitations also enable us to conduct fairly comparisons between QUIC and HTTPS and reach more general conclusion. Since our research focuses on the vulnerability between QUIC and HTTPS under WFP attacks, rather than the attack performance against a single protocol, we believe that the conclusions we have achieved are feasible and valid.

We plan to improve our experiments under weaker constraints for future work and expect to draw conclusions about QUIC, and HTTP/3, in the open world. In the meantime, we think it is also significance to explore how features are transferred and applied across different protocols, which may help us to construct generalizable attack models.

8. Conclusion

Cryptographic protocol can protect the user's privacy and avoid exposing private information to the adversary. This paper discussed QUIC and HTTPS's vulnerability and the feasibility of feature transferring between

protocols under both early traffic and normal scenarios. We demonstrated that QUIC is more vulnerable to WFP attacks than HTTPS in the early traffic scenario but is similar in the normal scenario. We also quantitatively analyzed the latent feature representation space on QUIC and HTTPS to intuitively showing that features can represent inter-class and intra-class data more efficiently on QUIC than HTTPS, causing QUIC is more vulnerable than HTTPS on early traffic scenario. Besides, we confirmed that, when on normal traffic, features are transferable between protocols, and the feature importance is inherited; however, it is inefficient when on early traffic due to the different magnitudes of variation in the distribution of protocols. We also show that an adversary can always achieve a significant accuracy improvement within an acceptable tolerance even when the attack accuracy is originally low on QUIC, exposing the highly insecure of QUIC in the real deployment.

Acknowledgments

This work was supported by National Key R&D Program of China under grant NO.2018YFB0803604.

References

[1] P. Megyesi, Z. Krämer, S. Molnár, How quick is quic?, in: 2016 IEEE International Conference on Communications (ICC), IEEE, 2016, pp. 1–6.

[2] R. Hamilton, J. Iyengar, I. Swett, A. Wilk, et al., Quic: A udp-based secure and reliable transport for http/2, IETF, draft-tsvwg-quic-protocol-02 (2016).

[3] Google, Google transparency report (2020).
URL https://transparencyreport.google.com/https/overview

[4] T. Karagiannis, K. Papagiannaki, M. Faloutsos, Blinc: multilevel traffic classification in the dark, in: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications, 2005, pp. 229–240.

[5] R. Bar-Yanai, M. Langberg, D. Peleg, L. Roditty, Realtime classification for encrypted traffic, in: International Symposium on Experimental Algorithms, Springer, 2010, pp. 373–385.

[6] A. Lakhina, M. Crovella, C. Diot, Mining anomalies using traffic feature distributions, ACM SIGCOMM computer communication review 35 (4) (2005) 217–228.

[7] Y. Gu, A. McCallum, D. Towsley, Detecting anomalies in network traffic using maximum entropy estimation, in: Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement, 2005, pp. 32–32.

[8] H. Cheng, R. Avnur, Traffic analysis of ssl encrypted web browsing, URL citeseer. ist. psu. edu/656522. html (1998).

[9] M. Liberatore, B. N. Levine, Inferring the source of encrypted http connections, in: Proceedings of the 13th ACM conference on Computer and communications security, 2006, pp. 255–263.

[10] L. Bernaille, R. Teixeira, Early recognition of encrypted applications, in: International Conference on Passive and Active Network Measurement, Springer, 2007, pp. 165–175.

[11] A. R. Khakpour, A. X. Liu, An information-theoretical approach to high-speed flow nature identification, IEEE/ACM transactions on networking 21 (4) (2012) 1076–1089.

[12] M. Zhang, H. Zhang, B. Zhang, G. Lu, Encrypted traffic classification based on an improved clustering algorithm, in: International Conference on Trustworthy Computing and Services, Springer, 2012, pp. 124–131.

[13] A. Panchenko, F. Lanze, J. Pennekamp, T. Engel, A. Zinnen, M. Henze, K. Wehrle, Website fingerprinting at internet scale., in: NDSS, 2016, pp. 1–15.

[14] M. Dusi, A. Este, F. Gringoli, L. Salgarelli, Using gmm and svm-based techniques for the classification of ssh-encrypted traffic, in: 2009 IEEE International Conference on Communications, IEEE, 2009, pp. 1–6.

[15] V. Tong, H. A. Tran, S. Souihi, A. Mellouk, A novel quic traffic classifier based on convolutional neural networks, in: 2018 IEEE Global Communications Conference (GLOBECOM), IEEE, 2018, pp. 1–6.

[16] J. Hayes, G. Danezis, k-fingerprinting: A robust scalable website fingerprinting technique, in: 25th USENIX Security Symposium (USENIX Security 16), 2016, pp. 1187–1203.

[17] P. Biswal, O. Gnawali, Does quic make the web faster?, in: 2016 IEEE Global Communications Conference (GLOBECOM), IEEE, 2016, pp. 1–6.

[18] S. Cook, B. Mathieu, P. Truong, I. Hamchaoui, Quic: Better for what and for whom?, in: 2017 IEEE International Conference on Communications (ICC), IEEE, 2017, pp. 1–6.

[19] Quic loss recovery and congestion control (2016).
URL https://tools.ietf.org/html/draft-tsvwg-quic-loss-recovery-01

[20] A. Langley, W.-T. Chang, Quic crypto, Google, Revision 20161206 (2016) 2016.

[21] Google, Flow control in quic (2016).
URL https://docs.google.com/document/d/1F2YfdDXKpy20WVKJueEf4abn_LVZHhMUMS5gX6Pgjl4/edit

[22] Usage statistics of site elements for websites (2021).
URL https://w3techs.com/technologies/overview/site_element

[23] World university rankings 2019 (2019).
URL https://www.timeshighereducation.com/world-university-rankings/2019/world-ranking

[24] D. Herrmann, R. Wendolsky, H. Federrath, Website fingerprinting: attacking popular privacy enhancing technologies with the multinomial naïve-bayes classifier, in: Proceedings of the 2009 ACM workshop on Cloud computing security, 2009, pp. 31–42.

[25] M. Crotti, M. Dusi, F. Gringoli, L. Salgarelli, Traffic classification through simple statistical fingerprinting, ACM SIGCOMM Computer Communication Review 37 (1) (2007) 5–16.

[26] J. Yan, J. Kaur, Feature selection for website fingerprinting, Proceedings on Privacy Enhancing Technologies 2018 (4) (2018) 200–219.

[27] K. P. Dyer, S. E. Coull, T. Ristenpart, T. Shrimpton, Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail, in: 2012 IEEE symposium on security and privacy, IEEE, 2012, pp. 332–346.

[28] S. Rezaei, X. Liu, Multitask learning for network traffic classification, in: 2020 29th International Conference on Computer Communications and Networks (ICCCN), IEEE, 2020, pp. 1–9.