applied sciences

MDPI

Article

F-ACCUMUL: A Protocol Fingerprint and Accumulative Payload Length Sample-Based Tor-Snowflake Traffic-Identifying Framework

Junqiang Chen [1,2,3], Guang Cheng [1,2,3,*] and Hantao Mei [1,2,3]

[1] School of Cyber Science and Engineering, Southeast University, Nanjing 211189, China
[2] Jiangsu Province Engineering Research Center of Security for Ubiquitous Network, Nanjing 211189, China
[3] Purple Mountain Laboratories, Nanjing 211189, China
* Correspondence: chengguang@seu.edu.cn

Abstract: Tor is widely used to protect users' privacy, which is the most popular anonymous tool. Tor introduces multiple pluggable transports (PT) to help users avoid censorship. A number of traffic analysis methods have been devoted to de-anonymize these PT. Snowflake is the latest PT based on the WebRTC protocol and DTLS encryption protocol for peer-to-peer communication, differing from other PT, which defeat these traffic analysis methods. In this paper, we propose a Snowflake traffic identification framework, which can identify whether the user is accessing Tor and which hidden service he is visiting. Rule matching and DTLS handshake fingerprint features are utilized to classify Snowflake traffic. The linear interpolation of the accumulative payload length of the first n messages in the DTLS data transmission phase as additional features are extracted to identify the hidden service. The experimental results show that our identification framework F-ACCUMUL can effectively identify Tor-Snowflake traffic and Tor-Snowflake hidden service traffic.

Keywords: Tor; pluggable transports; traffic identification; hidden service

Citation: Chen, J.; Cheng, G.; Mei, H. F-ACCUMUL: A Protocol Fingerprint and Accumulative Payload Length Sample-Based Tor-Snowflake Traffic-Identifying Framework. *Appl. Sci.* **2023**, *13*, 622. https://doi.org/10.3390/app13010622

Academic Editor: Stefan Fischer

Received: 30 November 2022
Revised: 20 December 2022
Accepted: 29 December 2022
Published: 2 January 2023

1. Introduction

With the growing concern for privacy among Internet users, multiple anonymous techniques have been devoted to implement anonymous access, such as traffic obfuscation padding, multi-layer encryption, complex path transmission, and other means. Tor [1] is the most popular anonymous communication system, which owns 6114 active nodes with 2.93 million users and the data bandwidth transmitted of about 670 Gbit/s [2]. Tor also provides hidden services (HS) [1], which provides anonymity protection for the service side of the network using multi-hop reverse proxy or resource-sharing storage technology. The great anonymity provides a breeding ground for some illegal activities, such as drugs, firearms, human trade, hacking services, etc. [3], and poses a great obstacle to network regulation. To tackle the negative impact of Tor, researchers have begun to use traffic analysis extensively. Traffic analysis attempts to judge if a user is connecting to Tor or what service he is accessing by analyzing the user's traffic passively, which has become one of the approaches with the most potential to de-anonymize Tor [4].

In order to improve the anonymity and help users in censorship zones to access Tor, Tor introduced many integrate pluggable transport (PT) technologies, including Obfs4 [5], Meek [6], and so on. Users can obfuscate their traffic and secretly connect to Tor, thus getting around censorship. Several traffic analysis methods [7–10] have achieved success on de-anonymizing PT, such as Obfs4, Meek, etc. Snowflake, based on WebRTC [11] technology, is the latest release of PT, which can establish an encrypted connection between the Tor client and Tor network, and few researchers have worked on it yet. WebRTC is an advanced open-source protocol framework that is widely used to build multimedia transmission tunnels between browsers, which is a peer-to-peer real-time communication. Some

censorship circumvention methods bypass the regulation by modifying and inputting users' data into multimedia protocol tunnels [12,13], which have attracted increasing attention from researchers in recent years. WebRTC is widely used in many real-time communication applications (e.g., Facebook Messager, Google Hangouts, the gaming-focused chat Discord, etc.), thus the censorship cannot block Snowflake by blocking instances of WebRTC communication.

Snowflake is the latest release of Tor PT, which transmits traffic through the WebRTC data transmission channel and applies UDP as the transport layer framework protocol. As an emerging PT technology which effectively bypasses censorship, Snowflake is being used by an increasing number of users, but there is little research on traffic identification for it. Due to Snowflake being based on the UDP protocol for data transmission, and other popular Tor PT such as Obfs4 and Meek being based on TCP for data transmission, most of the existing traffic identification methods for Obfs4 and Meek do not apply to it. In addition, Snowflake relies on the WebRTC protocol framework for communication circuit construction, and the traffic generated will be largely confused with other WebRTC-based applications. These bring great challenges to the supervision. In this paper, we analyze the Snowflake protocol and propose a Tor-Snowflake traffic identification framework based on rule matching and DTLS handshake fingerprint features. Our framework can identify whether a user is accessing Tor with Snowflake, and can also make a determination about whether a user is accessing HS with Snowflake. The key contributions of this work are as follows:

- By analyzing the communication process and system construction of the Snowflake, we propose the rule-matching technique to pre-identify snowflake traffic quickly. Beyond this, we raise an accurate Snowflake traffic identification method based on the DTLS handshake fingerprint.
- Then, we propose a hidden service identification method based on cumulative length and statistical features. Specifically, we combine the statistical features and the cumulative load length sampling of messages in the DTLS data transmission phase, which in line with the difference in circuit establishment between Tor normal web access and hidden service access. Then, a classifier identifies if a Tor user is accessing a hidden service.
- Our experiments show that our proposed method can accurately identify Snowflake traffic with little time consumption, and efficiently identify Tor-Snowflake hidden services traffic.

The remainder of the paper is organized as follows. Section 2 discusses related work in Tor and Tor PT traffic identification, whereas Section 3 introduces our threat model and Snowflake communication principle. Architecture of the Tor-Snowflake traffic identification framework are described in Section 4, and a more detailed experimental evaluation is reported in Section 5. Section 6 concludes the paper.

2. Related Work

In this section, we first introduce the related work on Tor traffic analysis, and then describe the existing research on Tor PT traffic identification. We briefly introduce them as follows.

1. Tor traffic analysis

Many works have been devoted to Tor traffic analysis in recent years. They attempt to determine whether users are using Tor and what service or content they are accessing by passively observing user traffic. Lashkari et al. [14] presented a time-based method to characterize and identify Tor traffic within a flow. They confirm that time-based characteristics of traffic can be used to characterize Tor traffic and efficiently distinguish between different Tor application traffic. Montieri et al. [15] used the hierarchical classification (HC) framework on the public anonymous dataset Anon17 [16] to classify three famous anonymous tools' traffic (i.e., Tor, I2P, and JonDonym). They conducted experiments to analyze traffic

at different grain from three different levels, and the highest level of application category obtained a gain of F-measure up to +4.5% with respect to a similar work [17].

WF (Website Fingerprint) has also become an important branch of Tor traffic analysis. Panchenko et al. [18] proposed a novel website fingerprinting attack in an open-world scenario, and tested it on a huge real-world representative dataset. Their approach has a better performance than all state-of-the-art methods in classification accuracy and consumes less computational resources. They explored the limits of WF at the Internet scale. Rimmer et al. [19] proposed a novel deep learning (DL)-based WF attack method, which can automatically perform the feature engineering process. This DL-based WF approach overcomes the failure of traditional manual feature engineering to cope with the new changes introduced to the Tor network. They evaluated the method on the largest ever dataset of WF and found that the best DL model performed with over 2% higher accuracy than the state-of-the-art attack method [18]. In another work, Sirinam et al. [20] leveraged a convolutional neural network to construct a WF attack model and evaluated it against state-of-art defenses (e.g., WTF-PAD and Walkie-Talkie). Results showed over 98% accuracy on non-defended Tor traffic, and achieved more than 90% accuracy when employing WTF-PAD [21]. However, only 49.7% accuracy is achieved when applying Walkie-Talkie [22]. This method is still effective in open-world scenarios.

2. Tor PT traffic identification

Traffic analysis brings a huge threat to Tor's anonymity; therefore, Tor has developed some censorship-resistant PT tools to help people against traffic attacks in the Tor network. Many efforts have attempted to de-anonymize PT traffic. Wang et al. [23] presented the first comprehensive investigation of the detectability of five obfuscated traffic PT tools used in Tor and constructed a framework to show that censors can reliably identify these obfuscation tools with a sufficiently low false-positive rate. Shahbar et al. [24] applied flow-based traffic analysis to evaluate the detecting performance of Tor PT. Adopting a C4.5 classifier, they found that the use of Tor PT was distinguishable, although some PTs changed the distribution of traffic or mimicked other traffic. Guan et al. [7] studied PT-based Tor traffic identification under SSH-encrypted tunnels. The experimental results show that the availability and untraceability of tunneled Tor traffic can be reduced through traffic analysis based on machine learning algorithms.

Obfs4 is one of the most popular PTs. Through the analysis of the obfs4 communication process, He et al. [8] proposed a two-stage identification method combining coarse-grained filtering based on the randomness of message, time series of handshake phase and message length distribution with fine-grained identification based on the SVM algorithm for high-precision identification of Obfs4 traffic, and achieved 99% identification accuracy on their own experimental datasets.

Meek is another popular PT on which much work has been done. Starting from the connection characteristics of Meek, He et al. [9] combined static and dynamic packet features, and performed analysis on the Meek flow fragmentation mechanism. They extracted relevant features and achieved over 97% accuracy and 99% recall using the SVM machine learning algorithm. Wang et al. [10] proposed a method using deep learning methods to automatically learn and extract communication fingerprints using key packet sequences for the efficient identification of Meek-based Tor hidden services access activities. The method can significantly reduce the identification time and storage consumption and achieve a better identification accuracy.

Snowflake, an emerging Tor PT based on WebRTC connections, has few research works on it. Fifield et al. [25] analyzed differences in WebRTC instances by manual fingerprinting and experimentally discovered the possibility of classifying WebRTC applications using fingerprinting methods. They pointed out the limitations of Snowflake in bypassing censorship and indicated future directions for using WebRTC to evade censorship. The threat modeling and circumvention of censorship are discussed by the same author in [26]. They also discussed the design of Snowflake and WebRTC fingerprint information. S. Frolov et al. [27] collected a wide range of TLS traffic in real-world data, and used these

data to analyze TLS implementations of several popular censorship circumvention tools, including Snowflake. From the collected fingerprint information, Snowflake has some fingerprint features that significantly distinguish it from other applications. These works are fragmented presentations for Snowflake or analyses of the possibility of Snowflake identification by fingerprinting.

In summary, current research has focused more on the identification and classification of Tor traffic than Tor PT traffic. With the improvement of the censors' abilities, more and more users prefer using Tor PT for anonymous communication. Research on Tor PT will be a trend in the future. Previous work has provided direction and reference for our research, but some existing methods are no longer applicable due to the development and changes of Tor-Snowflake. Some of the works [10,28] identify hidden service access behavior under Tor PT through key message sequences, but the start and end locations of key messages are difficult to determine and can significantly affect the effectiveness of the identification. In this paper, we study the emerging Tor PT Snowflake, for which there is little research available on traffic identification and the identification of hidden service traffic.

3. Background

3.1. Threat Model

We begin by showing our threat model in Figure 1. We assume that the opponent can passively observe user traffic, such as a network censor located at an ISP or with control of a large-scale gateway. Censored users, who have normal access to web content, also tend to evade censors through encryption, steganography, tunneling, anonymization, mutation, morphing [29], and other methods. In the supervised network, the censor monitors the packets transmitted in the network and obtains packets' information without decrypting them. They have certain computational and storage capacity to process traffic messages or flow-related information. Then, they use statistical classification, machine learning, deep learning and other methods to determine the user's behavior.

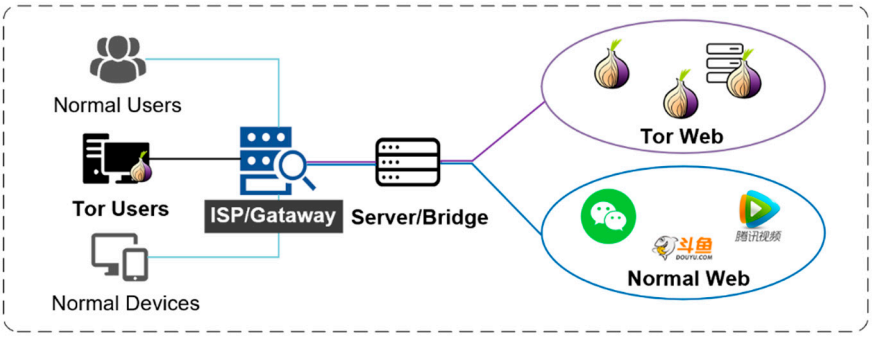

Figure 1. Threat model, we assume the opponent is an ISP or major gateway.

3.2. Communication Principle of Snowflake Protocol

The Pluggable Transport (PT) technology can obfuscate Tor traffic into the back-ground traffic by cryptographic padding, domain fronting, or tunneling. Snowflake is a state-of-the-art Tor PT based on WebRTC, built on the base of Flashproxy [30]. It also includes the domain fronting in its communication process. Snowflake is more like a hybrid of previous PT technologies, providing easy-to-use access to Tor for Tor users in the censorship region. Snowflake is based on establishing WebRTC connections between peers and sending access requests to the Tor network via browsers volunteering outside the censorship area. WebRTC, the core technology that Snowflake relies on, is a web framework and suite of protocols that support peer-to-peer connections between browsers. In the following section, we will introduce the communication principle of Snowflake from two aspects: the communication process of Snowflake and the UDP-based encrypted data transmission protocol DTLS (Datagram Transport Layer Security).

3.2.1. Snowflake Communication Process

Figure 2 illustrates the communication process of Snowflake. Snowflake consists of three main components. The snowflake client is a Tor client configured with the snowflake PT plugin, which is responsible for initializing service requests and establishing peer-to-peer connections with snowflake proxies. The snowflake proxy is a small in-browser WebRTC proxy that establishes a connection with the snowflake client by WebRTC and relays Tor traffic to the Tor network. The broker is responsible for rendezvous and is used to complete the online matching between the client and the snowflake proxies, which usually requires domain-fronting techniques for establishing communication. Snowflake uses these components to negotiate communications and build WebRTC connections to hide and forward traffic to the Tor network. Snowflake's communication process can be divided into three main phases: the preparation phase, the rendezvous phase, and the WebRTC connection establishment phase.

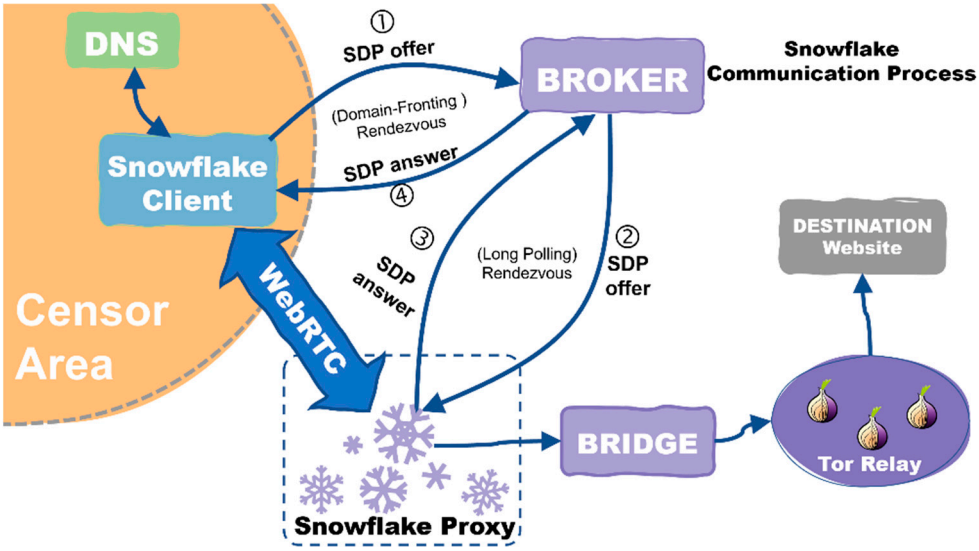

Figure 2. Snowflake communication process.

1. Preparation phase

From the traffic of Tor-Snowflake visited websites collected in the experiment, we found that the Snowflake client initiates DNS requests for a specific set of STUN domains before it starts communicating. After getting the corresponding DNS response, it will wait for a while and then initiate a DNS request for a specific domain-fronting server. Finally, based on the IP address of the domain-fronting service, the subsequent rendezvous phase is conducted. We refer to the process of a Snowflake client initiating a DNS request for a specific set of domains as the preparation phase for Snowflake communication. The discovery of this phase supports our subsequent proposal of the rule-based pre-identification method for Tor-snowflake traffic.

2. Rendezvous phase

The second stage in the Snowflake communication process is the rendezvous phase. In this phase, the snowflake client and proxy exchange information necessary for a WebRTC connection via the broker. The snowflake client establishes an HTTPS connection with the broker through domain-fronting technology. Domain-fronting technology [31] is an application layer technology that hides the real destination address of HTTP messages wrapped inside HTTPS by using different domain names at different communication protocol levels.

In the rendezvous phase, the snowflake proxy and snowflake client establish connection with the broker, respectively. By sending a POST request to the broker, both parties

exchange SDP (Session Description Protocol) information, which can support the establishment of WebRTC peer connection. The SDP data in the rendezvous phase is transmitted encrypted via TLS, so they cannot be extracted by message analysis. The detailed process is referred to in Figure 2.

3. WebRTC connection establishment phase

The last phase of Snowflake communication is WebRTC (Web Real Time Communication) connection establishment. WebRTC is a set of protocols and technology combination for negotiating bidirectional secure real-time communication between two WebRTC peers. It is mostly used for P2P real-time communication in audio and video application scenarios. WebRTC connection establishment consists of four processes: signaling exchange, connection establishment, secure encryption, and point-to-point communication.

The signaling exchange process is the rendezvous phase mentioned above. It presents the following process in the network traffic. After sending a DNS response to the domain-fronting servers [31], the snowflake client establishes a TLS connection and transfers SDP information through it immediately.

The connection establishment process is essentially NAT traversal using ICE (Interactive Connectivity Establishment) negotiation, which is a combination of the STUN (Session Traversal Utilities for NAT) [32] and TURN (Traversal Using Relays around NAT) protocols to build peer-to-peer connections. Snowflake uses the STUN protocol to achieve NAT traversal.

In a secure encryption process, Snowflake builds a data-channel based on the WebRTC DTLS protocol to achieve secure data transmission. DTLS is similar to the TLS protocol, except that DTLS is based on UDP as the transport layer. For a detailed analysis of the DTLS protocol communication process, see Section 3.2.2.

In the peer-to-peer communication process, SCTP (Stream Control Transmission Protocol) is used to send and receive DTLS-encrypted data-channel messages between two WebRTC peers with secure bidirectional communication.

Snowflake establishes a secure and reliable peer-to-peer communication tunnel between user browsers in the censorship region and volunteer browsers outside the censorship region through the above processes. It passes user-generated Tor traffic to the Tor core network to achieve censorship evasion, which poses a great challenge to the current censorship and regulation against Tor.

3.2.2. DTLS Protocol Analysis

Facing the problems of UDP protocol such as no authentication of both communication parties and no guarantee of reliable message transmission, DTLS provides an end-to-end secure data transmission channel for UDP. It uses PSK (Pre-Shared Key) or ECC (Elliptic Curve Cryptography) to achieve encryption during the handshake process, cookie authentication mechanism and certificate to achieve authentication of both communicating parties, adding sequential number, caching the message segment arriving out of order and retransmission mechanism to achieve reliable transmission. In this paper, we study the Snowflake based on DTLS1.2 [33] in WebRTC for encrypted data transmission. The process of DTLS connection establishment can be divided into two phases: an initial handshake phase and a data transfer phase.

In the initial handshake phase, similar to TLS, the client and server implement a DTLS handshake which transfers multiple types of handshake messages. It allows them to verify their identity and negotiate the keys, passwords, and other cryptographic parameters to be used in the connection. During the data transfer phase, the communication is encrypted with the agreed-upon method and secret.

Among these handshake messages, the Client Hello message and Server Hello message contain a lot of fingerprint information. The Client Hello message contains the highest supported version of DTLS, the handshake protocol length, random numbers, encryption suites supported by the client, and optional extensions. Server Hello contains the DTLS version supported by the server, the cipher suit chosen by the server, a random number, and

an optional extension. They also contain message sequence, fragment offset and fragment length fields. Since the length of the handshake message may exceed the MTU of the UDP datagram (usually limited to 1500 bytes), the handshake message is split into several fragments for transmission through the fragment mechanism. Since DTLS handshakes are not encrypted, some features exist in these fingerprints that can significantly identify different applications.

Snowflake's DTLS fingerprint supports the identification of Snowflake traffic. Snowflake is essentially a tool that attempts to mimic traffic, and it achieves obfuscation by trying to make Snowflake's WebRTC difficult to distinguish from other applications of WebRTC. But there are still some differences. From the study of S. Frolov et al. [27], it is clear that the fingerprint information extracted from Snowflake has high discrimination in the collected fingerprint dataset from the experiment. From the analysis of WebRTC fingerprints of Snowflake by D. Fifield et al. [26], it is clear that these differences are mainly found in the DTLS handshake phase fingerprint information. Therefore, we propose a Snowflake traffic identification method based on fingerprinting information in the DTLS handshake phase.

In the DTLS data transfer phase, both the client and server can transmit encrypted and authenticated data with the negotiated key. Although the content of the communication is encrypted, features such as the payload length, direction and time interval of the transmitted message can still reveal the circuit establishment process. These features' information supports our later-proposed method for identifying HS traffic in the Tor-Snowflake scenario.

4. Architecture of Tor-Snowflake Traffic Identification Framework

In this section, we present a framework that identifies Tor-Snowflake traffic and snowflake-based Tor HS access activity. The framework will be introduced in three parts: the general introduction of the Snowflake traffic identification framework and the pre-identification method, the Snowflake traffic identification method based on DTLS handshake fingerprint information, and the HS access activity identification method in the snowflake scenario based on DTLS data transmission message accumulative payload feature extraction.

4.1. Snowflake Traffic Identification Framework

This section proposes a traffic identification framework for Tor's latest PT Snowflake. The framework of the Snowflake traffic identification and Tor hidden service access activity identification under the Tor-Snowflake scenario is shown in Figure 3.

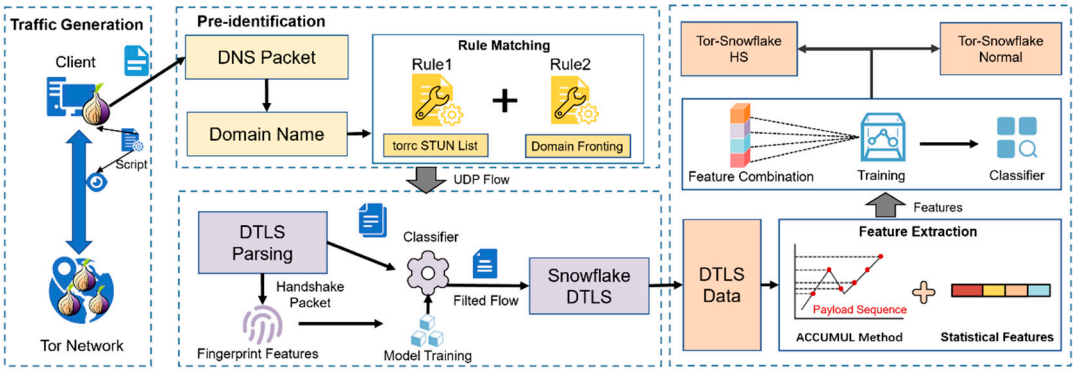

Figure 3. Tor-Snowflake traffic identification framework F-ACCUMUL.

First, we control clients through automated scripts and capture traffic generated from clients into pcap files. We also collect traffic on the controlled gateway. Then, the pre-identification of Snowflake access behavior is performed by a rule-based matching approach. We analyze the pecp files according to the special behavior pattern of DNS query before Snowflake communication. On this basis, we filtered suspicious UDP flow

and extracted handshake fingerprint features of the DTLS protocol. These features were combined and fed into an ML model and trained a classifier, which gives Snowflake's identification result. Furthermore, we extracted relevant statistical features and used our proposed ACCUMUL method to process Snowflake's DTLS flows. In the end, we used the extracted features to train a Snowflake traffic classification model based on machine learning algorithms to identify HS access activities in the Tor-Snowflake scenario.

In the following parts, we analyze the pre-identification method based on the Snowflake communication process from the perspective of Snowflake communication messages. Snowflake utilizes WebRTC technology to establish peer-to-peer connections that require NAT traversal through STUN (Session Traversal Utilities for NAT) [32], which is mostly used in real-time-communication scenarios. Snowflake provides a list of available STUN server domain information, which is usually hard-coded in Tor's configuration file torrc. When the Snowflake client first starts up, it randomly selects a subset of these STUN server lists to initiate DNS requests. It looks for the hostname resolution using both IPv4 and IPv6(DNS 'A' and 'AAAA' records). The set of DNS request messages sent is shown in the Figure 4.

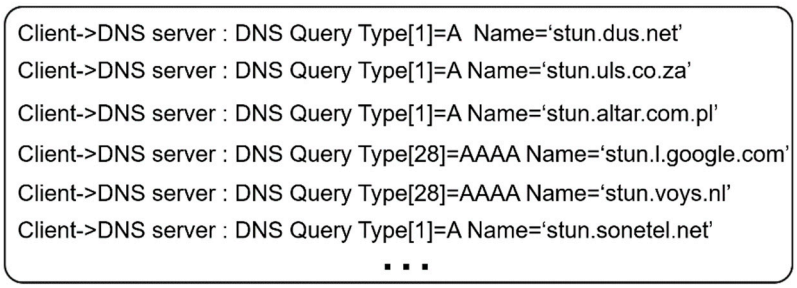

Figure 4. DNS requests in Tor-Snowflake communication.

After completing the DNS query to the STUN server, a DNS request to the domain-fronting server corresponding to the torrc hard-coded by snowflake PT will be sent. After obtaining the response, a TLS connection for the rendezvous phase will be established with the domain-fronting server immediately.

Based on the above analysis, the Tor-Snowflake traffic pre-identification proposed in this paper consists of two main rules. The first rule is to extract the DNS queries initiated by a single user in the packet to STUN servers in a short period of time. If these DNS queries point to multiple STUN servers and these STUN servers are in torrc's Snowflake hard-coded list, the user is probably using Snowflake to try to access Tor, then turn to the second rule of pre-identification. The second rule is based on the first rule, in which after the client gets a DNS response to the above STUN server, the client continues to launch DNS queries to the hard-coded domain-fronting server address in torrc. By the above two rules, the suspicious Snowflake traffic can be filtered out.

4.2. Snowflake Identification Based on DTLS Handshake Fingerprint

Snowflake data transmission relies on WebRTC technology to establish communication tunnels, and it uses DTLS to achieve secure and reliable encrypted data transmission [33]. DTLS reuses a lot of existing TLS code in its design, making the two protocol structures similar, with comparable security mechanisms and protection levels, and ensuring reliable data delivery. Therefore, the method of applying classification for TLS encrypted traffic will also be applicable to the field of DTLS traffic identification to a certain extent.

In this paper, we propose a Snowflake traffic identification method based on DTLS handshake fingerprinting information. This method is summarized from a previous study [26] on the analysis of the TLS used in Censorship Circumvention. The traditional TLS fingerprint-based application identification method usually constructs the extracted fingerprint information into fingerprint tags by the hash algorithm, and identifies the corresponding application by comparing it with a large database of fingerprint tags. We apply

machine learning methods to construct models for the classification and identification of Snowflake by using the extracted fingerprint information as features.

Based on the results of user pre-identification in the first step of the framework, we parse the filtered suspicious traffic, merge UDP flows according to the five-tuple information {srcIP, srcPort, dstIP, dstPort, Protocol}, and extract DTLS message protocols from them. According to the DTLS handshake protocol layer format specification, we parse the DTLS handshake messages and extract fingerprint information from the Client Hello and Server Hello messages, respectively.

For successfully parsed Client Hellos, we extracted the DTLS record version, record length, fragment offset, fragment length, length of cipher suites, list of cipher suites, length of extension, and list of extensions. The above fingerprint information was used to form the fingerprint characteristics of the DTLS Client Hello. In addition, the corresponding server response Server Hello message is parsed, allowing us to see what cipher suite and extensions were negotiated successfully. For each Server Hello message, we parsed the DTLS record version, record length, fragment offset, fragment length, chosen cipher suite, length of extension, and list of extensions. The DTLS handshake fingerprint information we selected to extract is shown in the Table 1.

Table 1. Features selected from DTLS handshake process.

Feature	Client Hello	Server Hello
DTLS record version	✓	✓
Record length	✓	✓
Fragment offset	✓	✓
Fragment length	✓	✓
Cipher suite length	✓	✓
Cipher suites	✓	
Extension Length	✓	✓
Extensions	✓	✓
Cipher suit chosen		✓

For the collected fingerprint features, we use the one-hot-encoding method for pre-processing, which uses N-bit status registers to encode N states. Each state has its own independent register bits, and at any moment, only one bit is valid, which is mostly used in the scenario of distance or similarity calculation between features when dealing with classification, regression, cluster, and other problems. Since with the machine learning Random Forest algorithm it is easy to compare the importance of the impact of different features on the classification results, we chose it to build a multi-classification model to implement Snowflake traffic identification and feature importance ranking based on DTLS fingerprint information features. The specific fingerprint recognition results and evaluation will be presented in Section 5.

4.3. Snowflake Tor HS Access Identification Based on DTLS Data Message

By default, Tor provides anonymity protection for users but does not hide the identity of the servers they visit, so the IP address information of a server can be obtained at the exit node or at a location between the server and the exit node. Tor addresses the potential privacy leaks faced by these service providers with the provision of hidden service. When a Tor user requests a resource that is provided by hidden service, instead of establishing a traditional three-hop circuit to the service provider, the Tor user establishes a connection to rendezvous points through a series of steps, and relays traffic through that node to access the hidden service. It shows that there is a significant difference in circuit construction between Tor hidden service activities and general access activities, and this difference is mainly reflected in the features of transmitted messages.

Tor-Snowflake still follows the communication and forwarding mechanism of Tor. The Snowflake client just establishes communication tunnels with volunteer nodes outside the censorship area with the help of WebRTC technology to achieve access to the Tor

network, which neither changes the operation mechanism of Tor nor affects the circuit establishment process. Therefore, Snowflake communication is the upper layer protocol of Tor communication, similar to the relationship between the UDP protocol and IP protocol. After Snowflake completes the establishment of the communication tunnel with the peer point, it encapsulates the Tor communication data messages generated by the client in the data-channel of WebRTC for transmission. Since there is a difference in circuit establishment between Tor HS activities and general access activities, we analyze the messages in the DTLS data transmission phase of Snowflake PT. We propose a method based on the accumulative total payload sampling of messages in the DTLS data transmission phase, where a fixed number of m additional features are extracted from the accumulated message payload length and combined with common traffic classification features such as statistical characteristics of the communication packets, statistical characteristics of the packets interval time, input–output message number ratio and size ratio to identify Tor- snowflake HS access activities. Since the circuit establishment process exists only at the beginning of the DTLS data transmission phase of Snowflake traffic, the first n messages of this phase are extracted and our proposed feature selection method is performed.

Based on the Snowflake flow identified in the previous subsection, the DTLS data transmission message is parsed to extract the encrypted payload length of the message. The constituted length sequence S = (p_1, . . . , p_m). $|p_i|$ indicates the payload length of messages, and the positive or negative value of p indicates the transmission direction of messages, when $p_i > 0$ indicates an output message, and $p_i < 0$ indicates an input message. The accumulative message payload in the data transmission phase can be expressed as A (S) = ((0,0), (i_1, a_1), (i_N,a_N)), where i_N indicates the message location index, $a_1 = p_1$, $a_i = a_{i-1} + p_i$ for i = 2, . . . , N. We obtain m equally spaced sampling points A_1, . . . , A_m by sampling in the segmented linear interpolation function of the A sequence. By this method, a fixed number of identifiable features can be extracted from Snowflake flows of different lengths. Each sampling point contains the accumulative characteristics of the length sequence of all previous messages, which implies the information of message length and transmission direction during the circuit construction process. In addition, the location of the sampling point sequence can also reflect the difference in different circuit construction processes in the message transmission phase to a certain extent. In the next section, we experimentally verify that the optimal combination of parameters (n, m) can be achieved when choosing the first 300 messages during the DTLS data transmission process and selecting 40 sampling points at equal intervals on the corresponding message payload sequence in our experiment. The specific fine-tune process is described in detail in Section 5.3. In the remainder of this paper, we refer to this feature extraction method based on linear interpolation sampling of the accumulative message payload length in the DTLS data transmission phase as ACCUMUL.

5. Experiment Evaluation

In this section, we perform experiments related to Snowflake traffic fingerprinting and the optimal combination of extracted number of messages **n** and sampling point frequency m. We analyze the effectiveness of Snowflake traffic identification and hidden service identification in Tor-Snowflake scenarios by quantitative evaluation methods. In this paper, we use the precision rate, recall rate, F1 score, and the time consumption of traffic identification as evaluation criteria.

5.1. Data Collection

When a client is configured to access the website with a specific obfuscator PT Snowflake, we capture the exchanged network traffic through the deployed automated network traffic collection environment to form the Snowflake traffic dataset. We used an experiment host with Ubuntu 22.04 and Tor 0.4.6.10 installed, which performed Snowflake general access activities and Snowflake hidden service access activities in a real network environment. We ran shell scripts and python scripts to control the state of the Tor service

and automatically access web pages to generate traffic. We utilized the python scapy library to automate traffic collection. For each visit, we start capturing traffic before opening a new Tor-Snowflake process to access the web page, when we know from the status code that a web page has been fully accessed and successfully responded, we close the Tor and Snowflake processes after a delay of a few seconds (usually less than 5 s). Finally, we terminate the traffic capture process. Therefore, in our experiment, when a client visits a website, new connections and circuits will always be established. The traffic we capture contains the full Tor or Tor hidden service circuit establishment process for each access. In order to ensure that Snowflake successfully establishes a connection before web access and to improve the availability of the collected traffic, we use the Linux kernel function inotify [34] to monitor the changes in the Tor log file notices.log, capture the signal that Snowflake PT successfully establishes a connection and use it as a flag to start web access.

Some previous work [35] has argued that traffic generated by automated scripts is not as close to the real network environment as that generated by manual access. We take this into consideration. In a real network environment, people visiting web pages will display different operation latency and random behavior patterns. These are the weaknesses of automated scripts. We overcome these drawbacks by adding latency in writing scripts and by configuring the relevant parameters to mimic human behavior. Since training a credible and available classifier requires multiple visits to the web to generate large amounts of traffic, an automated script is a reliable choice.

The set of addresses accessed in the experiment consists of hidden service domains and normal website domains. For the hidden service domains set, we randomly select 10,000 addresses from the websites previously obtained by our lab through the Tor hidden service address search engine. For the normal domains set, we extract the Alexa Top 10,000 websites to form the address set. Combining the impact of the prevailing network environment and the usage status of some websites, the effective traffic data generated by the experimentally accessed addresses are shown in Table 2.

Table 2. Tor-Snowflake dataset structure.

Measures	Flows
Tor-Snowflake Ordinary	5271
Tor-Snowflake HS	6150

In order to compare and determine the effectiveness of the method for identifying Snowflake traffic based on DTLS handshake fingerprinting features, we combined DTLS handshake traffic from other WebRTC applications collected during the study of Snowflake indistinguishable by K. MacMillan et al. [36] with randomly selected flows from the Snowflake traffic we collected, formed a new WebRTC DTLS handshake fingerprint dataset. The effective traffic flow of the dataset is shown in Table 3.

Table 3. WebRTC fingerprint dataset structure.

	Snowflake	Facebook Messenger	Google Hangouts	Discord
Flows	1855	1580	1995	1989

5.2. Fingerprint Model Identification Effect and Feature Importance Comparison

Based on the WebRTC DTLS handshake fingerprint dataset collected and composed in the previous subsection, we conducted corresponding experiments on Snowflake-based Tor traffic identification. We carried out comparison experiments on different machine learning algorithms for extracting handshake fingerprint features to identify Snowflake traffic, and discovered that the fingerprint features extracted by this method perform well and have similar results for five machine learning algorithms, namely Random Forest, XGBoost, AdaBoost, LinearSVM, and KNN. The specific experimental results are shown in Figure 5.

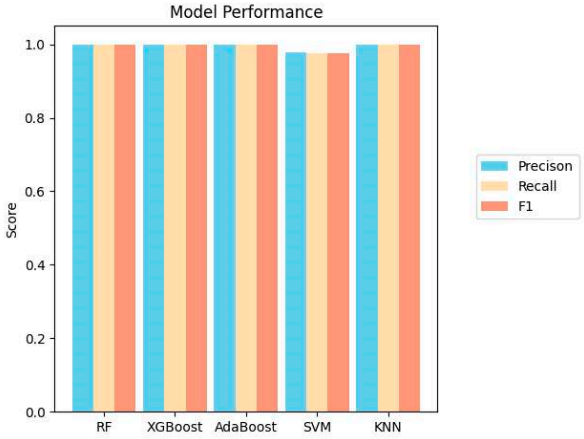

Figure 5. Tor-Snowflake traffic identification effect based on fingerprint features.

The experimental results demonstrate that the fingerprinting method for Tor-Snowflake traffic is effective and the average accuracy of different algorithms achieved more than 99.8%. This method can be applied to a wide range of machine learning models, and the identification requires processing only a small number of DTLS handshake messages. The experiment on the dataset shows that the method consumes less time for extracting fingerprint features and model training for classification, and has better real-time identification of Tor-Snowflake traffic.

The importance ranking of the fingerprint features in the experiment according to Random Forest is shown in Figure 6. The chosen cipher suite in Server Hello has the highest feature importance of 18%. The fragment length of Server Hello, the cipher suites, and the cipher suites length in Client Hello also have a great impact on the Snowflake traffic identification model. The first four features combined can achieve about 50% importance. From the server side of the Snowflake communication, the cipher suite chosen by the volunteers outside the censored area during the handshake negotiation largely reveals the application. The fragment length on the server side changes with the application form the experiment results. The cipher suites content and length of the Client Hello message often differ depending on client application. These fingerprint features can effectively guide the identification of Tor-Snowflake traffic.

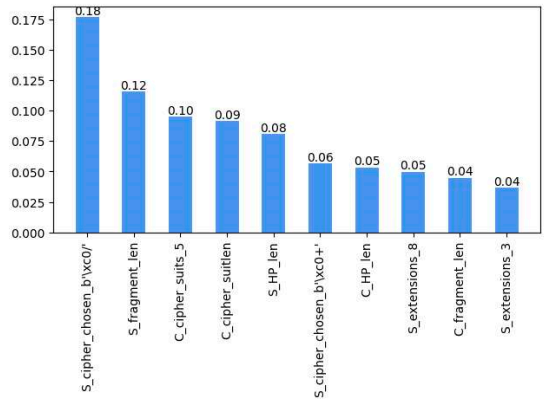

Figure 6. Importance ranking of the fingerprint features.

5.3. Optimal Combination of Parameters

From Section 4.3, the key role in the proposed Tor HS identification method for the Snowflake scenario is the selection of the first n messages in the data transmission phase and the number of selected messages with accumulative payload length sampling frequency m. The selection of the appropriate number of messages n can improve the identification efficiency of the whole framework and maintain a good identification accuracy. In addition,

for the selection of the m value, as m increases, the interval distance between sampling points becomes smaller, and the loss caused by sampling will be less. On the other hand, a large number of features will have a negative impact on the calculation time consumption, which will reduce the learning efficiency and increase the risk of overfitting. In order to obtain the best combination of parameters, we set the interval of the massage number n from 50 to 350 and take 50 as the step size. The number of sampling points m range from 20 to 200 and are taken in steps of 20. We construct the model using the Random Forest algorithm for testing. The results of the experiments are shown in Table 4.

Table 4. Experimental results for the optimal combination of parameters (n, m).

	n = 50	**n = 100**	**n = 200**	**n = 300**	**n = 350**
m = 20	0.5425/1.27	0.5826/1.22	0.8791/1.03	0.9807/0.84	0.9806/0.79
m = 40	0.5301/1.56	0.5721/1.46	0.8844/1.27	0.9812/1.04	0.9796/0.98
m = 60	NULL	0.5772/1.70	0.8771/1.43	0.9813/1.23	0.9795/1.14
m = 80	NULL	0.5913/1.92	0.8763/1.62	0.9796/1.42	0.9801/1.30
m = 100	NULL	0.5927/2.12	0.8750/1.81	0.9801/1.58	0.9813/1.45
m = 120	NULL	NULL	0.8783/2.01	0.9796/1.76	0.9796/1.59
m = 140	NULL	NULL	0.8761/2.20	0.9795/1.92	0.9796/1.76
m = 160	NULL	NULL	0.8778/2.37	0.9790/2.09	0.9807/1.92
m = 180	NULL	NULL	0.8764/2.56	0.9796/2.26	0.9807/2.09
m = 200	NULL	NULL	0.8757/2.55	0.9806/2.30	0.9801/2.09

The higher the precision rate, the smaller the number of n and m, and the less training time is consumed, for the optimal combination of parameters we need. The balance between the recognition effect of the model and the time consumption of feature extraction is achieved. Therefore, the optimal number of messages in the data transmission phase is the first 300 messages, and 40 sampling points are extracted as additional features by the ACCUMUL method for the accumulative payload length in the data transmission phase. From the experiment results, the optimal parameter combinations for (n, m) are (300, 40).

We believe that this method is more beneficial to reflecting the difference between the construction of Tor circuits and Tor HS circuits, because it is implemented for the process of circuit establishment by extracting the payload length of the transmitted data. There is no decryption of the data, and these features are easily extracted. To summarize the above, we believe that the method is portable and expandable, and can be applied to the future problem of difficulty in identifying HS traffic brought about by the new obfuscated PT.

5.4. Comparison of Different Machine Methods on Tor-Snowflake HS Identification

We use the same five machine learning algorithms to partition the training set from our own Tor-Snowflake dataset to train models and check the validity using a ten-fold cross-validation method. We select the first 300 messages of the DTLS data transmission and construct the dataset by sampling 40 sample points for the accumulative message payload length. We construct the experiment results in the form of a confusion matrix for calculating the accuracy rate, precision rate, recall rate, and F1-score. The recognition performance of the five models is shown in Table 5.

Table 5. Recognition performance of five models.

Classifier	**Accuracy**	**Precision**	**Recall**	**F1**	**Time**
Random Forest	0.9917	0.9860	0.9825	0.9871	1.0498
K-Neighbors	0.9903	0.9856	0.9831	0.9867	0.0126
XGBoost	0.9897	0.9848	0.9820	0.9859	0.3623
AdaBoost	0.9858	0.9824	0.9815	0.9836	0.9731
SVM-Linear K	0.9763	0.9551	0.9406	0.9559	0.2953

As seen from the table, our proposed method ACCUMUL achieves an accuracy of over 99% in identifying HS traffic under the Tor-Snowflake scenario when building models using RF and KNN. The result indicates that models constructed by extracting features through ACCUMUL method can effectively identify Tor-Snowflake HS access activities in the network. Among them, the Random Forest algorithm shows better results in most evaluation metrics. The KNN algorithm is close to the results of RF, but its model training time consumption is less. When using the ACCUMUL method to extract features to identify HS access activities, the KNN algorithm would be a better choice, which can improve the overall identification speed of the Tor-Snowflake traffic. The XGBoost and AdaBoost algorithms also have a good performance in the results, but the linear classifier SVM has a relatively poor performance. To summarize the above, the nonlinear classifier has a good performance in identifying Tor-Snowflake HS access activities, and it is more appropriate to choose KNN for model training of the framework for HS traffic identification.

6. Conclusions

Many illegal users and services hide their location and communications through Tor and Tor Hidden Service, so-called Dark-Web. Tor introduced several PT to improve users' anonymity and help them across the censorship. Multiple traffic classification methods have been developed to de-anonymize Tor and PT. Snowflake is the latest release of Tor PT based on the WebRTC, a multimedia peer-to-peer communication technology. It applies UDP as the transport layer framework protocol, which defeat previous traffic classification methods. In this paper, we propose a Tor-Snowflake traffic identification framework F-ACCUMUL, which can identify if a user is connecting Tor through Snowflake PT and whether he is accessing a hidden service. Specifically, we use a rule-matching method to pre-identify traffic quickly. Then, we extract some fingerprint features of the Snowflake DTLS handshake to identify the snowflake traffic accurately. On the basis of this, the first 300 messages of the Snowflake DTLS data transmission phase are extracted and the accumulative payload length are sampled (ACCUMUL). Finally, the Tor-Snowflake HS traffic is identified. The results show that our proposed Tor-Snowflake traffic identification framework can identify Snowflake traffic quickly and accurately. In addition, the feature extraction ACCUMUL method can efficiently identify Tor-Snowflake HS access activities by extracting additional features in combination with statistical features related to traffic identification when choosing the parameter combination of (300, 40).

There are still some deficiencies in our research. The F-ACCUMUL Snowflake traffic identification framework is able to identify Snowflake traffic in the network quickly and accurately. In addition, it can effectively identify the activity of accessing HS in Tor-Snowflake scenarios. However, our proposed identification method based on Snowflake handshake fingerprinting can be resisted by PT developers by eliminating fingerprint information specific to the Snowflake DTLS handshake phase or morphing. This may lead to a long-term offensive and defensive stalemate. In addition, our proposed framework uses machine learning algorithms and does not involve deep learning algorithms. Therefore, in the future, we need to propose a more robust approach for Tor-Snowflake traffic identification that can adapt to the changes of Snowflake version updates. We will also test the effectiveness of our proposed ACCUMUL method to identify HS access activities on other Tor PTs.

Author Contributions: Conceptualization, J.C. and G.C.; methodology, J.C. and H.M.; software, J.C.; validation, J.C. and H.M.; formal analysis, J.C.; investigation, J.C.; resources, J.C.; data curation, J.C.; writing—original draft preparation, J.C.; writing—review and editing, J.C. and H.M.; visualization, J.C.; supervision, G.C.; project administration, G.C.; funding acquisition, G.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Joint Key Program of the National Natural Science Foundation of China under grant number U22B2025 and the General Program of the National Natural Science Foundation of China under grant number 62172093.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Dingledine, R.; Mathewson, N.; Syverson, P.F. Tor: The Second-Generation Onion Router. In Proceedings of the 13th USENIX Security Symposium, San Diego, CA, USA, 9–13 August 2004; Blaze, M., Ed.; USENIX: San Diego, CA, USA, 2004; pp. 303–320.
2. Tor Metrics. Available online: https://metrics.torproject.org/ (accessed on 25 November 2022).
3. Rawat, R.; Rajawat, A.S.; Mahor, V.; Shaw, R.N.; Ghosh, A. Dark web—Onion hidden service discovery and crawling for profiling morphing, unstructured crime and vulnerabilities prediction. In *Innovations in Electrical and Electronic Engineering*; Springer: Berlin, Germany, 2021; pp. 717–734.
4. Karunanayake, I.; Ahmed, N.; Malaney, R.; Islam, R.; Jha, S.K. De-anonymisation attacks on Tor: A Survey. *IEEE Commun. Surv. Tutor.* **2021**, *23*, 2324–2350. [CrossRef]
5. Obfs4. Available online: https://support.torproject.org/glossary/obfs4/ (accessed on 25 November 2022).
6. Meek. Available online: https://support.torproject.org/glossary/meek/ (accessed on 25 November 2022).
7. Guan, Z.; Gou, G.; Guan, Y.; Wang, B. An Empirical Analysis of Plugin-Based Tor Traffic over SSH Tunnel. In Proceedings of the MILCOM 2019–2019 IEEE Military Communications Conference (MILCOM), Norfolk, VA, USA, 12–14 November 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 616–621.
8. He, Y.; Hu, L.; Gao, R. Detection of tor traffic hiding under obfs4 protocol based on two-level filtering. In Proceedings of the 2019 2nd International Conference on Data Intelligence and Security (ICDIS), South Padre Island, TX, USA, 28–30 June 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 195–200.
9. He, Y.; Li, X.; Chen, M.; Wang, W. Identification of tor anonymous communication with cloud traffic obfuscation. *Adv. Eng. Sci.* **2017**, *49*, 121–132.
10. Wang, X.; Chen, Z.; Li, Z.; Huang, W.; Wang, M.; Pan, S.; Shi, J. Identification of MEEK-Based TOR Hidden Service Access Using the Key Packet Sequence. In Proceedings of the International Conference on Computational Science, London, UK, 21–23 June 2022; Springer: Berlin, Germany, 2022; pp. 554–568.
11. Carlucci, G.; De Cicco, L.; Holmer, S.; Mascolo, S. Analysis and design of the google congestion control for web real-time communication (WebRTC). In Proceedings of the 7th International Conference on Multimedia Systems, Klagenfurt, Austria, 10–13 May 2016; pp. 1–12.
12. Barradas, D.; Santos, N.; Rodrigues, L.; Nunes, V. Poking a hole in the wall: Efficient censorship-resistant Internet communications by parasitizing on WebRTC. In Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, 9–13 November 2020; pp. 35–48.
13. Figueira, G.; Barradas, D.; Santos, N. Stegozoa: Enhancing WebRTC Covert Channels with Video Steganography for Internet Censorship Circumvention. In Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security, Nagasak, Tokyo, 30 May–2 June 2022; pp. 1154–1167.
14. Lashkari, A.H.; Draper-Gil, G.; Mamun, M.S.I.; Ghorbani, A.A. Characterization of tor traffic using time based features. In Proceedings of the ICISSP, Porto, Portugal, 19–21 February 2017; pp. 253–262.
15. Montieri, A.; Ciuonzo, D.; Bovenzi, G.; Persico, V.; Pescapé, A. A dive into the dark web: Hierarchical traffic classification of anonymity tools. *IEEE Trans. Netw. Sci. Eng.* **2019**, *7*, 1043–1054. [CrossRef]
16. Shahbar, K.; Zincir-Heywood, A.N. Anon17: Network traffic dataset of anonymity services. *Fac. Comput. Sci. Dalhous. Univ. Tech. Rep.* **2017**.
17. Montieri, A.; Ciuonzo, D.; Aceto, G.; Pescape, A. Anonymity services tor, i2p, jondonym: Classifying in the dark (web). *IEEE Trans. Dependable Secur. Comput.* **2018**, *17*, 662–675. [CrossRef]
18. Panchenko, A.; Lanze, F.; Pennekamp, J.; Engel, T.; Zinnen, A.; Henze, M.; Wehrle, K. Website Fingerprinting at Internet Scale. In Proceedings of the NDSS, San Diego, CA, USA, 21–24 February 2016.
19. Rimmer, V.; Preuveneers, D.; Juarez, M.; Van Goethem, T.; Joosen, W. Automated website fingerprinting through deep learning. In Proceedings of the Network and Distributed Systems Security (NDSS) Symposium, San Diego, CA, USA, 18–21 February 2018.
20. Sirinam, P.; Imani, M.; Juarez, M.; Wright, M. Deep fingerprinting: Undermining website fingerprinting defenses with deep learning. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, Toronto, ON, Canada, 15–19 October 2018; pp. 1928–1943.
21. Juárez, M.; Imani, M.; Perry, M.; Dıaz, C.; Wright, M. WTF-PAD: Toward an efficient website fingerprinting defense for tor. *arXiv* **2015**, arXiv:1512.00524.
22. Wang, T.; Goldberg, I. {Walkie-Talkie}: An Efficient Defense against Passive Website Fingerprinting Attacks. In Proceedings of the 26th USENIX Security Symposium (USENIX Security 17), Vancouver, BC, Canada, 16–18 August 2017; pp. 1375–1390.
23. Wang, L.; Dyer, K.P.; Akella, A.; Ristenpart, T.; Shrimpton, T. Seeing through network-protocol obfuscation. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, 12–16 October 2015; pp. 57–69.

24. Shahbar, K.; Zincir-Heywood, A.N. An analysis of Tor pluggable transports under adversarial conditions. In Proceedings of the 2017 IEEE Symposium Series on Computational Intelligence (SSCI), Honolulu, HI, USA, 27 November–1 December 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–7.
25. Fifield, D.; Epner, M.G. Fingerprintability of WebRTC. *arXiv* **2016**, arXiv:1605.08805.
26. Fifield, D. *Threat Modeling and Circumvention of Internet Censorship*; University of California, Berkeley: Berkeley, CA, USA, 2017.
27. Frolov, S.; Wustrow, E. The use of TLS in Censorship Circumvention. In Proceedings of the NDSS, San Diego, CA, USA, 24–29 February 2019.
28. Wang, X.; Li, Z.; Huang, W.; Wang, M.; Shi, J.; Yang, Y. Towards Comprehensive Analysis of Tor Hidden Service Access Behavior Identification under Obfs4 Scenario. In Proceedings of the 2021 ACM International Conference on Intelligent Computing and its Emerging Applications, Jinan, China, 28–29 December 2021; pp. 205–210.
29. Salman, O.; Elhajj, I.H.; Kayssi, A.; Chehab, A. A review on machine learning–based approaches for Internet traffic classification. *Ann. Telecommun.* **2020**, *75*, 673–710. [CrossRef]
30. Flashproxy. Available online: https://github.com/arlolra/flashproxy (accessed on 25 November 2022).
31. Fifield, D.; Lan, C.; Hynes, R.; Wegmann, P.; Paxson, V. Blocking-resistant communication through domain fronting. *Proc. Priv. Enhancing Technol.* **2015**, *2015*, 46–64. [CrossRef]
32. Petit-Huguenin, M.; Salgueiro, G.; Rosenberg, J.; Wing, D.; Mahy, R.; Matthews, P. Session Traversal Utilities for NAT (STUN). *RFC* **2020**, *8489*, 1–67. [CrossRef]
33. DTLS 1.2 rfc6347. Available online: https://datatracker.ietf.org/doc/html/rfc6347 (accessed on 25 November 2022).
34. Fournier, G.; Afchain, S.; Baubeau, S. Runtime Security Monitoring with eBPF. In Proceedings of the 17th SSTIC Symposium sur la Sécurité des Technologies de l'Information et de la Communication, Rennes, France, 2–4 June 2021.
35. Aceto, G.; Ciuonzo, D.; Montieri, A.; Pescapé, A. Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges. *IEEE Trans. Netw. Serv. Manag.* **2019**, *16*, 445–458. [CrossRef]
36. MacMillan, K.; Holland, J.; Mittal, P. Evaluating snowflake as an indistinguishable censorship circumvention tool. *arXiv* **2020**, arXiv:2008.03254.