


## Article

# A Survey on TLS-Encrypted Malware Network Traffic Analysis Applicable to Security Operations Centers

Chaeyeon Oh <sup>1</sup>, Joonseo Ha <sup>2</sup> and Heejun Roh <sup>1,\*</sup><sup>1</sup> Department of Cyber Security, Graduate School, Korea University, Sejong 30019, Korea; cy0106@korea.ac.kr<sup>2</sup> Cyber Security Major, Division of Applied Mathematical Sciences, Korea University Sejong Campus, Sejong 30019, Korea; cujs1106@korea.ac.kr

\* Correspondence: hjroh@korea.ac.kr; Tel.: +82-44-860-1312

**Abstract:** Recently, a majority of security operations centers (SOCs) have been facing a critical issue of increased adoption of transport layer security (TLS) encryption on the Internet, in network traffic analysis (NTA). To this end, in this survey article, we present existing research on NTA and related areas, primarily focusing on TLS-encrypted traffic to detect and classify malicious traffic with deployment scenarios for SOCs. Security experts in SOCs and researchers in academia can obtain useful information from our survey, as the main focus of our survey is NTA methods applicable to malware detection and family classification. Especially, we have discussed pros and cons of three main deployment models for encrypted NTA: TLS interception, inspection using cryptographic functions, and passive inspection without decryption. In addition, we have discussed the state-of-the-art methods in TLS-encrypted NTA for each component of a machine learning pipeline, typically used in the state-of-the-art methods.

**Keywords:** network traffic analysis; traffic classification; security operations center; transport layer security; malware



check for updates

**Citation:** Oh, C.; Ha, J.; Roh, H. A Survey on TLS-Encrypted Malware Network Traffic Analysis Applicable to Security Operations Centers. *Appl. Sci.* **2022**, *12*, 155. <https://doi.org/10.3390/app12010155>

Academic Editors: Carsten R. Maple and Arcangelo Castiglione

Received: 13 November 2021

Accepted: 21 December 2021

Published: 24 December 2021

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Since the last two decades, security operations centers (SOCs) can be found in multiple organizations (for example, enterprises, government, and universities), which are frequent targets of cybersecurity attacks by adversaries. As the focal point for various security operations and computer network-based defense, an SOC is typically a group comprising security experts, which conducts various security operations including detection, analysis, response, reporting, and prevention of cybersecurity incidents [1]. Despite their practical importance to organizations, especially in the last few years, there is only a fragmented and widespread literature focusing on various issues in SOCs [2,3].

Amongst a large set of services provided by SOCs, network intrusion monitoring, detection, and analysis (or network security monitoring [4,5]) are highly relevant to NTA. From an academic perspective, NTA is a branch that constitutes inferential methods to obtain traffic-related information of end hosts, users, application processes, and protocols from network traces (e.g., captured packets (in pcap format [6]), flow records (in NetFlow or IPFIX format [7,8]), and more). With the rapid growth of the Internet, various approaches in NTA have been investigated in different contexts, such as traffic engineering, network security, accounting, and advertising. Security experts in SOCs can achieve their goals, such as intrusion (or malware traffic) detection and traffic classification, using these methods. According to a survey conducted by ENEA Qosmos Division [9], 87% of security expert participants are familiar with NTA, and majority of their organizations already use NTA.

Recently, however, the majority of SOCs have been facing a critical issue with NTA: increasing adoption of traffic encryption on the Internet [10,11]. While the secure sockets layer (SSL), the predecessor to transport layer security (TLS), appeared in the middle of the 1990s to provide end-to-end communication privacy over the Internet, only 44.3% of web connections of European residential customers used HTTPS (a secure version

of the HyperText Transfer Protocol using SSL/TLS) in September 2014 [12]. However, in October 2021, Google [13] and Let's Encrypt [14] reported that more than 80% of the web pages loaded in Chrome and Firefox browsers (which allows to sharing of usage statistics) used HTTPS, primarily owing to the push by major web browsers and community efforts such as Let's Encrypt [15] and HTTPS Everywhere [16]. Through these trends, existing NTA methods, relying primarily on application layer payload processing (for example, rule-based or signature-based intrusion detection, and deep packet inspection), lose their utility for encrypted traffic [17].

In this survey article, we present existing research on NTA and related areas primarily focusing on TLS-encrypted malware traffic, which can be utilized by security experts in SOCs. While there are multiple related surveys [18–27] available on NTA and traffic classification areas, our approach has the following distinguished contributions:

- TLS is a widely used end-to-end encryption protocol with a wide variety of applications in diverse configurations [28]. Additionally, various malware families (especially Trickbot and Dridex) abuse TLS encryption [29,30], which is one of the biggest challenges faced by SOCs in recent years. Furthermore, the fraction of TLS-encryption that flows among malware flows is dramatically increasing: there were industrial reports in April 2021 stating that nearly a half of the malware uses TLS [31], and further in the second quarter of 2021, it stated that 91.5% of malware arrives over TLS-encrypted traffic [32]. As we have primarily discussed NTA methods applicable to malware detection and family classification, security experts in SOCs and researchers in academia can obtain useful information from our survey.
- While various surveys only focus on comparison between existing methods, we also cover industrial and community efforts on the so-called TLS fingerprinting techniques. Similar to multiple data-driven approaches, the performance of NTA is directly related to the quality of the dataset. Fortunately, several open source threat intelligence (OSINT) feeds [33,34] now provide TLS fingerprint information. Therefore, through our discussion, better traffic analysis results can be achieved by integrating such information.

Table 1 shows a comparison of this survey with other related surveys concerning encrypted NTA in terms of protocols, problem domains, and methods. Papadogiannaki et al. [18] has a comprehensive and up-to-date survey on encrypted NTA methods and their countermeasures, but it is considerably less detailed especially in the security domain. Pacheco et al. [19] surveyed machine learning-based (encrypted and unencrypted) traffic classification methods. However, it does not include the methods for encrypted malware traffic in recent years. To the best of our knowledge, Velan et al. [20] presented the first survey in encrypted NTA in 2015; however, considerable significant research has been executed in this area since. Aceto et al. [21] evaluated several existing deep learning-based methods using experiments focused on mobile applications. Conti et al. [22] provided a comprehensive survey on NTA methods for mobile devices where a majority of the methods capture the mobile traffic at mobile devices or at Wi-Fi access points. While some enterprise SOCs with wireless networks can utilize the methods discussed in [22], mobile device-specific and wireless link-specific features are not available in many SOCs which protect servers in wired networks. In this context, we focus on the NTA methods without link-specific features. Poh et al. [23] presented a survey on privacy-preserving inspection in middleboxes with only a slight coverage on machine learning techniques. Rezaei et al. [24] briefly overviewed deep learning-based methods for encrypted traffic classification. Additionally, Shen et al. [25] provided a brief overview of machine learning-based encrypted traffic classification; however, the primary topic of this article is feature selection and optimization for a website fingerprinting dataset. These surveys [24,25] address either machine learning-based or deep learning-based methods. While we have included comprehensive approaches with consideration of the security domain. Shbair et al. [26] described research efforts on services identification inside HTTPS, while de Carnavalet et al. [27] extensively introduced industry practices on TLS interception. In this paper, we have discussed the state-of-the-art methods applied in TLS-encrypted NTA, especially applicable to malware

detection and family classification for SOCs. We believe that security experts in SOCs can understand both the industrial and academic trends on TLS-encrypted malware NTA from [27] and this article, respectively.

**Table 1.** A comparison of this paper with other related surveys, where ML refers machine learning and DL refers deep learning.

Survey	Protocols	Problem Domains	Methods	Notes
[18]	Various	Various	Various	<ul style="list-style-type: none"> <li>Comprehensive and up-to-date survey on encrypted NTA methods</li> <li>Insufficient detail for the security domain</li> </ul>
[19]	Various	Various	ML-based only	<ul style="list-style-type: none"> <li>Comprehensive survey on ML-based methods</li> <li>Omits recent works for encrypted malware traffic</li> </ul>
[20]	Various	Various	Various	<ul style="list-style-type: none"> <li>The first in encrypted traffic analysis area</li> <li>Published in 2015 so that lacks the state-of-the-art</li> </ul>
[21]	Various	Mobile Apps	DL-based only	<ul style="list-style-type: none"> <li>Experimental evaluation among existing DL-based methods</li> </ul>
[22]	Various	Mobile Apps	Various	<ul style="list-style-type: none"> <li>The most comprehensive for mobile traffic</li> <li>May not suitable for many SOCs protecting servers</li> </ul>
[23]	Various	Detection	Various	<ul style="list-style-type: none"> <li>Comprehensive survey on privacy preserving inspection in middleboxes</li> <li>Too little coverage on ML-based methods</li> </ul>
[24]	Various	Traffic Classification	DL-based only	<ul style="list-style-type: none"> <li>Brief overview on DL-based methods</li> </ul>
[25]	Various	Website fingerprinting	Feature selection only	<ul style="list-style-type: none"> <li>Brief overview on ML-based methods</li> <li>Focus on website fingerprinting dataset</li> </ul>
[26]	HTTPS	Web Apps	Various	<ul style="list-style-type: none"> <li>Services identification inside HTTPS</li> </ul>
[27]	TLS	Various	Interception-based only	<ul style="list-style-type: none"> <li>Industry practices analysis of TLS interception</li> </ul>
This Paper	TLS	Malware Traffic	Various, focusing on ML-based	<ul style="list-style-type: none"> <li>The state-of-the-art for ML-based malware detection and family classification</li> </ul>

The remainder of this paper is organized as follows. In Section 2, we provide several backgrounds, such as the goals of NTA for SOCs and the basics of SSL/TLS. Section 3 presents the three deployment models of NTA solutions: TLS interception without a private key, passive inspection using cryptographic functions, and inspection without decryption. Among the scenarios, we introduce several approaches in inspection without decryption in detail, considering the recent advances in the area. In Section 4, using an entire pipeline of machine learning-based analysis, we discuss the state-of-the-art methods in each component of the pipeline. Conclusions of this survey and future directions in TLS-encrypted NTA are drawn in Section 5.

## 2. Background

In this section, we introduce the background information.

### 2.1. Basics of SSL/TLS

SSL is the de facto standard Internet protocol used to establish secure end-to-end sessions over transmission control protocol (TCP) for providing communications privacy. According to the final draft of SSL 3.0 [35], SSL was designed to prevent several security attacks, such as eavesdropping, tampering, and message forgery. However, SSL 3.0 was deprecated in June 2015 [36].

TLS is the successor of SSL with backward compatibility with SSL, which was firstly published in RFC 2246 [37] in 1999. The current up-to-date version of TLS is TLS 1.3, defined in RFC 8446 [38]. However, a few of its previous versions (i.e., TLS 1.0 and TLS 1.1) were deprecated in March 2021 [39]. According to a report by Qualys SSL Labs [40] in October 2021, 99.6% of the surveyed websites support TLS 1.2 [41], and approximately half

(49.7%) of the websites support TLS 1.3, while less than a half support TLS 1.0 and TLS 1.1. Therefore, unless explicitly specified, we have primarily focused on TLS 1.2 in this survey.

Following the TCP connection establishment procedure performed between two endpoints, a TLS session is initiated by one endpoint by sending a `ClientHello` message, and such endpoint is referred to as the client of the TLS session. The `ClientHello` message contains the client's TLS version, a list of supported ciphersuites (i.e., cryptographic options) ordered by preference of the client, and a list of requested extensions (i.e., extended functionality from servers), such as server name indication (SNI) extension defined in RFC 6066 [42]. Note that the `HostName` field in the SNI extension includes the fully qualified domain name system (DNS) hostname of the server (i.e., another endpoint), which can primarily be used for hosting multiple virtual servers, also known as virtual hosts, in an end host (*for example*, to enable request direction to an appropriate virtual server without decryption).

Next, the server responds with a `ServerHello` message containing the server's chosen ciphersuite among the client-offered ciphersuites, the server's chosen extensions among the client-requested extensions, a `Certificate` message containing a sequence (chain) of certificates for proving the identity of the server, and finally a `ServerHelloDone` message to indicate the end of the response. The client then verifies the authenticity of the given certificate chain. Once the `ClientKeyExchange` message is sent from the client, and `ChangeCipherSpec` and `Finished` messages are sent from both endpoints, the endpoints can exchange encrypted application data.

## 2.2. The Goals of Network Traffic Analysis for SOCs

While there are various types and sizes of SOCs, NTA is performed to achieve different purposes by security experts in SOCs whose role includes preliminary threat detection, triage of events, and incident response [43]. While in this paper we focus on malware detection and malware family classification only, we have listed three main goals of security experts in SOCs associated with network security monitoring as follows:

- *Malware detection*: In malware (traffic) detection, NTA is used to detect network traffic containing various types of malicious content, or contributing to malicious applications. Traditionally, detection of malicious traffic is analyzed according to pre-configured rules for known attacks, but machine learning-based detection has been proposed as a complement of the signature-based network intrusion detection systems [44]. Malware detection methods typically utilize accumulated attack knowledge so that collecting and regularly updating the knowledge base is important in SOCs [2].
- *(Network) Anomaly Detection*: Network anomaly detection, or anomaly based intrusion detection is the problem to detect exceptional patterns in network traffic which can be distinguished from the expected normal network traffic pattern [45]. A broad range of anomaly detection techniques such as statistical, unsupervised, and rule-based techniques have been proposed in literature [46]. Furthermore, deep learning-based anomaly detection systems are actively discussed [47]. However, in real-world SOCs, the potential of human security experts may be more trusted than the automated methods so that some SOCs utilize or develop practical machine learning-based anomaly detection solutions combined with information visualization [48,49], which is out of our scope.
- *Application identification*: NTAs for application identification identify the network traffic from particular applications, including unauthorized applications. This can be used for specific policy enforcement in SOCs (e.g., block Amazon traffic during work hours). Recently, especially for mobile traffic, there are several machine learning-based solutions where mobile application identification and even user actions can be identified [21,22], which is sometimes called user behavior analytics (UBA) in the context of SOCs [50]. While malware family classification can be seen as a variant of the conventional application identification problem, to the best of our knowledge, there is no NTA method to identify fine-grained behavior of malware from encrypted traffic.

### 3. The Deployment Models

To analyze encrypted traffic, including TLS-encrypted traffic, deployment models are widely used to deploy either, middleboxes for traffic interception, or traffic sensors for passive inspection in practice. We discuss three main deployment models for encrypted NTA: TLS interception, inspection using cryptographic functions, and passive inspection without decryption.

#### 3.1. TLS Interception without Private Key

As described in Section 2.1, TLS was primarily designed to establish an encrypted end-to-end session between the client and the server. However, in TLS interception, the client establishes an end-to-end TLS session with a middlebox (typically a TLS proxy in this context, but it may consist of more inspection-related functionalities, such as router, firewall, intrusion detection system (IDS)/intrusion prevention system (IPS), and content filter). As the middlebox is an endpoint of the TLS session, there is no hurdle to decrypt the application layer data. Hence, TLS interception transforms encrypted NTA into payload-based traffic analysis (also known as deep-packet inspection (DPI)), which is well established in the literature [51]. Thus, for TLS interception, TLS proxies and HTTPS proxies, which can be considered as a combination of TLS proxy and HTTP proxy, are widely deployed, especially in enterprise SOCs [27,52].

Notably, as shown in Figure 1 following the inspection and analysis, the middlebox forwards the data to the server via another end-to-end TLS session (i.e., with re-encryption) between the middlebox (on behalf of the client) and the server, where security policies can be enforced for the TLS traffic. This implies that although TLS is derived from a design motivated by the end-to-end argument [53], TLS interception breaks the end-to-end security property of TLS, which incurs concerns about man-in-the-middle (MITM) attacks on TLS. Furthermore, both the client and server should trust the middlebox, or the middlebox should impersonate the server (e.g., with certificate delegation [54], or forged certificates [55]). As such, TLS interception has drawn attention in various debates [27,56,57].

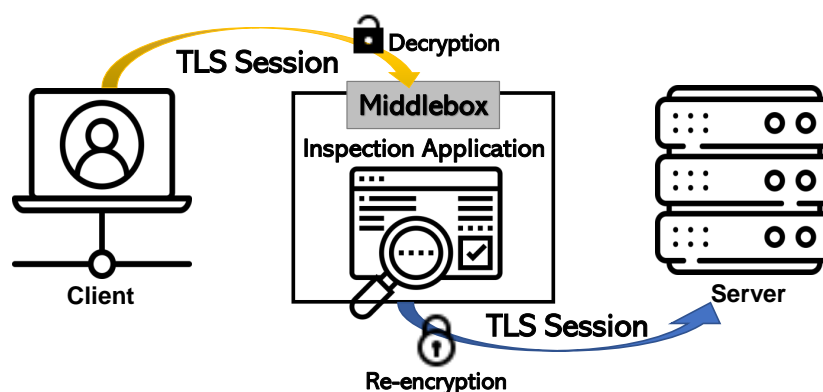


Figure 1. TLS interception.

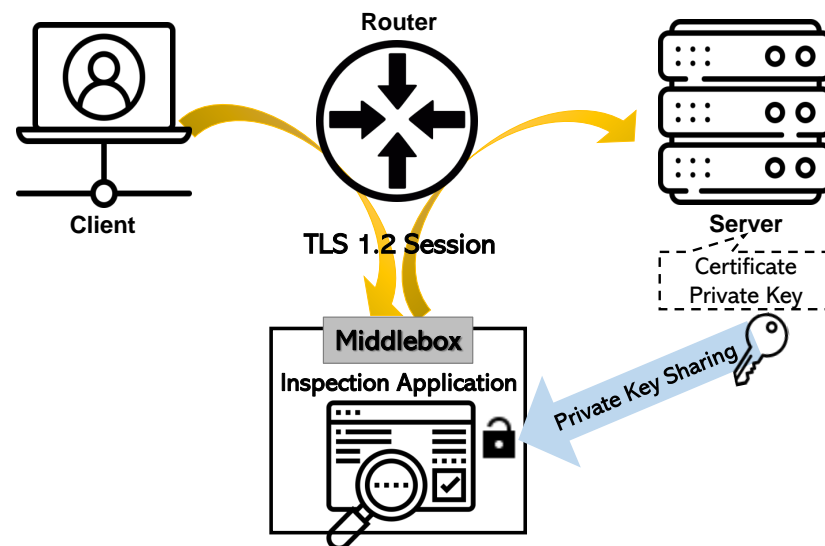
#### 3.2. Inspection Using Cryptographic Functions

The second category of the deployment models is inspection using cryptographic functions. We can further classify it into two sub-categories: TLS inspection with a private session key, and TLS inspection with searchable encryption [58–60].

##### 3.2.1. TLS Inspection with a Private Key

In certain configurations of TLS, such as TLS 1.2 with Rivest–Shamir–Adleman (RSA)-based ciphersuite, when the server shares the certificate private key with the middlebox as shown in Figure 2, TLS-encrypted traffic can be decrypted [27]. Similarly, Wireshark, a well-known network protocol analyzer, has a feature to decrypt TLS-encrypted traffic in the aforementioned configurations. Therefore, out-of-band (passive) TLS inspection is possible using a private key and appropriate configurations. However, this approach cannot

be applied in other ciphersuites in TLS 1.2, such as Diffie–Hellman-based ciphersuites. In addition, TLS 1.3 does not support ciphers without forward secrecy, such as RSA-based ciphersuites. Additionally, de Carnavalet and van Oorschot [27] discussed static Diffie–Hellman key sharing use cases and issues in detail.



**Figure 2.** TLS inspection with a private key.

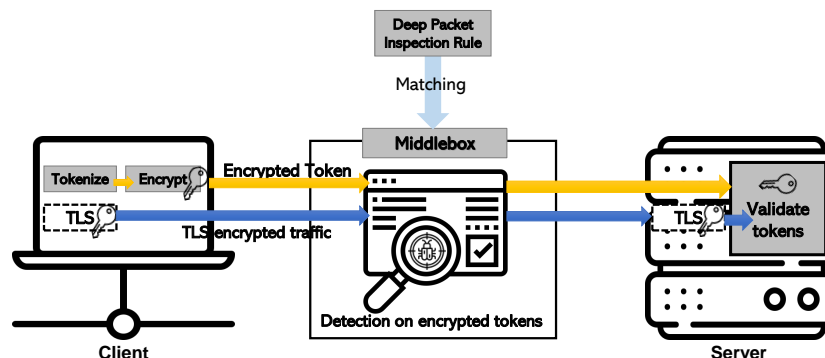
### 3.2.2. Privacy-Preserving Inspection through Searchable Encryption

While the previous deployment models have been popularized by the industry, the models are based on trusting middleboxes. However, the middleboxes with vulnerabilities can also be used by adversaries to compromise the privacy of the client and server [61]. According to Waked et al. [62], while the levels vary, all tested enterprise-grade TLS interception middleboxes are vulnerable.

Several studies have been conducted to enable privacy-preserving inspection with the help of searchable encryption techniques (e.g., [58]). For example, BlindBox [63] as shown in Figure 3 is a pioneering work on privacy-preserving deep packet inspection, based on a searchable encryption technique. While two privacy models are implemented, the common idea is for the client to transmit encrypted tokens generated from plaintext of a (unidirectional for simplicity) TLS session, to the middlebox through an out-of-band channel. Next, the middlebox attempts to perform deep packet inspection rule matching for the encrypted tokens, which is enabled by the searchable encryption technique. In addition, as the encrypted tokens could be different from the TLS-encrypted traffic, the receiver (which has the valid decryption key for the TLS session) cooperates with the middlebox (which should not have the key) by checking whether the receiver-decrypted tokens (forwarded from the middlebox) and the recovered plaintext from the TLS session are the same. The authors of BlindBox extend their system to support a wide range of middlebox services such as firewall, network address translation (NAT), HTTP proxy, and deep packet inspection [64]. Yuan et al. [65] proposed an architecture to perform private preserving deep packet inspection with a novel rule filter for achieving better performance than BlindBox. Ning et al. [66] utilize a reusable obfuscation mechanism for faster encrypted rule generation.

Although the idea of privacy-preserving inspection through searchable encryption is interesting and innovative, unfortunately, such approaches are less promising in the current generation of SOCs. At first, as compared with other deployment solutions, BlindBox and the following studies (e.g., [65,66]) exhibit poor performance. For instance, in BlindBox, given an IDS with typically 3000 rules, the required client-side time is 97 s. Furthermore, BlindBox requires an out-of-band channel with its own protocol in conjunction with TLS, which is an implausible assumption for malware; malware may use other encrypted channels to hide its malicious network behavior. Note that similar weakness can be

found in recent inspection methods using cryptographic functions such as BlindIDS [67], IA2-TLS [68], P2DPI [69], etc., and other methods [70,71] relying on trusted execution environments, such as Intel SGX [72] in the middlebox, they are beyond the scope of this survey.



**Figure 3.** BlindBox system architecture as a representative example of inspection through searchable encryption.

### 3.3. Inspection without Decryption

The main motivation of this approach is that TLS-encrypted traffic itself exposes unencrypted metadata, and is equipped with other measurable properties (for example, packet length sequence and inter-arrival time sequence of a flow) that can be used to infer certain information related to the encrypted content.

While Papadogiannaki and Ioannidis [73] propose that the packet length sequence can be used in exact signature matching for encrypted traffic, and a few exact pattern matching-based NTA methods can be found especially in TLS fingerprinting (Section 4.4), a majority of the solutions in this category adopt graphical, statistical, or machine learning algorithms.

A representative example in graphical methods is graphlet in BLINC [74]. Graphlet is a transport layer interaction pattern between hosts represented by a graph with the intent to identify the network application. In BLINC, heuristics are used for application classification with graphlet. Statistical methods have been discussed in the context of application protocol classification. For example, Velan et al. [75] compared flow-based, packet-based, and byte-based statistics and observes that flow-based statistics are more stable than others. While statistical methods have established a wide range of literature, the majority of the recent works discussed in malware detection and family classification employ machine learning techniques. For the machine learning-based methods, we have discussed their solutions in Section 4.

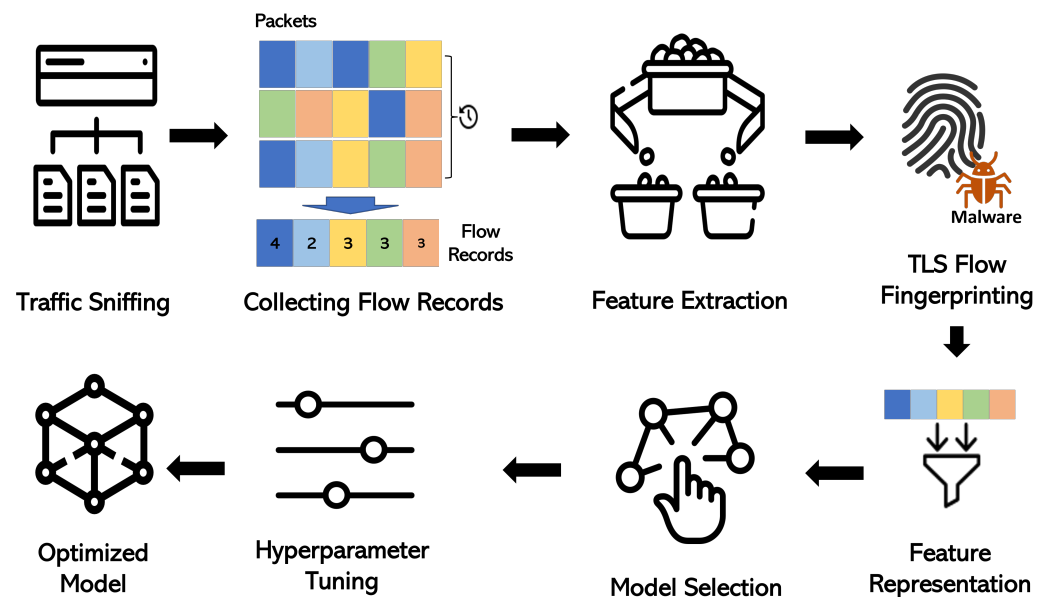
To avoid potential network performance degradation due to the on-the-path inspection using complex machine learning-based algorithms, various studies in this category implicitly assume that encrypted traffic is inspected off-the-path (for example, by traffic sniffing with switch port mirroring [76] or network taps, or flow record collection through NetFlow sensors [8,77]).

However, we should note that in general, the lightweight inspection without decryption can be deployed on-the-path, as shown in Figure 1. For instance, there are near real-time protocol identification solutions without decryption, such as iPoque's (acquired by Rohde and Schwarz) protocol and application classification engine (PACE) [78] and nDPI [79]. Similarly, there are several studies that identify application layer protocol with the first few packets only [80,81], even with encrypted traffic [82]. Nevertheless, such solutions primarily focus on protocol identification in the middlebox to forward network traffic to a protocol-specific proxy (e.g., TLS proxy in Section 3.1) for interception.

## 4. Machine Learning Pipeline for Passive Inspection of TLS-Encrypted Traffic

In passive inspection and analysis of TLS-encrypted traffic, it is more effective to describe a machine learning pipeline in advance, typically used in the state-of-the-art

models. Figure 4 shows a visual representation of our summary for such a pipeline. Based on this pipeline, in this section, we have discussed the state-of-the-art methods in TLS-encrypted NTA for each component of the pipeline.



**Figure 4.** Machine learning pipeline for passive inspection of TLS-encrypted traffic.

#### 4.1. Traffic Sniffing

Packet sniffers such as tcpdump can be used to collect TLS-encrypted traffic. Given features used in the machine learning pipeline, using packet sniffers with appropriate packet filters is desirable to reduce unnecessary packets dramatically, which further improves the performance of traffic analysis (e.g., throughput). For example, assuming a TCP segment contains a TLS message in its payload when the first byte of the TCP payload is 22, while the sixth byte is 1, the TLS message is a ClientHello message. Using this information, a security expert can manually extract the HostName field in the SNI extension contained in the ClientHello message.

#### 4.2. Collecting Flow Records

Once the TLS-encrypted traffic is sniffed, collection of flow records should be conducted, as various machine-learning based methods assume that the raw input is a unidirectional/bidirectional TLS flow. A flow record may have various raw data for the corresponding flow and packets.

In practice, conventional flow records can be collected from the middleboxes (e.g., routers, switches), or software-based traffic sensors (e.g., nProbe), which can export NetFlow/IPFIX. While conventional NetFlow records have miniscule information for TLS, McGrew, and Anderson [83] proposed enhanced TLS flow records, which contain the sequence of packet lengths and (interarrival) times (SPLT), the byte distribution (BD) in the TLS flow data (an array keeping a count for each byte value in the packet payloads for the TLS flow), and TLS handshake metadata (features that can be collected in the TLS handshaking procedure described in Section 2.1). The enhanced flow records can be collected using Cisco joy [84], which is an open-source prototype of Cisco's encrypted traffic analytics (ETA) [11] and a precursor of Cisco mercury, while the current version of Cisco mercury does not support to collect some fields in the enhanced flow records.

Note, that while multiple TLS-encrypted NTA techniques utilize a subset of the enhanced TLS flow records in [83], there are several approaches to utilize high-level connection logs as flow records in bot detection [85,86] and malware family classification [87]. These approaches can be effective solutions for SOCs, because such information can be readily collected using conventional network security monitoring systems, such as Zeek (formerly known as Bro [88]).



However, it is unclear whether these researches [85–87] are applicable in TLS-encrypted traffic, as there has been no performance evaluation for TLS-encrypted traffic dataset.

#### 4.3. Feature Extraction

While a flow record consists of detailed, but important information (i.e., features) on the corresponding flow, there can be less significant features compared to the other models. In some cases, it is better to represent, or summarize some features into a transformed feature to achieve certain goals (for example, interpretability of machine learning, and compact fingerprinting). Different naming, or division into several steps (e.g., feature extraction and feature selection) can be identified in the literature, but we refer to this procedure as feature extraction. Shen et al. [25] provides an appealing tutorial on feature extraction for encrypted traffic classification.

A flow record consists of raw data for the corresponding flow and packets. Such raw data may include the following types of features:

- Variable-size sequential data type: TLS message type sequence, packet length sequence, interarrival time sequence, and time-slotted Zeek connection state log [86] have variable sizes, which is not suitable as an input for certain machine learning algorithms. There are several studies to transform variable-size data into statistical representative values (e.g., max, min, median, standard deviation, etc.) or a specific probabilistic/statistical object, such as a histogram and its self-similarity matrix [89], a first-order Markov chain [90], a second-order Markov chain [91], a hidden-Markov model [92], each of which can be represented as a finite-dimensional vector, while only statistical information remains in such models. Among these, Markov chain transformation has been widely used in TLS-encrypted traffic classification. Note that the approach in [89] has only been validated for unencrypted traffic; hence, we consider the adoption of the proposed approach into encrypted traffic under prospects for future work. In contrast, there are several approaches to utilize machine learning algorithms, which allows variable-size input. FS-Net [93] proposes an end-to-end traffic classification model as a variant of the recurrent neural network (RNN), which allows the packet length sequence of a flow record to be an input. According to [93], FS-Net outperforms several Markov-chain based approaches in the true positive rate and the false positive rate. Shen et al. [94] proposed a novel graph-based representation of packet length sequences (with the direction of each packet between the client and the server), known as *traffic interaction graph* (TIG). This research also proposes a graph neural network, which can classify decentralized applications on Ethereum from TLS encrypted traffic.
- Categorical data type: Each element of TLS client-offered ciphersuite list and TLS client-advertised extension list has a unique value with a finite number of cases, namely  $n$ , to allow better representation of a  $n$  bit vector using one-hot encoding, although the order information of the list would be lost. For example, Anderson and McGrew [17] observed that there are only 176 cases for each element in TLS client-offered ciphersuite list in their dataset. They also reported that applying order-preserving representation on the list did not increase the performance significantly.
- Numeric data type: There are several numeric data type fields in TCP header and TLS message header of each packet, and it is not necessary for such data to be transformed into other data types.
- String data type: the `HostName` field in the SNI extension, the `Certificate` message in TLS handshaking, the `subjectAltName` field in the `Certificate` message and TLS flow data can be considered as string data. As each character has a unique value and a string has variable length, these data can be considered as variable-size sequential data types. In this context, the byte distribution in [83] can be observed as a histogram of TLS flow data. However, in various approaches such as [91], only the string length is extracted as a feature. In addition, ref. [17] reports that the mismatch between the `subjectAltName` field and the `HostName` field, if available, can be an effective feature for malware detection.

#### 4.4. TLS Flow Fingerprinting

In practice, TLS fingerprint is an indicator of compromise (IoC) [95], which summarizes *one or more* TLS flow records with the same label, where the label has a dependency on its problem domain (e.g., malware/benign in malware detection [96], a specific malware family in malware family classification [29], mobile app in mobile traffic classification [21], and user agent string in browser fingerprinting [97,98]). Therefore, a TLS fingerprint can be used as a clarified input for machine learning algorithms in training/testing, as well as a model representation for a class (i.e., a specific label).

A widely used and active TLS fingerprinting method is JA3 [99]. This MD5 hash-based TLS client fingerprinting technique was proposed by John B. Althouse, Jeff Atkinson, and Josh Atkins of Salesforce in 2017 and named after its three authors with the same initials. A JA3 fingerprint summarizes SSL version, offering ciphersuite list, TLS extension list, and elliptic curve-related information in TLS `ClientHello` message of a TLS session with the MD5 hash function, while ignoring Google's GREASE (Generate Random Extensions And Sustain Extensibility) [100].

As an example, when we have a `ClientHello` message where the version field has value `0x0303 = 771` (i.e., TLS 1.2), the list of supported ciphersuites contains the following hyphen-separated values `4-5-10-9-100-98-3-6-19-18-99`, and there is no TLS extension list and elliptic curve-related information, then the comma-concatenated string `771,4-5-10-9-100-98-3-6-19-18-99,,` is the input of the MD5 hash function. The resulting JA3 fingerprint is `07571e689c94dd5474350e204a2f3ade`. Note that for better visibility of malicious client-server combination (e.g., Tor client and Tor server), JA3S, a server-side version of JA3, was also proposed.

Currently, various practical TLS fingerprinting databases, such as `mod_sslhaf` [101] from Qualys SSL Labs, `p0f` [102] of Marek Majkowski, `FingerPrinTLS` [103] of Lee Brotherton, and JA3-based OSINT feeds [33,34] accumulate labeled TLS fingerprints in different goals (i.e., different types of labels), while all the databases are built for exact matching scenarios and tools. For example, ref. [99] recommends using JA3 for blacklist-based access control of TLS-encrypted malware traffic, and whitelist-based access control of legitimate applications in locked-down environments. However, despite its popularity, it is unclear whether JA3 is a reliable fingerprint for such scenarios, owing to the lack of evaluation results. To the best of our knowledge, ref. [104] is the only research on JA3's reliability. The authors insist that JA3 is not sufficient for mobile app identifications; however, a combination of JA3, JA3S, and SNI can improve reliability. Note that Kotzias et al. [105] reported 7.3% fingerprint collision in their longitudinal passive dataset while applying a client fingerprinting technique similar to JA3.

Meanwhile, in the literature, there are several proposals to adopt approximate, machine learning-based matching for TLS fingerprinting. Korczynski and Duda [90] propose using stochastic fingerprints for TLS-encrypted traffic in application classification. In this study, TLS message type sequences for each application to create a first-order homogeneous Markov chain fingerprint, and the classifier, is based on the maximum likelihood (ML) criterion. Inspired by Frolov and Wustrow [106], Anderson and McGrew [107] utilized the Levenshtein distance for approximate matching when exact matching failed, even though the approach exhibits a worse performance than exact matching. Nevertheless, approximate matching should be further studied, considering the evolution of TLS-encrypted traffic for the same label.

Additionally, `Cisco joy` and `Cisco mercury` provide the largest TLS fingerprint database labeled with potential (malicious or legitimate) application and operating system information, collected from malware sandbox and enterprise networks. However, it is not popularly adopted in the industry. In contrast, while there are multiple security tools and middleboxes to support JA3/JA3S in industry (e.g., `FlowMon` [108]) and security communities, its community database is relatively small. Hence, `JA3cury` [109] proposed a technique to translate each fingerprint record in `mercury` database into JA3.

#### 4.5. Feature Representation

While features in a flow record or a TLS fingerprint can be used as a raw input to a machine learning algorithm, in some cases, it is better to further transform into being more machine learning friendly. For example, Anderson and McGrew [17] proposed contextual flows, which correlate a TLS-encrypted traffic with DNS flows and HTTP flows to enhance the performance of the machine learning classifier (especially the accuracy at a 0.00% false discovery rate). While [17] just combines the features of the TLS flow and the contextual flow, better feature representation of the feature set could enhance the performance of the machine learning classifier.

Recently, as discussed in Section 4.3, Shen et al. [94] proposed a graph-based representation called TIG to represent decentralized application flows. The representation clearly explains packet direction, packet length, packet burst, and packet ordering information to allow the GNN to extract such information.

Another recent advancement in this field is nPrint [110]. nPrint is a complete (i.e., every bit of a packet header is included), inherently normalized (for machine learning models), and aligned (i.e., each feature is always located at the same offset for every packet) packet representation. With this representation, automated machine learning (AutoML) systems can learn the importance of each feature without relying on manual feature engineering (which is heavily conducted in Anderson and McGrew [17]'s model). In [110], the authors successfully exhibited the per-bit feature importance visually, for several traffic analysis scenarios, such as active device fingerprinting, passive OS detection, and browser and app identifications.

#### 4.6. Machine Learning Algorithms and Model Selection

Various machine learning algorithms are available owing to extensive studies in this field. Once the feature representation is completed, various the algorithms are readily applied in TLS-encrypted traffic. For example, Anderson and McGrew [96] provide a detailed comparison for malware detection among several well-known machine learning models: linear regression, logistic regression, support vector machine (SVM), decision tree, random forest, and multilayer perceptron (MLP), given a set of extensive dataset engineered by security experts. Furthermore, the researchers also considered the possibility of noisy labels. According to their work, random forest is the most robust machine learning classifier for malware detection.

When several machine learning algorithms are required to be evaluated, along with a comparison for model selection, we are required to employ performance metrics. Accuracy, precision, recall, and F1-score are the typically used metrics in encrypted NTA. One interesting metric especially proposed for SOCs is accuracy at a 0.00% false discovery rate (FDR), appeared in [17,83,96]. Note that FDR is defined as the expectation of a false positive/(false positive + true positive) [111] and performance evaluation with controlling FDR has been widely used in statistics and genomics. In contrast, while an exception [112] can be found in traffic classification literature, controlling FDR was rarely conducted in malware detection literature. As we can observe in its definition, FDR highly depends on false positives. Clearly, an incident response team in a SOC may not conduct machine learning-based methods if too many false positives occur, and a recent research [113] conduct an online survey to understand SOC analysts' perspective on this issue in depth. Thus, we can conduct feature selections for each machine learning algorithm to control the FDR. In this context, ref. [17,83] successfully justified the necessity for combining TLS metadata, SPLT, and BD feature sets to achieve better accuracy at a 0.00 % FDR, where the accuracy at a 0.00% FDR refers to the accuracy in the controlled trial.

In contrast, an increasing trend can be observed in the research efforts to conduct NTA without security experts with the assistance of deep learning. Rezaei and Liu [24] introduced a set of deep learning-based methods for traffic classification. Similarly, Aceto et al. [21] provide an excellent systematic framework for comparison of deep learning architectures for mobile encrypted traffic classification.

Meanwhile, with the advance of AutoML systems, model selection can be automated. In nPrintML [110], AutoGluon-Tabular [114] is used, which trains, optimizes, and tests over 50 machine learning models, such as tree-based methods, deep neural network models, and neighbor-based classification models.

#### 4.7. Hyperparameter Tuning

In machine learning, hyperparameter tuning is described as the process to determine the right combination of hyperparameters for a machine learning algorithm. For example, in Anderson and McGrew [96], a simple grid search over a set of standard values is performed for a cross-validation dataset. However, as highlighted in [21], the hyperparameter tuning of machine learning algorithms for encrypted traffic classification is substantially overlooked in literature. As a solution, Holland et al. [110] recently proposed nPrintML, a system to automate feature extraction and hyperparameter tuning, designed for various NTA tasks.

### 5. Conclusions

In this survey article, we discuss several TLS-encrypted NTA methods and their deployment models in the context of malware detection and family classification for security experts in SOCs. We observe that while TLS interception is widely used in industry, the rise of privacy issues leads for researchers and some vendors to recommend inspection without decryption. Another approach to utilize searchable encryption is promising, but the current generation of SOCs has no incentive to deploy such solutions owing to performance issues and an implausible assumption for malware. Thus, we discuss the state-of-the-art methods which are suitable for SOCs which inspect TLS-encrypted traffic without decryption, focusing on the machine learning-based methods. Especially, we emphasize the current trend in TLS fingerprinting in industry and academia, which can be helpful for security experts who intend to introduce machine learning-based methods in SOCs.

While a substantial number of studies have been conducted in this field, including some groundbreaking works in recent years, there is still room for further improvement as follows:

- The existing proposals have been validated in different and small datasets. While lack of diverse, large, and sharable datasets with labels is a persistent problem in NTA [115], sharing TLS fingerprints in OSINT feeds seems to be relatively plausible. Thus, designing OSINT-friendly TLS fingerprinting techniques with more features optimized for machine learning-based NTA can be a promising research direction.
- With the fast adoption of TLS 1.3, visibility of TLS-encrypted traffic using TLS interception is rapidly decreasing in many SOCs, even though the enhanced flow records are collected. It is because that in TLS 1.3, many features in TLS handshake metadata are no longer collectible due to inherent secure design. It implies that more features in TLS-encrypted traffic should be collected with novel feature representations, well-designed machine learning algorithms, and model optimization techniques, under the diverse constraints of SOCs (privacy, cost, automation, scalability, etc.). Recent advances in deep learning-based NTA can be a potential research direction.
- The current academic literature lacks consideration in real-time and online processing for NTA. Considering the higher requirements of deep learning-based methods, we may need to be aware of systematic and holistic approaches in NTA.

**Author Contributions:** Conceptualization, H.R. and C.O.; methodology, H.R.; investigation, H.R. C.O., and J.H.; writing—original draft preparation, H.R., C.O. and J.H.; writing—review and editing, C.O. and H.R.; funding acquisition, H.R. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Korea Institute of Science and Technology Information (KISTI).

**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. Zimmerman, C. *Ten Strategies of a World-Class Cybersecurity Operations Center*; The MITRE Corporation: McLean, VA, USA, 2014.
2. Vielberth, M.; Bohm, F.; Fichtinger, I.; Pernul, G. Security Operations Center: A Systematic Study and Open Challenges. *IEEE Access* **2020**, *8*, 227756–227779. [CrossRef]
3. Kokulu, F.B.; Shoshitaishvili, Y.; Soneji, A.; Zhao, Z.; Ahn, G.J.; Bao, T.; Doupé, A. Matched and Mismatched SOCs: A Qualitative Study on Security Operations Center Issues. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (ACM CCS), London, UK, 11–15 November 2019; pp. 1955–1970. [CrossRef]
4. Bejtlich, R. *The Practice of Network Security Monitoring: Understanding Incident Detection and Response*; No Starch Press: San Francisco, CA, USA, 2013.
5. Sanders, C.; Smith, J. *Applied Network Security Monitoring: Collection, Detection, and Analysis*; Syngress: Burlington, MA, USA, 2014.
6. Richardson, M.; Harris, G. *PCAP Capture File Format*; Technical Report Draft-Gharris-Opsawg-Pcap-02; Internet Engineering Task Force: Fremont, CA, USA, 2021.
7. Trammell, B.; Boschi, E. An Introduction to IP Flow Information Export (IPFIX). *IEEE Commun. Mag.* **2011**, *49*, 89–95. [CrossRef]
8. Santos, O. *Network Security with NetFlow and IPFIX: Big Data Analytics for Information Security*; Cisco Press: Indianapolis, IN, USA, 2015.
9. ENEA Qosmos Division. *Importance of Network Traffic Analysis (NTA) for SOCs*; Technical Report; ENEA Qosmos Division: Paris, France, 2019.
10. Symantec. *A Technology Brief on SSL/TLS Traffic*; Symantec Corporation World Headquarters: Mountain View, CA, USA, 2017.
11. Cisco. Cisco Encrypted Traffic Analytics. 2019. Available online: [https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/sec\\_data\\_eta/configuration/xs-16-10/sec-data-encrypted-traffic-analytics-xe-16-10-book/sec-data-encrypted-traffic-analytics-xe-16-6-book\\_chapter\\_01.pdf?dtd=oossdc000283](https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/sec_data_eta/configuration/xs-16-10/sec-data-encrypted-traffic-analytics-xe-16-10-book/sec-data-encrypted-traffic-analytics-xe-16-6-book_chapter_01.pdf?dtd=oossdc000283) (accessed on 11 November 2021).
12. Naylor, D.; Finamore, A.; Leontiadis, I.; Grunenberger, Y.; Mellia, M.; Munafò, M.; Papagiannaki, K.; Steenkiste, P. The Cost of the “S” in HTTPS. In Proceedings of the 10th Conference on Emerging Networking Experiments and Technologies (ACM CoNEXT), Sydney, Australia, 2–5 December 2014; pp. 133–140. [CrossRef]
13. Google. HTTPS Encryption on the Web. 2021. Available online: <https://transparencyreport.google.com/https/overview> (accessed on 11 November 2021).
14. “Let’s Encrypt”. Let’s Encrypt Stats. 2021. Available online: <https://letsencrypt.org/stats/> (accessed on 11 November 2021).
15. Aas, J.; Barnes, R.; Case, B.; Durumeric, Z.; Eckersley, P.; Flores-López, A.; Halderman, J.A.; Hoffman-Andrews, J.; Kasten, J.; Rescorla, E.; et al. Let’s Encrypt: An Automated Certificate Authority to Encrypt the Entire Web. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (ACM CCS) CCS ’19, London, UK, 11–15 November 2019; Association for Computing Machinery: New York, NY, USA, 2019; pp. 2473–2487. [CrossRef]
16. Mayer, W.; Schmiedecker, M. TLScompare: Crowdsourcing Rules for HTTPS Everywhere. In Proceedings of the 25th International Conference Companion on World Wide Web (WWW), Montreal, QC, Canada, 11–15 April 2016; pp. 471–476. [CrossRef]
17. Anderson, B.; McGrew, D. Identifying Encrypted Malware Traffic with Contextual Flow Data. In Proceedings of the 9th ACM Workshop on Artificial Intelligence and Security (ACM AISeC’2016), Co-Located with ACM CCS 2016, Vienna, Austria, 28 October 2016; Association for Computing Machinery, Inc.: New York, NY, USA, 2016; pp. 35–46. [CrossRef]
18. Papadogiannaki, E.; Ioannidis, S. A Survey on Encrypted Network Traffic Analysis Applications, Techniques, and Countermeasures. *ACM Comput. Surv.* **2021**, *54*, 1–35. [CrossRef]
19. Pacheco, F.; Exposito, E.; Gineste, M.; Baudoin, C.; Aguilar, J. Towards the Deployment of Machine Learning Solutions in Network Traffic Classification: A Systematic Survey. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 1988–2014. [CrossRef]
20. Velan, P.; Čermák, M.; Čeleda, P.; Drašar, M. A survey of methods for encrypted traffic classification and analysis. *Int. J. Netw. Manag.* **2015**, *25*, 355–374. [CrossRef]
21. Aceto, G.; Ciunzo, 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]
22. Conti, M.; Li, Q.Q.; Maragno, A.; Spolaor, R. The Dark Side(-Channel) of Mobile Devices: A Survey on Network Traffic Analysis. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 2658–2713. [CrossRef]
23. Poh, G.S.; Divakaran, D.M.; Lim, H.W.; Ning, J.; Desai, A. A Survey of Privacy-Preserving Techniques for Encrypted Traffic Inspection over Network Middleboxes. *arXiv* **2021**, arXiv:2101.04338.
24. Rezaei, S.; Liu, X. Deep Learning for Encrypted Traffic Classification: An Overview. *IEEE Commun. Mag.* **2019**, *57*, 76–81. [CrossRef]
25. Shen, M.; Liu, Y.; Zhu, L.; Xu, K.; Du, X.; Guizani, N. Optimizing feature selection for efficient encrypted traffic classification: A systematic approach. *IEEE Netw.* **2020**, *34*, 20–27. [CrossRef]
26. Shbair, W.M.; Cholez, T.; Francois, J.; Chrisment, I. A Survey of HTTPS Traffic and Services Identification Approaches. *arXiv* **2020**, arXiv:2008.08339.
27. De Carnavalet, X.C.; van Oorschot, P.C. A survey and Analysis of TLS Interception Mechanisms and Motivations. *arXiv* **2020**, arXiv:2010.16388.

28. McKay, K.; Cooper, D. *Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations*; Technical Report; NIST: Gaithersburg, MD, USA, 2019. [CrossRef]
29. Anderson, B.; Paul, S.; McGrew, D. Deciphering malware's use of TLS (without decryption). *J. Comput. Virol. Hacking Tech.* **2018**, *14*, 195–211. [CrossRef]
30. Warburton, D. *The 2021 TLS Telemetry Report*; Technical Report; F5 Labs: Washington, DC, USA, 2021.
31. Gallagher, S. *Nearly Half of Malware Now Use TLS to Conceal Communications*; Technical Report; SophosLabs: Tokyo, Japan, 2021.
32. WatchGuard Threat Lab. *Internet Security Report: Q2 2021*; Technical Report; Watchguard: Seattle, WA, USA, 2021.
33. Abuse.ch. No SSLBL | Malicious JA3 Fingerprints. Available online: <https://ssllbl.abuse.ch/ja3-fingerprints/> (accessed on 11 November 2021).
34. SSL Fingerprint JA3. Available online: <https://ja3er.com/> (accessed on 11 November 2021).
35. Freier, A.O.; Karlton, P.; Kocher, P.C. The Secure Sockets Layer (SSL) Protocol Version 3.0. RFC 6101. 2011. Available online: <https://www.rfc-editor.org/rfc/rfc6101> (accessed on 11 November 2021). [CrossRef]
36. Barnes, R.; Thomson, M.; Pironti, A.; Langley, A. Deprecating Secure Sockets Layer Version 3.0. RFC 7568. 2015. Available online: <https://www.rfc-editor.org/rfc/rfc7568> (accessed on 11 November 2021). [CrossRef]
37. Allen, C.; Dierks, T. The TLS Protocol Version 1.0. RFC 2246. 1999. Available online: <https://www.rfc-editor.org/rfc/rfc2246> (accessed on 11 November 2021). [CrossRef]
38. Rescorla, E. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446. 2018. Available online: <https://www.rfc-editor.org/rfc/rfc8446> (accessed on 11 November 2021). [CrossRef]
39. Moriarty, K.; Farrell, S. Deprecating TLS 1.0 and TLS 1.1. RFC 8996. 2021. Available online: <https://www.rfc-editor.org/rfc/rfc8996> (accessed on 11 November 2021). [CrossRef]
40. Qualys, I. Qualys SSL Labs—SSL Pulse. 2021. Available online: <https://www.ssllabs.com/ssl-pulse/> (accessed on 11 November 2021).
41. Rescorla, E.; Dierks, T. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246. 2008. Available online: <https://rfc-editor.org/rfc/rfc5246> (accessed on 11 November 2021). [CrossRef]
42. Eastlake, D.E., 3rd. Transport Layer Security (TLS) Extensions: Extension Definitions. RFC 6066. 2011. Available online: <https://rfc-editor.org/rfc/rfc6066> (accessed on 11 November 2021). [CrossRef]
43. Axon, L.; AlAhmadi, B.A.; Nurse, J.R.C.; Goldsmith, M.; Creese, S. Data presentation in security operations centres: Exploring the potential for sonification to enhance existing practice. *J. Cybersecur.* **2020**, *6*, tyaa004,
44. Fu, C.; Li, Q.; Shen, M.; Xu, K. Realtime Robust Malicious Traffic Detection via Frequency Domain Analysis. In Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (ACM CCS), CCS '21, Virtual, 15–19 November 2021; Association for Computing Machinery: New York, NY, USA, 2021; pp. 3431–3446. [CrossRef]
45. Bhuyan, M.H.; Bhattacharyya, D.K.; Kalita, J.K. Network anomaly detection: Methods, systems and tools. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 303–336. [CrossRef]
46. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly Detection: A Survey. *ACM Comput. Surv.* **2009**, *41*, 1–58. [CrossRef]
47. Aldweesh, A.; Derhab, A.; Emam, A.Z. Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues. *Knowl. Based Syst.* **2020**, *189*, 105124. [CrossRef]
48. Goodall, J.R.; Ragan, E.D.; Steed, C.A.; Reed, J.W.; Richardson, G.D.; Huffer, K.M.; Bridges, R.A.; Laska, J.A. Situ: Identifying and Explaining Suspicious Behavior in Networks. *IEEE Trans. Vis. Comput. Graph.* **2019**, *25*, 204–214. [CrossRef]
49. Choi, I.; Lee, J.; Kwon, T.; Kim, K.; Choi, Y.; Song, J. An Easy-to-use Framework to Build and Operate AI-based Intrusion Detection for In-situ Monitoring. In Proceedings of the 2021 16th Asia Joint Conference on Information Security (AsiaJICIS), Seoul, Korea, 19–20 August 2021; pp. 1–8. [CrossRef]
50. Smith, M. The SOC is Dead, Long Live the SOC! *ITNOW* **2020**, *62*, 34–35. [CrossRef]
51. Finsterbusch, M.; Richter, C.; Rocha, E.; Müller, J.A.; Hänßgen, K. A survey of payload-based traffic classification approaches. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 1135–1156. [CrossRef]
52. Durumeric, Z.; Ma, Z.; Springall, D.; Barnes, R.; Sullivan, N.; Bursztein, E.; Bailey, M.; Halderman, J.A.; Paxson, V. The Security Impact of HTTPS Interception. In Proceedings of the 2017 Network and Distributed System Security Symposium (NDSS), San Diego, CA, USA, 26 February–1 March 2017; Internet Society: Reston, VA, USA, 2017. [CrossRef]
53. Saltzer, J.H.; Reed, D.P.; Clark, D.D. End-to-end arguments in system design. *ACM Trans. Comput. Syst. (TOCS)* **1984**, *2*, 277–288. [CrossRef]
54. Liang, J.; Jiang, J.; Duan, H.; Li, K.; Wan, T.; Wu, J. When HTTPS meets CDN: A case of authentication in delegated service. In Proceedings of the 2014 IEEE Symposium on Security and Privacy (IEEE S&P), Berkeley, CA, USA, 18–21 May 2014; pp. 67–82. [CrossRef]
55. Huang, L.S.; Rice, A.; Ellingsen, E.; Jackson, C. Analyzing forged SSL certificates in the wild. In Proceedings of the IEEE Symposium on Security and Privacy (IEEE S&P), Berkeley, CA, USA, 18–21 May 2014; pp. 83–97. [CrossRef]
56. Dekker, M. The HTTPS Interception Dilemma: Pros and Cons. 2017. Available online: <https://www.helpnetsecurity.com/2017/03/08/https-interception-dilemma/> (accessed on 11 November 2021).
57. Clark, J.; Van Oorschot, P.C. SoK: SSL and HTTPS: Revisiting past challenges and evaluating certificate trust model enhancements. In Proceedings of the 2013 IEEE Symposium on Security and Privacy (IEEE S&P), Berkeley, CA, USA, 19–22 May 2013; pp. 511–525. [CrossRef]
58. Song, D.X.; Wagner, D.; Perrig, A. Practical techniques for searches on encrypted data. In Proceedings of the IEEE Computer Society Symposium on Security and Privacy (IEEE S&P), Berkeley, CA, USA, 14–17 May 2000; pp. 44–55. [CrossRef]

59. Curtmola, R.; Garay, J.; Kamara, S.; Ostrovsky, R. Searchable symmetric encryption. In Proceedings of the 13th ACM Conference on Computer and Communications Security (ACM CCS), Virginia, VA, USA, 3 October–3 November 2006; Volume 402, pp. 79–88. [CrossRef]
60. Bösch, C.; Hartel, P.; Jonker, W.; Peter, A. A survey of provably secure searchable encryption. *ACM Comput. Surv.* **2014**, *47*, 1–51. [CrossRef]
61. O’Neill, M.; Ruoti, S.; Seamons, K.; Zappala, D. TLS Proxies. In Proceedings of the 2016 ACM Internet Measurement Conference (ACM IMC), Santa Monica, CA, USA, 14–16 November 2016; ACM: New York, NY, USA, 2016; pp. 551–557. [CrossRef]
62. Waked, L.; Mannan, M.; Youssef, A. The Sorry State of TLS Security in Enterprise Interception Appliances. *Digit. Threat. Res. Pract.* **2020**, *1*, 1–26. [CrossRef]
63. Sherry, J.; Lan, C.; Popa, R.A.; Ratnasamy, S. BlindBox. *ACM SIGCOMM Comput. Commun. Rev.* **2015**, *45*, 213–226. [CrossRef]
64. Lan, C.; Sherry, J.; Popa, R.A.; Ratnasamy, S.; Liu, Z. Embark: Securely Outsourcing Middleboxes to the Cloud. In Proceedings of the 13th USENIX Symposium on Networked Systems Design and Implementation (USENIX NSDI), Santa Clara, CA, USA, 16–18 March 2016; pp. 255–273.
65. Yuan, X.; Wang, X.; Lin, J.; Wang, C. Privacy-preserving deep packet inspection in outsourced middleboxes. In Proceedings of the 35th Annual IEEE International Conference on Computer Communications (IEEE INFOCOM), San Francisco, CA, USA, 10–14 April 2016. [CrossRef]
66. Ning, J.; Poh, G.S.; Loh, J.C.; Chia, J.; Chang, E.C. PrivDPI: Privacy-preserving encrypted traffic inspection with reusable obfuscated rules. In Proceedings of the ACM Conference on Computer and Communications Security (ACM CCS), New York, NY, USA, 11–15 November 2019; pp. 1657–1670. [CrossRef]
67. Canard, S.; Diop, A.; Kheir, N.; Paindavoine, M.; Sabt, M. BlindIDS: Market-Compliant and Privacy-Friendly Intrusion Detection System over Encrypted Traffic. In Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security (ACM ASIACCS), ASIA CCS ’17, Abu Dhabi, United Arab Emirates, 2–6 April 2017; Association for Computing Machinery: New York, NY, USA, 2017; pp. 561–574. [CrossRef]
68. Baek, J.; Kim, J.; Susilo, W. Inspecting TLS Anytime Anywhere: A New Approach to TLS Interception. In Proceedings of the 15th ACM Asia Conference on Computer and Communications Security (ACM ASIACCS), ASIA CCS ’20, Taipei, Taiwan, 5–9 October 2020; Association for Computing Machinery: New York, NY, USA, 2020; pp. 116–126. [CrossRef]
69. Kim, J.; Camtepe, S.; Baek, J.; Susilo, W.; Pieprzyk, J.; Nepal, S. P2DPI: Practical and Privacy-Preserving Deep Packet Inspection. In Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security, ASIA CCS ’21, Virtual Event, 7–11 June 2021; Association for Computing Machinery: New York, NY, USA, 2021; pp. 135–146. [CrossRef]
70. Han, J.; Kim, S.; Ha, J.; Han, D. SGX-Box. In Proceedings of the First Asia-Pacific Workshop on Networking (APNet), Hong Kong, China, 3–4 August 2017; ACM: New York, NY, USA, 2017; pp. 99–105. [CrossRef]
71. Naylor, D.; Li, R.; Gkantsidis, C.; Karagiannis, T.; Steenkiste, P. And then there were more: Secure communication for more than two parties. In Proceedings of the 13th International Conference on Emerging Networking EXperiments and Technologies (ACM CoNEXT 2017), New York, NY, USA, 12–15 December 2017; pp. 88–100. [CrossRef]
72. Costan, V.; Devadas, S. Intel SGX Explained. Technical Report. 2016. Available online: <http://css.csail.mit.edu/6.858/2020/readings/costan-sgx.pdf> (accessed on 11 November 2021).
73. Papadogiannaki, E.; Ioannidis, S. Acceleration of Intrusion Detection in Encrypted Network Traffic Using Heterogeneous Hardware. *Sensors* **2021**, *21*, 1140. [CrossRef] [PubMed]
74. Karagiannis, T.; Papagiannaki, K.; Faloutsos, M. BLINC: Multilevel Traffic Classification in the Dark. In Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, SIGCOMM ’05, Philadelphia, PA, USA, 22 August 2005; Association for Computing Machinery: New York, NY, USA, 2005; pp. 229–240. [CrossRef]
75. Velan, P.; Medková, J.; Jirsík, T.; Čeleda, P. Network traffic characterisation using flow-based statistics. In Proceedings of the 2016 IEEE/IFIP Network Operations and Management Symposium (IEEE/IFIP NOMS), Istanbul, Turkey, 25–29 April 2016; pp. 907–912. [CrossRef]
76. Sanders, C. *Practical Packet Analysis: Using Wireshark to Solve Real-World Network Problems*, 3rd ed.; No Starch Press: San Francisco, CA, USA, 2017.
77. Collins, M. *Network Security through Data Analysis: From Data to Action*, 2nd ed.; O’Reilly: Sebastopol, CA, USA, 2017; p. 428.
78. Rohde & Schwarz Company. R&S® PACE 2—First Packet Classification in An Encrypted World. 2021. Available online: <https://www.ipoque.com/news-media/resources/brochures/dpi-engine-pace-2-first-packet-classification> (accessed on 11 November 2021).
79. Deri, L.; Martinelli, M.; Bujlow, T.; Cardigliano, A. nDPI: Open-source high-speed deep packet inspection. In Proceedings of the 2014 International Wireless Communications and Mobile Computing Conference (IWCMC), Nicosia, Cyprus, 4–8 August 2014; pp. 617–622. [CrossRef]
80. Bernaille, L.; Teixeira, R.; Salamatian, K. Early Application Identification. In Proceedings of the 2nd Conference on Future Networking Technologies (ACM CoNEXT), New York, NY, USA, 4–7 December 2006. [CrossRef]
81. Bernaille, L.; Teixeira, R. Implementation issues of early application identification. In Proceedings of the 3rd Asian Conference on Internet Engineering: Sustainable Internet (AINTEC), Phuket, Thailand, 27–29 November 2007; pp. 156–166. [CrossRef]
82. Bernaille, L.; Teixeira, R. Early recognition of encrypted applications. In Proceedings of the 8th International Conference on Passive and Active Network Measurement (PAM), Louvain-la-Neuve, Belgium, 5–6 April 2007; pp. 165–175. [CrossRef]

83. McGrew, D.; Anderson, B. Enhanced telemetry for encrypted threat analytics. In Proceedings of the 24th IEEE ICNP Workshop on Machine Learning in Computer Networks (NetworkML 2016), Singapore, 8 November 2016; pp. 1–6. [CrossRef]
84. McGrew, D.; Anderson, B.; Perricone, P.; Hudson, B. Joy: A Package for Capturing and Analyzing Network Flow Data and Intraflow Data, for Network Research, Forensics, and Security Monitoring. 2016. Available online: <https://github.com/cisco/joy> (accessed on 11 November 2021).
85. Tegeler, F.; Fu, X.; Vigna, G.; Kruegel, C. BotFinder: Finding Bots in Network Traffic Without Deep Packet Inspection. In Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies (ACM CoNEXT, 2012), Nice, France, 10–13 December 2012; ACM Press: New York, NY, USA, 2012; p. 349. [CrossRef]
86. Alahmadi, B.A.; Mariconti, E.; Spolaor, R.; Stringhini, G.; Martinovic, I. BOTection: Bot Detection by Building Markov Chain Models of Bots Network Behavior. In Proceedings of the 15th ACM Asia Conference on Computer and Communications Security, (ACM ASIACCS), New York, NY, USA, 5–9 October 2020; pp. 652–664. [CrossRef]
87. AlAhmadi, B.A.; Martinovic, I. MalClassifier: Malware Family Classification Using Network Flow Sequence Behaviour. In Proceedings of the 13th APWG Symposium on Electronic Crime Research (eCrime), San Diego, CA, USA, 15–17 May 2018; pp. 1–13. [CrossRef]
88. Paxson, V. Bro: A system for detecting network intruders in real-time. *Comput. Netw.* **1999**, *31*, 2435–2463. [CrossRef]
89. Bartos, K.; Sofka, M.; Systems, C.; Franc, V.; Bartos, K.; Sofka, M. Optimized Invariant Representation of Network Traffic for Detecting Unseen Malware Variants. In Proceedings of the 25th USENIX Security Symposium (USENIX Security), Austin, TX, USA, 10–12 August 2016; pp. 807–822.
90. Korczyński, M.; Duda, A. Markov Chain Fingerprinting to Classify Encrypted Traffic. In Proceedings of the 33rd IEEE International Conference on Computer Communications (IEEE INFOCOM), Toronto, ON, Canada, 27 April–2 May 2014. [CrossRef]
91. Shen, M.; Wei, M.; Zhu, L.; Wang, M. Classification of Encrypted Traffic with Second-Order Markov Chains and Application Attribute Bigrams. *IEEE Trans. Inf. Forensics Secur.* **2017**, *12*, 1830–1843. [CrossRef]
92. Fu, Y.; Xiong, H.; Lu, X.; Yang, J.; Chen, C. Service Usage Classification with Encrypted Internet Traffic in Mobile Messaging Apps. *IEEE Trans. Mob. Comput.* **2016**, *15*, 2851–2864. [CrossRef]
93. Liu, C.; He, L.; Xiong, G.; Cao, Z.; Li, Z. FS-Net: A Flow Sequence Network for Encrypted Traffic Classification. In Proceedings of the 38th IEEE International Conference on Computer Communications (IEEE INFOCOM), Paris, France, 29 April–2 May 2019. [CrossRef]
94. Shen, M.; Zhang, J.; Zhu, L.; Xu, K.; Du, X. Accurate Decentralized Application Identification via Encrypted Traffic Analysis Using Graph Neural Networks. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 2367–2380. [CrossRef]
95. Paine, K.; Whitehouse, O.; Sellwood, J. *Indicators of Compromise (IoCs) and Their Role in Attack Defence*; Technical Report Draft-Paine-Smart-Indicators-of-Compromise-03; Internet Engineering Task Force: Fremont, CA, USA, 2021.
96. Anderson, B.; McGrew, D. Machine Learning for Encrypted Malware Traffic Classification: Accounting for Noisy Labels and Non-stationarity. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (ACM KDD), Halifax, NS, Canada, 13–17 August 2017; Volume Part F1296. [CrossRef]
97. Husák, M.; Čermák, M.; Jirsík, T.; Čeleda, P. HTTPS traffic analysis and client identification using passive SSL/TLS fingerprinting. *EURASIP J. Inf. Secur.* **2016**, *2016*, 1–14. [CrossRef]
98. Laperdrix, P.; Bielova, N.; Baudry, B.; Avoine, G. Browser Fingerprinting: A Survey. *ACM Trans. Web* **2020**, *14*, 1–33. [CrossRef]
99. Althouse, J.B.; Atkinson, J.; Atkins, J. Open Sourcing JA3: SSL/TLS Client Fingerprinting for Malware Detection. 2017. Available online: <https://engineering.salesforce.com/open-sourcing-ja3-92c9e53c3c41> (accessed on 11 November 2021).
100. Benjamin, D. Applying Generate Random Extensions And Sustain Extensibility (GREASE) to TLS Extensibility. RFC 8701. 2020. Available online: <https://rfc-editor.org/rfc/rfc8701.txt> (accessed on 11 November 2021). [CrossRef]
101. Ristic, I. HTTP Client Fingerprinting Using SSL Handshake Analysis. 2009. Available online: <https://www.ssllabs.com/projects/client-fingerprinting/> (accessed on 11 November 2021).
102. Majkowski, M. SSL Fingerprinting for p0f. 2012. Available online: <https://idea.popcount.org/2012-06-17-ssl-fingerprinting-for-p0f/> (accessed on 11 November 2021).
103. Brotherston, L. TLS Fingerprinting: Smarter Defending & Stealthier Attacking. 2015. Available online: <https://blog.squarelemon.com/tls-fingerprinting/> (accessed on 11 November 2021).
104. Matoušek, P.; Burgetová, I.; Ryšavý, O.; Victor, M. On Reliability of JA3 Hashes for Fingerprinting Mobile Applications. In Proceedings of the 12th EAI International Conference on Digital Forensics & Cyber Crime (EAI ICDF2C), Singapore, 7–9 December 2021; Volume 351, pp. 1–22. [CrossRef]
105. Kotzias, P.; Paterson, K.G.; Razaghpahan, A.; Vallina-Rodriguez, N.; Amann, J.; Caballero, J. Coming of age: A longitudinal study of TLS deployment. In Proceedings of the Internet Measurement Conference, Boston, MA, USA, 31 October–2 November 2018; pp. 415–428. [CrossRef]
106. Frolov, S.; Wustrow, E. The use of TLS in Censorship Circumvention. In Proceedings of the 2019 Network and Distributed System Security Symposium (NDSS), San Diego, CA, USA, 24–27 February 2019; Internet Society: Reston, VA, USA, 2019. [CrossRef]
107. Anderson, B.; McGrew, D. Accurate TLS Fingerprinting using Destination Context and Knowledge Bases. *arXiv* **2020**, arXiv:2009.01939.
108. Artur, K.; Tomas, V.; Roman, L. Encrypted Traffic Analysis: The Data Privacy-Preserving Way to Regain Visibility into Encrypted Communication. 2019. Available online: <https://www.flowmon.com/en/solutions/security-operations/encrypted-traffic-analysis> (accessed on 11 November 2021).



109. Hynek, K.; Luk, C. JA3curey—A New Approach to TLS Fingerprinting by Merging Fingerprinting Methods. Presented at Excel@FIT 2021. Available online: <http://excel.fit.vutbr.cz/submissions/2021/013/13.pdf> (accessed on 11 November 2021).
110. Holland, J.; Schmitt, P.; Feamster, N.; Mittal, P. New Directions in Automated Traffic Analysis. In Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (ACM CCS), CCS '21, Virtual Event, 15–19 November 2021; Association for Computing Machinery: New York, NY, USA, 2021; pp. 3366–3383. [[CrossRef](#)]
111. Benjamini, Y.; Hochberg, Y. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. *J. R. Stat. Soc. Ser. B Methodol.* **1995**, *57*, 289–300. [[CrossRef](#)]
112. Nechay, D.; Pointurier, Y.; Coates, M. Controlling False Alarm/Discovery Rates in Online Internet Traffic Flow Classification. In Proceedings of the IEEE International Conference on Computer Communications (IEEE INFOCOM), Rio de Janeiro, Brazil, 19–25 April 2009; pp. 684–692. [[CrossRef](#)]
113. Alahmadi, B.A.; Axon, L.; Martinovic, I. 99% False Positives: A Qualitative Study of SOC Analysts' Perspectives on Security Alarms. In Proceedings of the 31st USENIX Security Symposium (USENIX Security), Boston, MA, USA, 10–12 August 2022. Available online: <https://www.usenix.org/conference/usenixsecurity22/presentation/alahmadi> (accessed on 11 November 2021).
114. Erickson, N.; Mueller, J.; Shirkov, A.; Zhang, H.; Larroy, P.; Li, M.; Smola, A. AutoGluon-Tabular: Robust and Accurate AutoML for Structured Data. *arXiv* **2020**, arXiv:2003.06505.
115. Dainotti, A.; Pescapé, A.; Claffy, K.C. Issues and future directions in traffic classification. *IEEE Netw.* **2012**, *26*, 35–40. [[CrossRef](#)]