This article has been accepted for publication in IEEE Communications Surveys & Tutorials. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/COMST.2022.3208196

Machine Learning-Powered Encrypted Network Traffic Analysis: A Comprehensive Survey

Meng Shen, *Member, IEEE,* Ke Ye, Xingtong Liu, Liehuang Zhu, *Member, IEEE,* Jiawen Kang, *Member, IEEE,* Shui Yu, *Senior Member, IEEE,* Qi Li, *Senior Member, IEEE,* Ke Xu, *Senior Member, IEEE*

***Abstract*—Traffic analysis is the process of monitoring network activities, discovering specific patterns, and gleaning valuable information from network traffic. It can be applied in various fields such as network assert probing and anomaly detection. With the advent of network traffic encryption, however, traffic analysis becomes an arduous task. Due to the invisibility of packet payload, traditional traffic analysis methods relying on capturing valuable information from plaintext payload are likely to lose efficacy. Machine learning has been emerging as a powerful tool to extract informative features without getting access to payload, and thus is widely employed in encrypted traffic analysis.**

In this paper, we present a comprehensive survey on recent achievements in machine learning-powered encrypted traffic analysis. To begin with, we review the literature in this area and summarize the analysis goals that serve as the basis for literature classification. Then, we abstract the workflow of encrypted traffic analysis with machine learning tools, including traffic collection, traffic representation, traffic analysis method, and performance evaluation. For the surveyed studies, the requirements of classification granularity and information timeliness may vary a lot for different analysis goals. Hence, in terms of the goal of traffic analysis, we present a comprehensive review on existing studies according to four categories: network asset identification, network characterization, privacy leakage detection, and anomaly detection. Finally, we discuss the challenges and directions for future research on encrypted traffic analysis.

***Index Terms*—Encrypted traffic analysis, traffic classification, machine learning, deep learning, anomaly detection.**

I. Introduction

WITH the rapid increase of Internet traffic, the security of network connections becomes significantly crucial, as a large amount of user sensitive information is transmitted on the Internet, such as bank accounts and payment records. To ensure security and privacy, data encryption technologies, e.g., Secure Socket Layer/Transport Layer Security (SSL/TLS) [1], have been widely used to protect network connections. According to the Google Transparency Report [2], more than 95% of services provided by Google have applied encryption protocols to protect their connections, so that the resulting data packets are transmitted in a more secure way and can only be decrypted by legitimate receivers.

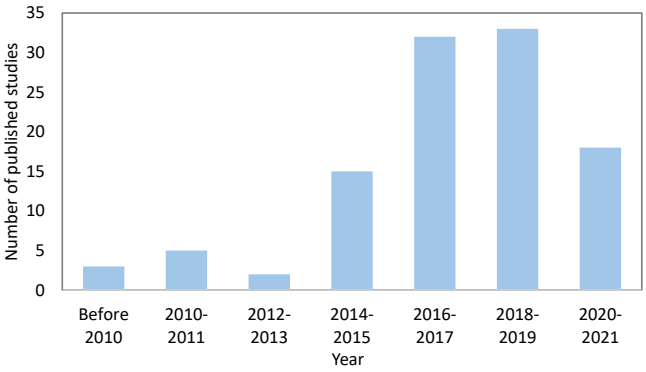


Fig. 1: Number of investigated papers on encrypted traffic analysis sorted by published year

The ever-growing encrypted traffic, however, brings new challenges to network traffic analysis, which is a useful tool for administrators in network management and network anomaly detection. Traditional traffic analysis methods usually rely on valuable information from *plaintext* payload [11]. They are likely to lose efficacy for encrypted traffic, as payload information is no longer available. For instance, many attackers take advantage of encryption protocols to hide malicious contents and evade anomaly detection. It becomes more difficult for network administrators to find suspicious patterns in encrypted traffic. As another example, network service providers (ISPs) usually measure delivery quality (e.g., video resolution, stall frequency) of video streams by analyzing contents in HTTP request header [12, 13] and then take actions accordingly. However, end-to-end encryption adopted by content providers sets a new barrier for ISPs to measure video delivery quality.

From the perspective of end users, traffic encryption also poses new threats to user privacy. In general, traffic encryption protocols (e.g., TLS) can protect Internet users from eavesdroppers that attempt to decipher or modify the content in their network connections. However, the privacy of end users is still threatened by advanced side-channel attacks. For instance, an eavesdropper can record their encrypted traffic and learn sensitive information, such as the websites a user is visiting

This work is partially supported by National Key R&D Program of China with No. 2020YFB1006101, NSFC Projects with Nos. 62222201, 62102099, 62132011, 61972039 and 61932016, Beijing Nova Program with No. Z201100006820006, China National Funds for Distinguished Young Scientists with No. 61825204, Beijing Outstanding Young Scientist Program with No. BJJWZYJH012019100003011.

M. Shen, X. Liu, and L. Zhu are with the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China. (e-mail: {shenmeng, liuxingtong, liehuangz}@bit.edu.cn).

K. Ye is with the School of Computer Science, Beijing Institute of Technology, Beijing, China (e-mail: yeke@bit.edu.cn).

J. Kang is with the School of Automation, Guangdong University of Technology, Guangzhou, China (e-mail: kavinkang@gdut.edu.cn).

S. Yu is with the School of Computer Science, University of Technology Sydney, NSW, Australia (e-mail: shui.yu@uts.edu.au).

Q. Li is with the Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing, China (e-mail: qli01@tsinghua.edu.cn).

K. Xu is with the Department of Computer Science, Tsinghua University, Beijing, China (e-mail: xuke@tsinghua.edu.cn).

TABLE I: The differences between other existing works and our survey

	Survey Paper	Year	Description
Uncrypted	Buczak *et. al.* [3]	2016	Introducing data mining and machine learning methods used for cyber security intrusion detection
	Jing *et. al.* [4]	2018	Summarizing the security data and data analysis methods for DDoS and Worm attack detection
	Fernandes *et.al.* [5]	2019	Overviewing the network data types and methods for network anomaly detection
	Kwon *et.al.* [6]	2017	Surveying the deep learning methods used in network anomaly detection
Encrypted	Velan *et.al.* [7]	2015	Summarizing approaches for encrypted traffic analysis, mainly focusing on traditional machine learning methods
	Rezaei *et.al.* [8]	2019	Overviewing the deep learning technique for encrypted traffic classification
	Conti *et.al.* [9]	2018	Review the studies that contributed to network traffic analysis targeting mobile devices
	Pacheco *et.al.* [10]	2018	Introducing machine learning solutions in network traffic classification
	Our Survey	2021	Providing a survey of machine learning-powered encrypted traffic analysis methods categorized by analysis goals, including network asset identification, network characterization, privacy leakage detection and anomaly detection

and the actions taken by a user in mobile applications [14–16]. Encrypted traffic analysis provides us with a useful tool to get more insights into the information leakage from network connections, and then defenses can be designed and implemented accordingly.

To deal with these new challenges caused by traffic encryption, machine learning techniques have been employed to extract useful information from encrypted traffic, with no need for access to packet payloads. The machine learning-powered encrypted traffic analysis leverages statistical features or behavioral features of encrypted traffic, which are less affected when encryption protocols are adopted. These methods show great capabilities in dealing with extremely large amounts of data, which is suitable for building classification models without being specifically programmed. Moreover, as the branch of machine learning, deep learning obviates the process of manual feature extraction, which makes it a desirable approach for encrypted traffic analysis, especially in dealing with the constantly varying traffic patterns.

There are a fruitful number of studies on encrypted traffic analysis during the past decade. We investigated 108 papers that were published between 2007 and 2021, as depicted in Fig. 1. Among the literature investigated, machine learning techniques play an important role in encrypted traffic analysis. Therefore, it is quite necessary to conduct a comprehensive survey that summarizes the recent achievements in machine learning-powered encrypted traffic analysis and sheds a new light on future research directions.

A. Differences From Existing Surveys

With the rapid development of machine learning algorithms, there are many surveys on applications of machine learning in various scenarios, including cognitive radios [17], computer vision [18, 19], Internet of Things (IoT) [20], economics and econometrics [21]. For instance, Bkassiny *et al.* [17] discuss the role of various machine learning models in cognitive radios, which is defined as radio devices that can adapt to the environment automatically. Wang *et al.* [18] provide an overview of generative adversarial networks applied in computer vision, including high-quality image generation, diverse image generation and stabilizing training. Different from these surveys, this survey focuses on the applications of machine learning in encrypted traffic analysis.

There are several surveys on traffic analysis, which are summarized in Table I. Buczak *et al.* [3] conduct a survey of machine learning and data mining methods applied for intrusion detection. Jing *et al.* [4] focus on the Distributed Denial of Service (DDoS) and Worm attack detection. Fernandes *et al.* [5] and Kwon *et al.* [6] investigate traffic analysis methods for network anomaly detection. These surveys are not comprehensive enough, as they only concern about malicious behavior detection, where most studies focus on unencrypted network traffic. Due to the invisibility of packet payload, methods for unencrypted traffic relying on plaintext payload signatures are likely to lose efficacy for encrypted traffic.

Several recent surveys pay attention to traffic analysis methods applicable in the encrypted scenario [7–10]. Among these studies, Conti *et al.* [9] review the applications of traffic analysis targeting *mobile* devices, and Pacheco *et al.* [10] review the application of machine learning in traffic analysis, including both unencrypted and encrypted traffic. Numerous recent studies in this field show two research trends: 1) encrypted traffic analysis can find its applications in a wider range of scenarios in both fixed and mobile networks, from network asset recognition to network anomaly detection, and 2) deep learning techniques are increasingly employed to demonstrate their superiority over traditional machine learning models in encrypted traffic analysis. Therefore, compared with the existing surveys, we cover an extensive range of applications of encrypted traffic analysis based on the advanced machine learning techniques.

B. Contributions

The popularity of encrypted communication makes a survey dedicated to encrypted traffic analysis necessary. Compared with surveys that have been published on network traffic classification, we systematically introduce the encryption protocols and pay more attention to the complete workflow of encrypted traffic analysis. Moreover, we focus on machine learning techniques that are widely employed for a variety of analysis goals. The main contributions of this survey are summarized as follows:

1) We abstract the workflow of encrypted traffic analysis from a great amount of concrete traffic analysis approaches, which presents an overview that helps readers grasp the general process on traffic analysis, including

This article has been accepted for publication in IEEE Communications Surveys & Tutorials. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/COMST.2022.3208196

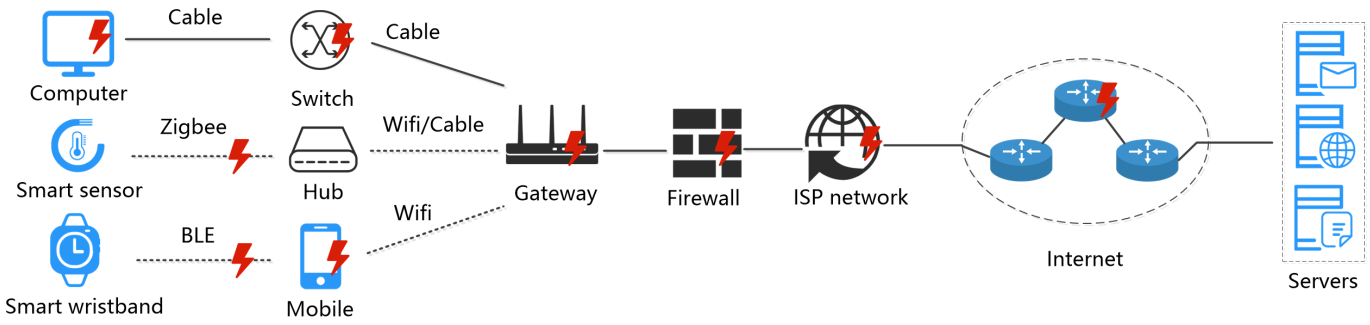


Fig. 2: System model of encrypted traffic analysis

traffic collection, traffic representation, traffic analysis method design, and performance evaluation.

2) To the best of our knowledge, we are the first to survey existing studies that employ machine learning techniques in encrypted traffic analysis. We present a systematic classification of the state-of-the-art methods according to analysis goals, including network asset identification, network characterization, privacy leakage detection and anomaly detection. This type of organization allows us to exhibit the variety of classification granularity and information timeliness among different analysis goals.
3) We provide further insights into the defects of existing studies, and discuss in detail the future research challenges and directions on encrypted traffic analysis, which provides readers with possible directions for developing innovative solutions.

C. Survey Organization

We commence with an overview on the application scenarios of encrypted traffic analysis in Section II-B, which depicts a whole picture according to their analysis goals. In Section III-C, we introduce the background of machine learning techniques that are commonly used in encrypted traffic analysis. Then, we present a description of the general framework of encrypted traffic analysis in Section IV, which serves as a guideline to review and compare the studies with specific analysis goals.

Sections V-VIII introduce recent studies on encrypted traffic analysis according to the 4 types of analysis goals. In each section, we attempt to extract the information of each literature following the stages of the general framework in Section IV. By this way, we can observe that the similarities and differences thoroughly among the existing studies.

Next, Section IX elaborates on the challenges and future research directions of encrypted traffic analysis. We still follow the steps in the general framework of encrypted traffic analysis and state the research challenges and opportunities in traffic dataset construction, traffic representation, analysis model building, and proposed countermeasures.

Finally, Section X concludes this survey. We summarize the achievements made by state-of-the-art on encrypted traffic analysis to highlight the valuable information that can be excavated in traffic encryption scenarios. We also point out potential research directions that require further investigations.

II. Overview of Encrypted Traffic Analysis

In this section, we first introduce a general system model of encrypted traffic analysis, and then propose a criterion, according to which we can classify existing papers using a layered structure. The list of common abbreviations and explanations are summarized in Table II.

A. System Model

The system model of encrypted traffic analysis is shown in Fig. 2. Encrypted traffic generated by different kinds of end devices, e.g., computers, IoT devices, and mobile phones, is sent to remote servers, which may pass through the gateway, firewall, local ISP networks and the Internet. Traffic monitors can collect traffic at different points along this path and then conduct traffic analysis according to their goals. For example, a network administrator may monitor whether the network is under attack by analyzing the traffic passing by the firewall; an attacker can refer from an end user's traffic which website or application the user is visiting. Moreover, ISPs can measure network service quality perceived by end users based on the traffic traversing their networks.

B. Taxonomy of Encrypted Traffic Analysis

Considering that a fruitful of papers have been published during the last decade, it is challenging to classify these papers according to appropriate criteria. Structuring the literature in a comprehensive and clear way is non-trivial, as we would end up with completely different categories according to various classification criteria.

The requirements of classification granularity and information timeliness may vary between different analysis goals for the surveyed studies. For instance, network attack detection is generally regarded as a binary classification problem (i.e., whether the traffic is malicious) while website fingerprinting is regarded as a multi-class problem (i.e., which website the user is visiting). Compared with website fingerprinting, QoE metric measurement has a higher requirement for real-time performance, since an ISP needs to adjust the quality of content transmission according to the actual situation. Accordingly, encrypted traffic analysis methods highly depend on the analysis goals, including traffic feature extraction and machine learning model selection. The researchers attempt to optimize

TABLE II: Common abbreviations and explanations used in this paper

Abbreviation	Explanation	Abbreviation	Explanation	Abbreviation	Explanation
AF	Application Fingerprinting	CNN	Convolutional Neural Network	DDoS	Distributed Denial-of-Service
DT	Decision Tree	GNN	Graph Neural Network	IoT	Internet of Things
IPSec	Internet Protocol Security	ISP	Internet Service Provider	k-NN	k Nearest Neighbor
LSTM	Long Short-Term Memory	OS	Operating System	QoE	Quality of Experience
QoS	Quality of Service	QUIC	Quick UDP Internet Connection	R2L	Remote to Local
RF	Random Forest	SDN	Software Defined Network	SSH	Secure Shell
SSL	Secure Socket Layer	SVM	Support Vector Machine	TLS	Transport Layer Security
Tor	The Onion Router	U2R	User to Root	WF	Website Fingerprinting

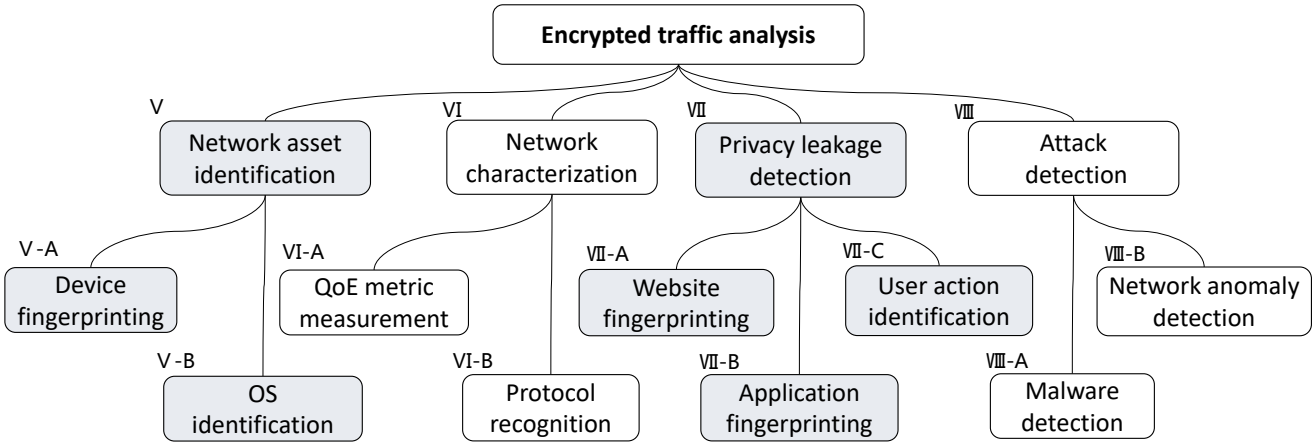


Fig. 3: Classification of existing studies on encrypted traffic analysis according to their analysis goals

their analysis methods to meet the requirements of a specific application scenario. Thus, we group the existing studies by the analysis goals, as illustrated in the hierarchical structure in Fig. 3. At the top level, we identify four macro-goals that correspond to different application domains, including network asset identification, network characterization, privacy leakage detection, and anomaly detection. At the lower level, several micro-goals are summarized within each application domain, corresponding to the specific targets.

- *Network asset identification* targets identifying physical network equipment and the operating system (OS). On the one hand, with an increasing number of network devices connected to the Internet, it becomes more difficult for network administrators to fully understand the network assets under their control. On the other hand, malicious attackers can accurately grasp the vulnerabilities of devices by identifying which version it is. As different types of network devices and OS versions may lead to various characteristics in communication traffic, we can perform network asset identification based on traffic analysis even if the traffic is encrypted. Coupled with the above two aspects, we shall introduce a variety of analysis methods as well as traffic representation methods.
- *Network characterization* is to have an understanding of service delivery quality and related protocols by analyzing the corresponding traffic. Video streaming traffic shows a trend of rapid growth, where the demands for high bandwidth and low latency increase accordingly. The control of transmission quality by service providers is inseparable from the perception of user experience. However, with the adoption of encrypted protocols, such as SSL/TLS and QUIC, video streaming services produce an increasing amount of encrypted traffic, leaving limited features for network characterization. To solve this issue, a lot of studies focus on measuring and characterizing the video-based service delivery quality from encrypted traffic, e.g., the QoE perceived by end users. These characteristics can help network providers to figure out the long-term or short-term quality of their network service and optimize their routing strategies.
- *Privacy leakage detection* focuses on the analysis of information that may be leaked by encrypted traffic. Although the encryption protocols are proposed to protect the content of traffic packet, there are still differences in the traffic of different websites or applications, which provides a possible way for privacy leakage, such as what website or application a victim is visiting, as well as the in-app actions during the visits (e.g., sending an email in Gmail [22]). This difference may be reflected in features such as packet length, peak packet numbers, etc. With this

target, we review three categories of privacy leakage, i.e., Website Fingerprinting (WF), Application Fingerprinting (AF), and user action identification.

- *Attack detection* mainly aims to detect diverse malware and network anomaly. Recently, we have witnessed a rapid growth of malware targeting PCs, mobile phones and IoT devices, such as WannaCry and Petya. There are also an increasing number of network attacks on various platforms, such as enterprise networks, campus networks, IoT networks and blockchain networks [23–25]. The adoption of encryption techniques makes payload-based anomaly detection methods ineffective, as the traditional detection methods commonly scan packet contents to figure out malicious patterns based on the signature library. However, the traffic generated by abnormal behaviors is still different from that of legitimate behaviors, making it possible to distinguish abnormal traffic from benign traffic even in an encrypted scenario.

III. Background of Machine Learning

As this survey mainly focuses on the application of machine learning in encrypted traffic analysis, in this section, we briefly introduce the background of machine learning and review several commonly-used algorithms.

Machine learning aims to build models that can improve the future performance of a target through learning from the past experiences [26–30]. It is a highly interdisciplinary field based on other fields such as statistics and optimization theory [31], which serves multiple tasks in different scenarios, e.g., medical industry [32], IoT [20] and financial industry [21]. Machine learning also plays an important role in encrypted traffic classification, which can be demonstrated by the fact that almost 85% of the studies investigated employ machine learning methods for various analysis goals.

A. Categories of Machine Learning Methods

From the aspect of whether labels are required, machine learning can be classified as supervised learning, unsupervised learning, semi-supervised learning and reinforcement learning. As for supervised learning, the input data has a pre-determined label, e.g, the type of traffic. The model is trained to achieve a higher level of classification accuracy. For unsupervised learning, there is no label for training dataset, the model is designed by deducing existing patterns of samples [26]. Semi-supervised learning is used for the dateset with both labeled and unlabeled samples, typically most of them are unlabeled as unlabeled data is less expensive [33]. Reinforcement learning is trained to decide actions yield the most rewards by the trial and error, which is based on the interactions with environments and past experiences of learning [34].

Taking supervised learning as an example, we introduce a general model of machine learning. Supervised learning uses a training dataset to train a model, which is leveraged for the prediction of the test dataset. For a training dataset $D = \{(x_i, y_i)\}, i = 1, 2, .., N$, where $x_i \in \mathbb{X}$ denotes an input sample (e.g., an extracted feature vector), and $y_i \in \mathbb{Y}$ denotes an output sample (e.g., a specific label for classification problems). In the training phase, a decision model $\hat{Y} = f(X)$ is obtained, which maps the inputs to outputs. Through the training algorithm, the decision model tries to minimize the differences between the predicted value $\hat{y_i}$ and the real value y based on a loss function $L(y_i, f(x_i))$. The model that minimizes the loss function is taken as the final decision model $f^*(X)$. In the testing phase, for each sample in the test dataset, the model gives the corresponding prediction value.

Inspired by the existing surveys [35, 36], we also consider model complexity as another criterion to categorize machine learning methods. According to this criterion, machine learning can be roughly divided into two categories, namely *traditional machine learning methods* and *deep learning methods*. The main difference of them is that most deep learning methods leverage cascades of neural network layers, which contain nonlinear processing units for feature extraction [37]. In contrast, traditional machine learning methods without nonlinear processing units usually require more feature engineering [38].

For traffic analysis, feature extraction is a crucial part. To improve the accuracy of traffic classification, many studies focus on how to extract effective semantic information from byte stream to fed into traditional machine learning models. However, with the advent of deep learning, feature engineering is no longer the only concern. The combination of raw information and neural network models becomes another way to achieve high accuracy. We observed that feature engineering combined with traditional machine learning, and raw information combined with deep learning are the two common types of approaches for traffic analysis. The former focuses more on the extraction of effective features, while the latter focuses more on the construction of models with strong feature extraction capability. Due to this reason, we will mainly introduce machine learning according to this criterion.

B. Traditional Machine Learning Methods

Typical examples of traditional machine learning methods include Naïve Bayes, Markov Model, k-Nearest Neighbor (k-NN), Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF), and clustering, which are briefly described as follows.

- *Naïve Bayes* [39] method is a classification method based on Bayes' theorem and condition independence. It is called naïve because it assumes that the features are independent of each other. It can make probabilistic inferences for classification and decision-making problems.
- *Markov Model* [40] is a stochastic model, which gives the probabilities of random state changing. It composes of state, state transition probability, and the initial probability distribution. The first-order Markov model assumes that the current state depends only on the last state and is irrelevant to the earlier states.
- *k-NN* [41] is a commonly used supervised learning method. Given the target sample, it finds out the nearest k samples, and predicts the information according to these neighborhoods. The distant calculation methods include Euclidean distance, Manhattan distance, etc.

- *SVM* [42] is a flexible supervised algorithm for both classification and regression. In order to solve the complex classification problems, SVM uses a kernel function to map the features of low dimensional space to the higher one. Common kernel functions include linear kernel, polynomial kernel, gaussian kernel and Sigmoid kernel.
- *DT* [43] is composed of directed edges and nodes which include the root node, internal nodes and leaf nodes. The path from the root node to each leaf node corresponds to a decision sequence. The commonly used algorithms for DT are ID3, C4.5 and CART.
- *RF* [44]is a collection of traditional DTs. Given the shortcoming that DT is easy to overfit, RF adopts the voting mechanism based on multiple DTs for improvement. To be specific, each tree in RF is trained by a sub dataset sampled from the original dataset and then the final result is obtained through the voting of all DTs.
- *k-Means* [45] is a commonly used clustering algorithm. It assigns each object to the nearest cluster based on the distances from cluster centers, and then recalculates the centers. This process is repeated until convergence and finally all objects are divided into k clusters.

C. Deep Learning Methods

Traditional machine learning technologies are essentially shallow learning, which has some limitations in terms of accuracy for classification tasks. In particular, their generalization abilities are restricted when dealing with complex problems. Hence, deep learning is introduced to address these limitations. We mainly introduce three typical deep learning methods that are widely employed in encrypted traffic analysis.

- *Convolutional Neural Network (CNN)* [46] is a kind of neural network with convolution operation added, which can weaken local features and enhance generalization. The special thing about CNN is that this method has the convolutional layer, which uses the convolution operation to filter the input features. Normally, the convolutional layer and the pooling layer appear alternately. After the combination of several sets of convolutional and pooling layers, a fully connected layer will be added at the end.
- *Graph Neural Network (GNN)* [47] is a deep learning method that processes the data represented in the graph domain [48]. As an effective graph analysis method, GNN is widely used in social network and physics system analysis, which leverage graphs as the denotation of the relationships underlying data.
- *Long Short-Term Memory (LSTM)* [49] is a special kind of Recurrent Neural Network (RNN) used to process sequences, which has achieved excellent performances in the field of natural language processing. LSTM has the ability to selectively remember the previous output results and superimpose them on different positions of the current cell to achieve the purpose of data analysis.

IV. A General Framework of Encrypted Traffic Analysis

In this section, we summarize the general framework of encrypted traffic analysis. We present the framework overview and then elaborate on each component in the subsection.

The overview of the general framework of encrypted traffic analysis is illustrated in Fig. 4, where four components are evolved, namely traffic collection, traffic representation, traffic analysis method, and performance evaluation.

A. Traffic Encryption Mechanisms

We summarize the encryption protocols and anonymity mechanisms currently used for secure communication before introducing each individual component in Fig. 4. Without traffic encryption, sensitive information transmitted in cleartext would be effortlessly sniffed by malicious eavesdroppers who have access to network interfaces. Moreover, cleartext communication provides convenience for intentional attackers inserting false information, which makes communication unreliable. Therefore, traffic encryption mechanisms have been implemented for communication security guarantees.

The interaction process of an encryption protocol usually includes two parts: a secure connection establishment phase and an encrypted data transmission phase. The secure connection establishment phase mainly includes handshake, authentication and encryption algorithm negotiation, which is the preparation for encrypted data transmission. The encrypted data transmission phase ensures safe transmission with the use of the encryption algorithm negotiated in the previous phase.

Next, we introduce mainly used encryption protocols including Internet Protocol Security (IPSec) on the network layer, SSL/TLS and Quick UDP Internet Connections (QUIC) between the transport layer and application layer, Secure Shell (SSH) and HTTP over Secure Socket Layer (HTTPS) on the application layer as well as anonymity mechanisms such as The Onion Router (Tor) in detail. The encryption protocols and anonymity mechanisms are organized according to their positions in the network structure as Fig. 5.

- *IPSec* [50].IPSec is a set of network transmission protocols for network layer communication security, which is utilized for the authentication of communication parties as well as the confidentiality and integrity of the data [51]. IPSec mainly includes Security Association (SA),Internet Key Exchange (IKE), Authentication header (AH) and Encapsulated Security Payload (ESP).
- *SSL/TLS* [1]. SSL locates between a reliable connection-oriented network layer protocol and an application layer protocol. It creates an encryption channel between the communication parties to ensure the confidentiality of communication, preventing malicious attackers acquiring and modifying any information, which may contain sensitive data. TLS is a standard proposed by Internet Engineering Task Force (IETF) in 1999 as the successor of SSL, providing transport layer security directly on top of the TCP protocol [52]. The recent version of TLS is TLS 1.3 [53] defined in 2018.

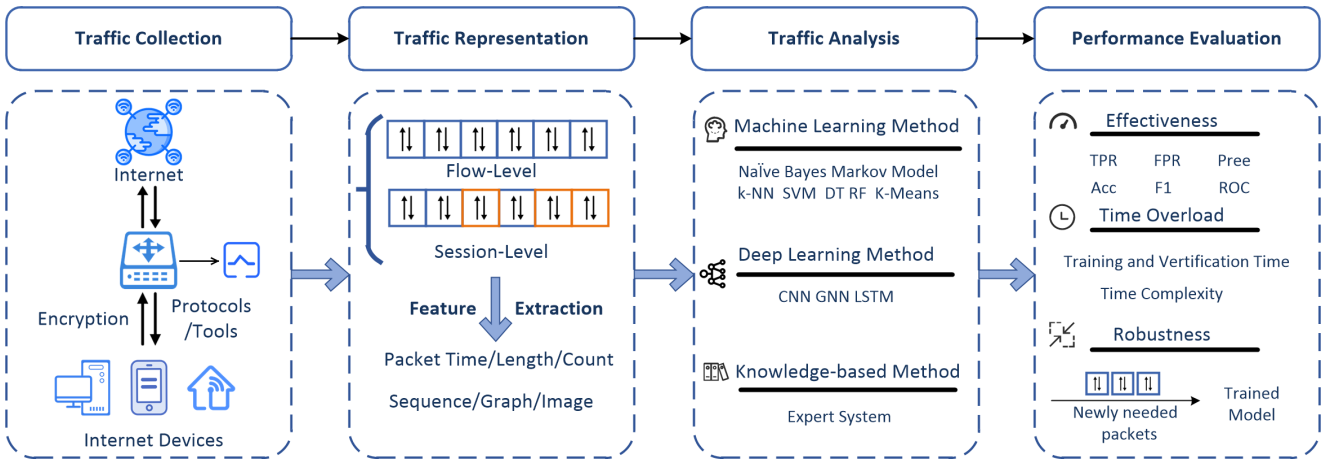


Fig. 4: A general workflow of encrypted traffic analysis

TLS consists of three components including handshake protocol, alert protocol and record protocol. Handshake protocol is mainly used for encryption algorithm negotiation, key generation and identity authentication. Alert protocol is exploited for warning message notification once one of the parties discovers the exception. Record protocol is responsible for data translation between communication parties. The data is divided into multiple shorter fragments and each fragment is compressed separately. Thereafter, the compressed fragment will be added with a message authentication code which is used to ensure the integrity and perform data authentication. Finally, the compressed fragment with the message authentication code will be encrypted together with a symmetric password.

- *SSH* [54]. SSH is a security protocol based on the application layer, which is designed to provide security for the connection and interaction between the client and remote host, preventing information leakage during remote access. SSH establishes an encrypted channel between the communicating parties to ensure the confidentiality of the transmitted information and uses a key exchange algorithm to ensure the security of the key itself. It mainly consists of three parts including the transport layer protocol (SSH-TRANS), user authentication protocol (SSH-USERAUTH) and the connection protocol (SSH-CONNECT).
- *HTTPS* [55]. HTTPS is the extension of HTTP by wrapping it inside the SSL/TLS protocol, which is used for safety communication over the network. Compared with HTTP, HTTPS can be utilized for encrypted transmission, identity authentication and data integrity protection. For this reason, HTTPS plays an important role in online activities such as shopping, payment and website browsing.
- *QUIC* [56]. QUIC is a new generation of low-latency network transport layer protocol based on user datagram protocol (UDP), which is similar to TLS implemented on TCP. HTTP and several extended protocols are based on TCP protocol. However, the TCP protocol requires three times of handshakes before establishing a connection. The purpose of QUIC is to reduce network latency while ensuring reliability. QUIC needs to complete the confirmation of packet transmission by itself because it is built on UDP which is unreliable. Besides, QUIC realizes multiplexing without head-of-line blocking and can be used for applications or services with real-time requirements.
- *Tor* [57]. Tor is an open-source and free anonymous communication software. It is widely used because it can provide low-latency and low-overhead anonymous access services. Due to the similarity between the structure of the Tor network and onion, it is called the onion routing. Tor can help users deal with some malicious attacks effectively. As the Tor client starts, it will first run an onion agent locally and then get in touch with the directory server which stores global relay node information. After the onion agent obtains the relay node information, it will select three nodes according to the routing algorithm to form an anonymous communication link. The onion agent will shake hands with each node using the Diffie-Hellman (DH) handshake protocol to negotiate a session key. Only when the link is confirmed to be established, users' access information begins to be delivered. In this process, each node cannot discover the source and destination of the information. Tor randomly selects forwarding nodes and keeps updating. At the same time, it encrypts the packet contents with the negotiated keys to ensure confidentiality.
- *Nested encryption* [58]. National Security Agency (NSA) delivers the multi-site connectivity capability package to meet the demands of data transmission across networks with different security levels. It needs two independent encryption tunnels (i.e., inner encryption component and outer encryption component), which can use either IPSec or Media Access Control Security (MACsec). According to the times of encryption, the network is classified into the red network that consists of unencrypted data, the gray network that consists of data that has been encrypted

once, and the black network that consists of data that has been encrypted twice. When a packet is sent to the black network, it is encrypted by the two encryption components in turn. Similarly, at the destination, the received packet is unencrypted twice.

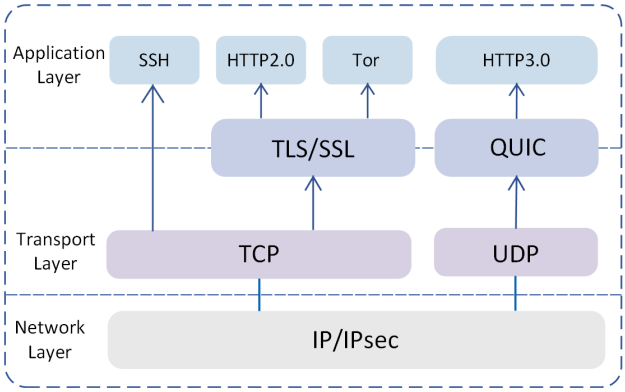


Fig. 5: Mechanisms in the network model

Encryption technologies are used to protect user privacy and communication security emerges endlessly. Although these encryption protocols or anonymity mechanisms protect the security of communication to a certain extent, it is still possible for the attacker to obtain some sensitive information by analyzing the encrypted traffic.

B. Traffic Collection

Traffic data collection is an important step in the field of encrypted traffic analysis, as a data set is the foundation of experiments. Therefore, data collection often acts as the starting point of encrypted traffic analysis.

As shown in Fig. 2, traffic can be captured at various *nodes* in a network, such as switches, routers, and gateways. In recent years, several network simulation techniques, such as NS3 [59], have been proposed to simulate network traffic. Software-Defined Network (SDN) [60, 61], which has a special network structure, can also facilitate traffic data collection. In this subsection, we are dedicated to introducing typical traffic collection tools that are commonly used for traffic capture.

- *Libpcap* [62]. It is a well-known network packet capture library based on C/C++, which sits at the data link layer and can display TCP/IP and other packets transmitted over the network. Libpcap defines *pcap* as the storage form of captured packets, which is widely used in traffic capture tools. Many other traffic capture tools, such as Tcpdump and Wireshark, are based on this library.
- *TCPdump* [63]. It is a packet analysis tool that runs on Linux systems, which is based on Libpcap. TCPdump offers the ability to capture network packets and record those packets in *pcap* files. It also enables users to filter network packets using regular expressions. Since TCPdump is based on the command line, users can set parameters to meet different requirements, such as limiting the number of captured packages.
- *Wireshark* [64]. It is a traffic collection tool that runs on Windows, which is known as the user interface version of tshark [65]. Similar to TCPdump, it enables user specific requirements on packet capture. It can display traffic information more visually through the user interface, such as displaying different protocol information in different colors. Compared with TCPdump, Wireshark shows a better user experience but consumes more power and memory [63].
- *Netflow* [66]. It is a network monitoring technique proposed by Cisco, which can be deployed on routers and gateways. Unlike the above three tools, Netflow provides a flow-level view of network traffic, recording the information on each flow, rather than providing the packet-level information as Wireshark and TCPdump do. In NetFlow, a flow is uniquely defined by the five tuples, namely the source IP address or port, the destination IP address or port, and the transport protocol. For enterprise users, a flow-level display of traffic can be more useful, manageable and readable than the packet-level display.
- *Hardware Probe* [67]. It is a network tool that can capture and analyze network packets at the physical layer. The hardware probe can provide more timely information, including physical layer data, without consuming network resources. While compared with software-based tools, the strategy based on hardware is expensive and inflexible.

C. Traffic Representations

The representation of traffic is essential for traffic analysis. For different application scenarios, the traffic representations may be diverse. A superior traffic representation can improve the effectiveness and reduce the overhead.

1) Representation Level: We introduce two mainly used traffic representation levels according to classification granularity including flow-level and session-level.

- *Flow-Level.* The flow is a collection of data packets with the specific common attributes. Generally, common attributes refer to the source or destination IP, source or destination port and protocol [68]. Moreover, in [69], a group of packets with specific common attributes collected in a pre-defined time interval are regarded as the basic unit of analysis, which is also considered as flow-level representation in this survey. From the flow-level representation, multiple statistical features such as an average packet count, maximum packet length, or minimum interval time can be extracted.
- *Session-Level.* The session is a collection of packets generated during the complete interaction between the client and server, e.g., packets generated by a complete website visit process from session-level traffic [70]. The packets in a session may have different attributes such as destination IP, therefore, a session may contain several flows. Especially, there is no limitation on the size of the flow, e.g., a flow may contain only one packet. However, a session must include a completely interactive process, so it includes at least two packets related to the establishment of the interactive process. Meanwhile, the statistical

features extracted from flow-level representation are also suitable for session-level representation as the essence of a session is still a collection of packets. In addition, other features such as the count of flows or the average duration of flows can be extracted as session-level features.

2) The Forms of Representations: The traffic representation corresponds to the input of classifiers. To be specific, multiple features such as packet-based features and statistical features can be used as the input of traditional machine learning classifiers. Besides, raw representations of traffic such as sequences [70] or graphs [71] may be used as the input of deep learning models. Different forms of traffic representations for encrypted traffic analysis are introduced as follows.

- *Packet-based Features.* The basic packet-based features are the information of packet header captured directly. Generally, the five-tuple of IP packets, including source and destination IP addresses, source and destination ports and protocol types, are the most commonly leveraged packet-based features. Moreover, the Time-To-Live of the IP header, the initial window size of the TCP header can also be extracted as features. These are the simplest features obtained directly from the packet.
- *Statistical Features.* Statistical features are defined as features extracted from a group of packets through abstraction or computation. The statistic values include expectations, deviations, average, minimum, maximum, medians, accumulation, etc. Combining these statistics with traffic attributes, such as packet size, inter-arrival time, packet count, results in a large number of statistical features, e.g., average of cumulative packet sizes, quartile of inter-arrival time and burst packet count.
- *Raw Traffic Representation.* In addition to selecting network traffic features manually, researchers can also use frameworks or models like deep learning algorithms to extract features automatically. Artificial feature extraction is based on experience, however, the raw representation transfers the offload of feature selection to the models. Several studies encode traffic into sequences [70], graphs [71] or images [72] for classification. For example, the traffic can be abstracted into graphs. The structure of the graph can be regarded as a set of interconnected nodes, each of which is treated as a packet. This is similar to the graph definition in data structure and can be augmented with directions, weights, etc.

D. Encrypted Traffic Analysis Methods

With the traffic representation obtained in Section IV-C, traffic analysis methods are applied according to different task requirements. The investigated encrypted traffic analysis methods are summarized in Table III. As a necessary supplementary, apart from machine learning methods, we also introduce knowledge-based methods, which are selected in certain scenarios.

1) Machine learning method: As shown in Table III, we can divide these methods into two categories, namely *traditional machine learning* and *deep learning*. In encrypted traffic analysis, the major difference between the two categories is that the former requires manually-crafted features as input, which needs complex feature engineering processes, while the latter enables end-to-end traffic classification. The process of feature extraction requires much prior knowledge about the task, e.g, the most contributing features for distinguishing target traffic from background traffic.

Traditional machine learning technologies are essentially shallow learning, which has some limitations in the accuracies for classification tasks. In particular, their generalization abilities are restricted when dealing with complex problems. In contrast, deep learning methods are introduced to address these limitations. Benefitting from the feature learning ability of deep learning, it enables raw traffic representation as input. For example, several studies represent traffic with sequences, e.g., packet direction sequences [70], and use a CNN as the classifier. Also, the CNN can be combined with a triplet network as the feature extractor [142]. Moreover, network traffic can also be represented by a graph structure, which digs out the hidden information [71], and is combined with GNNs for traffic classification. In addition, as there are correlations between the packets at the current moment and those in the previous period of time, LSTMs can be used to capture timing features of packet sequences [149].

The rapid development of machine learning, especially deep learning, provides a large number of analytical methods for encrypted traffic analysis. Unsupervised learning has even the ability to identify unknown targets, which provides more possibilities for abnormal traffic detection.

2) Knowledge-based method: It is used for traffic analysis based on prior knowledge. The knowledge-based method is generally composed of two parts, i.e., the knowledge base and the inference engine. The knowledge base is used to store facts, such as specific patterns and logical assertions. The inference engine is used to analyze the problem and match the content in the knowledge base. In encrypted traffic analysis, several studies extract features of traffic and match them using similarity analysis [73] or pre-established rules [74] to achieve the purpose of classification. However, the accuracy of knowledge-based detection greatly depends on the completeness of the knowledge base. Therefore, knowledge-based methods are not widely used in traffic analysis. With the types of network devices, applications and attack methods increasing, the knowledge base needs to be updated dynamically. Hence, most studies tend to leverage machine learning methods for encrypted traffic analysis.

E. Performance Evaluation Metrics

In order to verify the effectiveness of encrypted traffic analysis methods, several effective validation methods and evaluation metrics are proposed.

The validation method is used to confirm that the analytical method is suitable for its intended use. In machine learning, there are two commonly used datasets, training set and test set, which are utilized for model building and model validating.

Different partition methods are used for dividing the two datasets. Common partition methods are k-fold cross validation, hold-out, and bootstrapping. Among them, the k-

This article has been accepted for publication in IEEE Communications Surveys & Tutorials. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/COMST.2022.3208196

TABLE III: Summary of existing traffic analysis methods

	Classifer	Related works	Applications	Advantages	Disadvantages
Knowledge-based	-	[73], [74], [75], [76], [77], [78], [79], [80], [81], [82], [83], [84]	1. It classifies traffic through stacking simple rules. 2. Deal with tasks where the feature of target traffic is obvious, such as OS identification (Section V-B).	Traffic can be identified by matching specific features, which has high interpretability.	Depend on the effectiveness of the constructed knowledge library.
Supervised ML	Bayes classifier	[85], [86], [87], [88], [89], [90], [86], [91]	1. It is applied in scenarios where historical traffic labels are known. 2. Deal with classification tasks where traffic can be obtained ahead of time, such as IoT device fingerprinting (Section V-A).	1. Through learning from historical traffic datasets, it provides reliable and repeatable decisions. 2. Utilize highly interpretable algorithms, which makes it easy to explain the reasons for decision. 3. Take relatively little time for model training, usually only seconds to hours [38].	1. Require artificial feature extraction based on prior knowledge such as discrimination of different categories. 2. It is shallow learning, which performs pool in feature extraction. 3. It is less capable of dealing with time drift, especially when considering a long period, as the applications, malware, may be updated [94].
	Markov model	[16], [92], [93]			
	K-NN	[95], [96], [97], [98], [99], [100], [101], [102], [103]			
	SVM	[104], [68], [105], [106], [107], [108], [109], [110], [22]			
	DT	[111], [112], [113], [114], [115], [116], [117]			
	RF	[118], [119], [120], [121], [122], [123], [124], [125], [126], [127], [128], [129], [130], [131], [132], [133], [134], [135], [136]			
Unsupervised ML	K-Means	[137], [138]	1. It is applied for traffic samples that have no historical labels. 2. Deal with classification of unknown traffic categories, such as unknown malware detection (Section VIII).		
DL	CNN	[139], [140], [141], [70], [142], [143], [144], [145], [146], [147]	1. It nests a cascade of hidden layers with nonlinear processing unitss for feature extraction. 2. Deal with scenarios where there is few prior knowledge, as it allows to be fed with raw representation, such as the packet direction vector (Section VII).	1. The multilayer architecture of deep learning helps map the traffic to higher level representation [148]. 2. Perform well in scenarios where there are large and high dimensional data. 3. Enable end-to-end traffic analysis, as few complex feature engineering is required [145].	1. Require significant volumes of data that should be updated regularly. 2. Demand generous computing resources, which is expensive [38]. [148]. 3. Deep network is a black box, which makes it difficult to interpret as the complex hyperparameters and network structures
	GNN	[71]			
	LSTM	[149], [150], [151], [152]			

fold cross validation is the most frequently used validation method. In practice, 10-fold (i.e. $k = 10$) cross validation it is commonly used to evaluate to divide the data into 10 blocks in encrypted traffic analysis methods validation [86, 153, 154], which means $k = 10$. We first10-fold cross validation divides a complete dataset into an average of 10 mutually exclusive subsets. Then, one of the subsets is used as the test set in turns, and the union of the remaining 9 subsets is used as the training set. In the end, each subsetblock of data is used for training and testing, which maximizes the use of the dataset and the final validation result does not depend on the random choice of the training set.

At the same time, according to the coverage of the training set to the test set, we can divide the validation method into *closed-world* validation and *open-world* validation. In a *closed-world*, we assume that the training set contains all classes of samples in the test set, which means there is no unknown sample in the test set. However, the *open-world* corresponds to a more realistic scenario, in which the test set contains samples belonging to unknown classes. In the field of encrypted traffic analysis, unknown samples can represent new network devices [121], OS types [155], malware [154], network attacks [136], etc. Compared with the *closed-world* validation, the *open-world* validation generally has higher requirements for classification models.

After the selection of validation methods, multiple eval-

TABLE IV: Confusion matrix of classification result

Predicted class	Actual class	
	Condition positive	Condition negative
Condition positive	TP	FP
Condition negative	FN	TN

uation metrics are proposed. According to the different requirements of the classification on the accuracy, time and generalization, we can divide the evaluation metrics into effectiveness, time overhead and robustness.

1) Effectiveness: For both binary classification and multi-classification tasks, two metrics, $accuracy$ and $error\ rate$, are commonly used. $Accuracy$ is defined as the ratio of the number of samples labeled correctly over the total number of samples, while $Error\ rate$ is defined as the percentage of samples classified incorrectly in the total samples.

In particular, for binary classification, we can obtain a confusion matrix, as shown in Table IV. The confusion matrix includes four elements: true positive (TP), false positive (FP), true negative (TN), and false negative (FN). Among them, TP stands for positive samples predicted to be positive by the model, FP stands for negative samples predicted to be positive by the model, FN stands for positive samples predicted to be negative by the model, and TN stands for negative samples predicted to be negative by the model. Based on these four elements, the commonly-used evaluation metrics for effectiveness can be calculated.

- *TPR (Recall)* is the ratio between the number of positive samples correctly classified and the number of actual positive samples, ranging between 0 and 1.
- *FPR* is the ratio between the number of negative samples incorrectly classified and the number of actual negative samples, ranging between 0 and 1.
- *Precision (Prec)* is the ratio between the number of positive samples correctly classified and the number of samples classified as positive, ranging between 0 and 1.

Moreover, some metrics that consider two of the above metrics comprehensively are proposed.

- *F1* considers both Prec and Recall, which is represented as the harmonic mean of Prec and Recall.
- *Receiver Operating Characteristic (ROC)* curve is constituted by TPR and FPR, with TPR as the vertical axis and FPR as the horizontal axis. When we compare the performance of two models, if the ROC curve of the first model can completely 'cover' the second one's, we can consider the performance of the first one is better.
- *Area Under ROC Curve (AUC)* is the area under the ROC curve. If the ROC curves of the two comparison models cross, we can choose AUC as the evaluation metric.
- *Precision-Recall Curve (P-R Curve)* is constituted by Prec and Recall, with Prec as the vertical axis and Recall as the horizontal axis. When it comes to the imbalance dataset, P-R Curve is used to evaluate the performance of the model as an alternative to ROC Curve[70].

In addition, customized metrics are introduced into encrypted traffic analysis to evaluate the amount of leaked information [102]. Mutual information, which measures the mutual dependence between two variables, can be used as the feature engineering process for WF. The mutual information is defined as Formula 1:

$$I(F;W) = H(W) - H(W|F) \tag{1}$$

where F denotes the fingerprints of traffic, W denotes the website information, $I(F;W)$ means the amount of information can be attained from F about W, $H(\cdot)$ is the the entropy.

2) Time Overhead: Several encrypted traffic application scenarios have time requirements for methods, such as network attack detection and network device identification. For these scenarios, time overhead is another issue affecting the practicability of the model. In the encrypted traffic analysis field, time overhead evaluation metrics mainly include training time, validation time and time complexity.

- *Training and validation time*. They are important indicators commonly used for experimentally evaluating time overheads. Training time refers to the time required to train a traffic analysis model with a given set of training samples, while validation time refers to the time used to analyze the test set using the pre-trained model. To make a fair comparison among different analysis methods, the evaluation on training and validation time should be conducted on the same dataset [156–158].
- *Time complexity*. It provides a theoretical estimate of time overhead to run a certain traffic analysis method. Different from training and validation time, time complexity focuses on computational complexity, rather than concrete analysis time spent on specific datasets.

3) Generalization: As the traffic in the network changes dynamically over time, the previously constructed model may not be suitable after the traffic features change. It requires that the constructed model is capable of generalization, which means that the classifier is supposed to be robust to the data mismatch problem. It may occur due to the staleness of training data or structure difference between training and test set. Here, we introduce a metric to evaluate the generalization which is defined as the number of samples needed for retraining the previously trained model [142]. For example, in the field of website fingerprinting, it is impossible for an attacker to use all website traffic for model training. If the training set comes from website a, while the attacker is expected to monitor website b, there are different structures between training and test sets. Due to this issue, the previously trained model is no longer applicable to the current test set. Therefore, we are supposed to use new samples which have the same structures as the test set to retrain the model. If the model retrained by only a few samples performs well, it is considered to have strong generalization and transferability.

TABLE V: A Conclusion For Studies Related To Network Asset Identification.

	Analysis		Dataset		Feature[1]								Method[2]				Evaluation[3]			
	Ref	Year	Platform	Species	Level	Style	PH	PC	PL	PT	PD	Others	Category	Classifier	Train	Predict	OW	CW	ET	TO
Device Fingerprinting	[73]	2010	Access point	6	Flow	SF				✓			KB	-	Off	Off		✓	✓	
	[85]	2018	Mobile/IoT	34	Session	SF	✓						ST	Bayes	On	On	✓		✓	
	[104]	2017	Mobile/IoT	22	-	SF						Link-layer	ML	RF/CART/SVM	Off	Off	✓	✓	✓	
	[159]	2018	IoT	14	Session	SF	✓		✓			Entropy	ML	Gradient boosting	Off	Off		✓	✓	
	[95]	2018	IoT	22	Flow	SF			✓	✓			ML	k-NN	Off	Off		✓	✓	
	[118]	2017	IoT	21	Flow	SF	✓		✓	✓			ML	RF	Off	Off		✓	✓	
	[156]	2019	IoT	21	Flow	SF	✓						ML	AdaBoost	Off	Off		✓	✓	
	[68]	2017	IoT	9	Flow	SF		✓	✓	✓			ML	SVM	Off	On		✓	✓	
	[119]	2017	IoT	27	Flow	SF	✓						ML	RF	Off	On			✓	✓
	[120]	2019	IoT	4	Flow	SF			✓	✓	✓		ML/DL	RF/ANN	Off	On		✓	✓	
	[160]	2018	Mobile	2	Flow	SQ				✓			DL	CNN	Off	Off		✓	✓	
	[121]	2020	IoT	10	Flow	SF	✓		✓				ML/DL	RF/Autoencoder	Off	Off	✓		✓	
	[161]	2015	Mobile/IoT	37	Flow	SF				✓			DL	ANN	Off	Off	✓		✓	✓
OS Identification	[74]	2016	Computer	-	Session	SF	✓						KB	-	Off	Off		✓		
	[75]	2016	Mobile	4	Flow	SF						Spectrum	KB	-	Off	On		✓	✓	✓
	[86]	2014	Mobile	2	Flow	SF			✓		✓		ST	Bayes	Off	Off		✓	✓	
	[87]	2014	Computer/Mobile	3	Flow	SF	✓			✓			ST	Bayes	Off	Off		✓	✓	
	[111]	2014	Computer/Mobile	4	Flow	SF	✓						ML	DT	Off	Off		✓	✓	
	[122]	2017	Computer/Mobile	4	Session	SF	✓						ML	RF	Off	Off		✓	✓	
	[153]	2019	Computer/Mobile	5	Session	SF	✓	✓	✓				ML	Gradient boosting	Off	Off		✓	✓	
	[105]	2017	Computer	-	Session	SF	✓	✓	✓	✓	✓	Burst	ML	SVM	Off	Off		✓	✓	

[1] In the *Traffic Representation* columns, *Level* is traffic representation level, *Style* is traffic representation style, where SF denotes statitical features and SQ denotes sequence, *PH* is packet header, *PC* is packet content, *PL* is packet length, *PT* is packet timing, *PD* is packet direction.

[2] In the *Method* columns, *Train* is online training or offline training, *Predict* is online predicting or offline predicting.

[3] In the *Evaluation* columns, *OW* is open world, *CW* is closed world, *ET* is effectiveness, *TO* is time overhead.

V. Network Asset Identification

Network assets refer to valuable resources in computer networks, which vary from physical network equipment (e.g., servers, routers, switches, firewalls, and printers) to software assets such as operating system (OS). Currently, there are several search engines (e.g., Shodan [162], FOFA [163] and Zoomeye [164]) that provide information retrieval services of worldwide network assets connected to the Internet.

Traffic analysis methods are largely used in network asset probing process. On one hand, network administrators can keep themselves updated with the online status of their network assets, and can also find assets with potential security risks in time. On the other hand, malicious attackers can also accurately grasp vulnerabilities of devices (e.g., probing devices with stale protections) and carry out targeted attacks.

Traffic encryption has a significant impact on network asset identification, as traditional analysis methods on unencrypted traffic depend on inspecting certain fields in packet payload. For instance, the device type can be toilless attained from the non-encrypted traffic (e.g., the names and version numbers of standard libraries used by devices can be extracted at the HTTP packet header [165]), and the OS type is also clearly shown in the HTTP packet header (e.g., the user-agent field). All the information used above has been masked in encrypted traffic. Therefore, in the traffic encrypted scenario, it is non-trivial to identify network assets accurately without getting access to packet content. In this section, we will introduce the applications of encrypted traffic analysis in network asset identification from device fingerprinting to OS identification, which are summarized in Table V.

A. Device Fingerprinting

With the aim of network security, when a device gets connected to a network, the network administrator is supposed to authenticate this device and allocate it basic communication permissions according to the device type. Due to the numerous quantity and frequent alternation of network devices, authentication manually is unpractical, which makes it necessary to identify the device type automatically. However, the abnormal behaviors of several untrusted devices challenge the identification. For example, malicious devices may respond to queries with fake identities [159], or deviate from their expected behaviors, e.g., a temperature sensor tries to read information from a network camera. These problems motivate researchers to find effective and robust fingerprints that can be used to identify device type accurately.

Traditional machine learning methods. Machine learning methods are widely used in device probing, performing well in terms of device classification. Numerous studies compare the

effectiveness and overhead of different algorithms for device fingerprinting, such as SVM, random forest and decision tree. Most of the studies investigated focus on IoT devices, as the amount and types of IoT devices keep on increasing. Hence, we will mainly introduce the application of machine learning methods in IoT device identification.

The growing smart home IoT market introduces new challenges for network management. Although IoT devices bring us convenience, they become potential targets for adversaries, such as smart watches, network cameras and even sweeping robots. Notably, attacks involving IoT devices account for more than 25% of attack events in enterprises by 2020 [166]. Most IoT devices generate less traffic and are lower diversity, which makes it difficult for traffic-based identification [119].

To construct effective device fingerprints using transitional machine learning models, traffic representation is of great importance. Several studies try to make use of protocol header information, such as the link layer, network layer and application layer, for traffic representation.

Maiti *et al.* [104] make an observation that link layer features can be used for traffic representation. A set of consecutive link layer frames are combined together as a block, which is the basic unit for fingerprint construction. Traffic features can be extracted from blocks, including temporal properties, payload sizes and header information, and then fed to a random forest classifier. While feature extraction requires a block size of 30K frames, which may take up to days of traffic collection for a standby IoT device. It indicates that only link layer features are insufficient for efficient device fingerprinting.

To improve the efficiency on feature extraction, statistical features are explored in the following studies [95, 118]. Sivanathan *et al.* [118] propose a method to distinguish IoT traffic from non-IoT traffic and then identify specific IoT device types. Firstly, the number of different Internet servers contacted and unique DNS requests can be used to distinguish whether it is an IoT device. Further, they observe that 12 features (e.g., DNS interval, sleep time) are effective for distinguishing one IoT device from another. The extracted features are fed into a RF classifier, which has the viability of identifying IoT devices. However, this method splits the traffic by fixed-length windows, i.e., regard traffic in a fixed period of time as a sample, which may lead to the traffic generated by an activity to be split up, thus hindering classification results.

Msadek *et al.* [156] overcomes the shortcoming of manual parameter selection in earlier studies [118], by adding sliding windows for traffic splitting automatically. The basic idea is that, instead of shifting by a specific length, it moves traffic with the next unrelated activity, which ensures the relevant traffic in the same window. To construct device fingerprints, they combine the basic features based on dominant protocol analysis (e.g., types of protocols, port intervals) with derived features based on statistical distributions (e.g., maximum and minimum packet sizes). The feature fusion process can obtain more discriminative information for device fingerprints.

The methods described above have a common limitation that they cannot extract features until the completion of traffic transmission and thereby are not suitable for early identification. In practice, however, network administrators wish to know device types as soon as the devices join the network. To meet the needs of real-time device identification, several studies propose more time efficient methods, e.g., obtaining traffic representations using the first n packets in a connection.

IoT sentinel [119] extracts 23 features (e.g., IP options) from the protocol header from packets generated in the device setup process to meet the demands. For each newly joined device, the features of the first 12 packets are recorded by gateway to form device fingerprints. Based on the fingerprints provided by the gateway, the remote server identifies the type using RF and returns an isolation level, which is leveraged for restricting the communication behavior of the device. The extracted features are all related to specific fields in packet header, without taking into consideration the behavior features of devices.

Deep learning methods. We notice that transitional machine learning methods have two significant defects: 1) they all rely on artificial feature selection, which requires prior domain knowledge, and 2) they are tightly bounded with a specific task and cannot be easily transferred to different tasks. To overcome these limitations, deep learning techniques have been employed in recent studies [121, 160, 161].

Shanthi *et al.* [161] propose a mobile device classification method, which employs packet inter-arrival time as traffic representation and build a classifier using Artificial Neural Networks (ANNs). They test 14 devices in an isolated testbed and 23 devices (e.g., iPads, iPhones) in campus network. As is known that packet timing is largely affected by network fluctuation, such as packet loss and network jitters, these methods are difficult to maintain high-level performance in terms of accuracy. Similar to the previous work, packet inter-arrival time is also exploited by Aneja *et al.* [160]. To achieve better performance, they reshape the sequences of packet inter-arrival into a two-dimensional image, which is used to train a CNN classifier. Thus, the device recognition problem has turned into an image classification problem. However, they only consider two devices in their experiments, which is far less than the amount of devices in practice. Moreover, there is an unsolved issue of model retraining, i.e., the single multi-class classifier requires full model retraining when new types join, which requires further investigation.

Knowledge-based methods. These methods can be leveraged in routing device probing. Different Access Point (AP) types have specific internal architectures, which affects the speed of packet processing. Hence, the packet inter-arrival time is a distinguishing feature for routing devices. Gao *et al.* [73] extract distinct patterns of traffic by applying wavelet transform to the sequence of packet inter-arrival time. The wavelet analysis is used to reveal discriminative attributes of wireless devices. Through measuring similarities of the current pattern and the patterns in database, they can determine which type of AP the traffic belongs to. This method is restricted in real applications, because it largely relies on prior knowledge of specific architectures.

B. Operating System Identification

Operating System (OS) identification can find its applications in various scenarios. It enables network administrators to obtain the information of end devices connected to the network and ensure timely OS upgrades. From the perspective of malicious attackers, OS identification is leveraged as a powerful tool to learn vulnerabilities of the target system or tailor attacks on a certain stale OS. Therefore, the study on accurate and rapid OS identification is of great significance to network security.

Machine learning methods. In recent years, a large number of machine learning methods have been proposed for OS recognition. Multiple protocol header features are extracted, such as the Time-to-Live of the IP header, the initial window size and max segment size of the TCP header [122], as TCP/IP header can still be attained even if the traffic is encrypted. For example, p0f [167], a passive OS fingerprinting tool, relies on configuration differences of various network stack implementations reflecting in the packet header.

Al-Shehari *et al.* [111] extend p0f feature set by adding the features extracted from the header of FIN packets (FIN packets are TCP packets required to close connections). They utilize the C4.5 algorithm to identify the OS that does not have an exact match in the p0f signature database. However, only the older versions of OS are considered in feature selection, which leads to the newer OS can not be classified at the version level.

To classify the OS more accurately based on encrypted traffic, several statistical features are applied, such as mean and extreme value. In addition to TCP/IP and TLS header features, Fan *et al.* [153] extract several flow statistics features (e.g., mean packet throughput). They use light gradient boosting machine algorithm to identify the host OS information. This method can identify not only types but also OS versions. Muehlstein *et al.* [105] propose a method to identify the OS, browser, and application. They leverage both base features which are used in most traffic classification methods (e.g., packet inter-arrival time) and new features which are based on a comprehensive network traffic analysis (e.g., bursty behaviors). About 50 features are exploited in this work including both header features and statistical features. The feature selection is crucial, which affects the accuracy of the model directly. However, the statistical features are sensitive to OS upgrade, even a small magnitude of changes may reduce the effectiveness of the model, which makes robust feature extraction an important research direction.

Knowledge-based methods. These methods mainly leverage fingerprint libraries and pre-defined rules to identify the OS types [74]. However, this method will fail if there are multiple OS types corresponding to one key. To find more effective fingerprints, Ruffing *et al.* [75] present a method that extracts features of traffic in the frequency domain. To filter out the frequency component that is not helpful for OS identification, the genetic algorithm is applied to decide which one should be kept. By calculating the correlation between the features of the target traffic and the samples in fingerprint libraries, they can identify which type of OS the traffic belongs to. While the classification of minor OS versions is not solved by the proposed method.

C. Summary and lessons learned

In this section, we review existing studies that focus on encrypted traffic analysis applied in network asset identification, including device fingerprinting and OS identification. Due to the differences in protocols used between devices, the packet header information (e.g., TCP/IP fingerprints) is one of the mostly used features extracted from traffic, as these features can be attained even if the traffic is encrypted. In addition, the statistical features (e.g., mean packet size) are widely used to represent the uniqueness of traffic generated from different devices, achieving effective identification performances.

Our investigation depicts several remaining challenges in network asset identification. First, network devices, especially IoT devices, are well known for the huge scale deployment. Thus, continuous traffic collection and processing are required, which lead to significant improvement for both feature extraction methods and classification frameworks. Second, the huge scale is reflected in not only the generated traffic volume, but also the multiple asset types. In practical applications, there are types of devices that the model has not known before, which requires the methods to be able to handle more complex cases in the open-world scene. Finally, traffic source is a crucial part of the effectiveness evaluation. However, due to the lack of valid public dataset, most of current studies resort to the traffic generated in the lab environment. Therefore, a large-scale and real-world dataset is essential for network asset identification.

VI. Network Characterization

As numerous network services keep on emerging, the improvement on user experience and service quality becomes crucial to service providers. In this paper, we refer to network characterization as the process of traffic monitoring and analysis to learn network characteristics that are closely related to service quality. More specifically, we focus on two aspects of network characteristics, namely QoE metric measurement and protocol recognition.

The rapid growth of video streaming traffic asserts significant pressure on mobile network operators [168]. QoE metric measurement aims to extract critical QoE indicators (e.g., video bitrate, resolution, and stalling) from network traffic, which can provide network operators with more insights into the video delivery quality perceived by end users. Network protocol recognition is also of great importance to network operators, as it can provide a deep understanding of all kinds of protocols in the traffic traversing their networks.

While traffic encryption benefits user privacy protection, it has a negative impact on monitoring network characteristics. In unencrypted traffic, QoE measurement and protocol recognition are usually based on high performance DPI engines that carefully inspects packet contents. For instance, the string related to specific requests can be extracted from the HTTP header to indicate the delay aroused at the beginning of video sessions [12]. However, this information is hidden by the encryption of packet contents. Moreover, several private

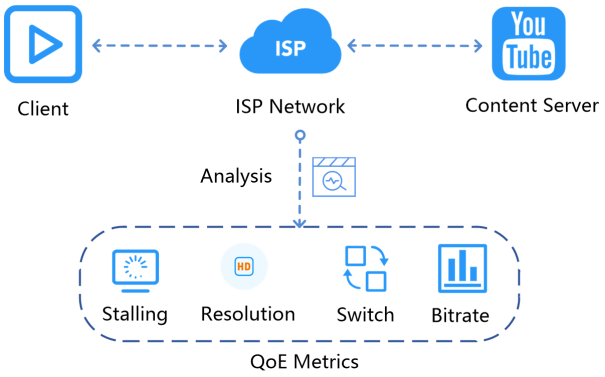


Fig. 6: QoE Metric Measurement Model

protocols mask their identities through encryption for evading censorship [169], increasing the difficulty of protocol recognition. In this section, we review and summarize existing studies on encrypted traffic analysis with the goals of QoE metric measurement and protocol recognition.

A. QoE Metric Measurement

With the exponential increase in multimedia services, QoE measurement is increasingly crucial to the optimization or development of corresponding services [172]. To measure QoE reliably and accurately, several researchers develop objective quality prediction models through traffic collection and analysis. The factors which influence quality (e.g., stalling event, resolution and bitrate) or overall QoE which takes all factors aforementioned into account are measurement metrics concerned by the current QoE-related studies. We list several metrics and show the model of QoE metric measurement in Fig. 6. Selection of metrics describing QoE, collection and prediction of traffic, especially real-time monitoring, are three key points in QoE metric measurement. The current studies of QoE metric measurement are summarized in Table VI.

Traditional machine learning methods. These methods utilize observable network features (e.g., packet length and packet time) and machine learning classifiers to predict QoE. Several studies choose different features and use multiple statistical models for accurate QoE metric measurement through analyzing encrypted traffic. The parameters and their weights of models are momentous for accurate prediction. Therefore, distinct weights are assigned to factors usually according to their contributions to the overall QoE to obtain an objective statistical model [170, 171]. The proliferation of smart vehicular terminals imposes serious challenges to vehicular networks [173]. Oche *et al.* [170] observe that the key factors impacting the QoE in a vehicular environment are frame rate, bit rate, packet loss, throughput and delay, which may lead to blocking, blurriness or blackouts of videos. They propose a QoE estimation function based on the multivariate statistical approach and ordinal regression analysis. The overall QoE is estimated as a weighted sum of the QoE influencing factors. Due to the convenience of predicting the overall QoE according to the evaluation function, it is suitable for real-time QoE prediction. However, the function requires regular parameter updates to meet the dynamic changes of services.

In addition to using the overall QoE aforementioned, several studies employ specific indicators (e.g., stall occurrence, resolution, and bitrate) to describe different levels of QoE. Dimopoulos *et al.* [123] propose a predictive model for measuring QoE degradation levels caused by three key factors, i.e. stalling, video resolution and quality variations. To obtain these factors from encrypted traffic, they extract 10 features from chunk size and packet inter-arrival time (e.g., the percentiles of chunk size), which are fed into RF for QoE prediction. While the approach extracts features after obtaining the traffic of an entire video session, which makes it unsuitable for real-time measurement.

Traffic collection is a crucial part of the classification process. Data diversity is important for creating robust and effective models. Specifically, in order to collect more realistic data, a variety of network conditions are supposed to be taken into consideration. To this end, Oršolić *et al.* [126] collect the YouTube video traffic under 39 different bandwidth scenarios to estimate the QoE of end users when watching videos. The bandwidth envelopes range from 0.3 to 7 Mbps to cover a large number of different scenarios, which differ in QoE metrics, e.g., stalling number. To be specific, they extract 17 statistical features corresponding to packet length, packet time, packet count and throughput as the input of 5 machine learning models, e.g., RF and Naïve Bayes. However, it is not clear whether the approach can obtain a superior performance when transferred to other platforms, as the features are extracted for YouTube traffic.

Similar to the aforementioned issues, the traffic generated by software running on different platforms or network environments may be various in the terms of element loading orders or packet sizes. In order to conduct a more complete study of video streaming, Pan *et al.* [125] collect their data from different networks, e.g., WiFi and 4G, or different areas, e.g., Hong Kong and Shanghai for evaluating the quality of video. Among them, the bitrate is regarded as the key factor influencing user experience. Thus the main goal is to acquire bitrate information. To be specific, multiple machine learning algorithms such as RF are used as classifiers in the method. However, in addition to quantifiable factors, more user subjective feelings should be considered to the scope of evaluation models, for instance, although high resolution is provided, it is not user-friendly if stalling occurs due to the poor network condition.

Deep learning methods. Traditional machine learning methods require the input of interpretable features, while deep learning can utilize raw traffic data to avoid hand-crafted feature selection. Shen *et al.* [139] design a CNN-based model for measuring fine-grained video QoE metrics, including startup delay, rebuffering, and video resolution. To achieve the goal of inferring QoE metrics in a real-time manner, only the Round-Trip Time (RTT) of upstream packets are used as a crucial feature. Real-world datasets collected from two large-scale content providers (i.e., YouTube and Bilibili) are utilized to demonstrate the effectiveness of the approach. While it will be invalid in video streaming based on UDP protocol, as there

TABLE VI: A Conclusion For Studies Related To QoE Metrics Measurement.

Analysis		Analysis Goals				Dataset		Representation[1]							Method[2]				Evaluation[3]	
Name	Year	Stalling	Resolution	Switch	BR	Scale	EM	Level	Style	PC	PL	PT	PD	Others	Category	Classifier	Train	Predict	ET	TO
[170]	2017		✓		✓	2 databases	HTTPS	Flow	SF			✓			ST	-	Off	Off	✓	
[171]	2018		✓		✓	-	-	Flow	SF		✓	✓			ST	-	Off	On	✓	
[123]	2016	✓	✓	✓		390k sessions	HTTPS	Session	SF		✓	✓		✓	ML	RF	Off	Off	✓	
[126]	2016	✓	✓			1060 traces	HTTPS	Session	SF	✓	✓	✓		✓	ML	RF	Off	Off	✓	
[124]	2018	✓	✓		✓	1000+ videos	QUIC	Flow	SF		✓	✓			ML	RF	Off	Off	✓	
[125]	2016	✓	✓		✓	-	HTTPS	Flow	SF	✓	✓	✓		✓	ML	RF	Off	On	✓	
[96]	2017					360 flows	-	Flow	SF		✓	✓	✓		ML	k-NN	Off	Off	✓	✓
[127]	2019	✓				3879 sessions	QUIC	Session	SF	✓	✓	✓	✓		ML	RF	Off	On	✓	
[139]	2020	✓	✓			2 databases	HTTPS	Flow	SF					✓	DL	CNN	Off	On	✓	✓

[1] In the *Traffic Representations* columns, *Level* is traffic representation level, *Style* is traffic representation style, where SF denotes statistical features, *PC* is packet content, *PL* is packet length, *PT* is packet timing, *PD* is packet direction.
[2] In the *Method* columns, *Train* is online training or offline training, *Predict* is online predicting or offline predicting.
[3] In the *Evaluation* columns, *ET* is effectiveness, *TO* is time overhead.

TABLE VII: A Conclusion For Studies Related To Protocol Recognition.

Analysis		Dataset		Traffic Representation[1]							Method[2]				Evaluation[3]			
Name	Year	Scale	EM	Level	Style	PC	PL	PT	PD	Others	Category	Classifier	Train	Predict	CW	OW	Efficiency	Overhead
[77]	2011	4 Protocols	-	Flow	SF		✓			✓	KB	-	Offline	Offline	✓		✓	
[78]	2014	30+ Protocols	TLS	Flow	SF					✓	KB	-	Offline	Offline/Online	✓	✓	✓	✓
[137]	2009	SSH Protocol	SSH	Flow	SF		✓	✓	✓		ML	k-Means	Offline	Online	✓		✓	
[174]	2017	Tor Service	Tor	Flow	SF		✓	✓			ML	Clustering	Offline	Offline	✓		✓	
[140]	2017	108 Services	-	Flow	GP		✓	✓	✓	✓	DL	CNN/RNN	Offline	Offline	✓		✓	
[141]	2019	5 Protocols	SSH	Flow	IM		✓	✓	✓		DL	CNN	Offline	Online	✓		✓	

[1] In the *Traffic Representations* columns, *Level* is traffic representation level, *Style* is traffic representation style, where SF denotes statistical features, *PC* is packet content, *PL* is packet length, *PT* is packet timing, *PD* is packet direction.
[2] In the *Method* columns, *Train* is online training or offline training, *Predict* is online predicting or offline predicting.
[3] In the *Evaluation* columns, *OW* is open world, *CW* is closed world.

are no RTT features in UDP traffic.

B. Protocol Recognition

Protocol recognition is an important foundation for network service providers and network administrators to provide differentiated Quality of Service (QoS) or carry out malware and intrusion detection. Traditionally, protocols are identified by ports which are assigned for the majority of standard protocols (e.g., HTTP, FTP and TELNET) by the Internet Assigned Numbers Authority (IANA). While port-based identification methods are less effective, as many applications allow users to customize the ports for communication [175]. Therefore, several approaches have been proposed to deal with this problem, which are summarised in Table VII.

Traditional machine learning methods. These methods have been widely used in protocol recognition, which is useful for network management and resource allocation. The extracted features, e.g., mean packet size, packet inter-arrival time, standard deviation in packet lengths and absolute deviation of packet sizes, rely on information that can be obtained from packet headers or the entire flow/session. For instance, Rao *et al.* [174] propose a method for Tor anonymous traffic identification based on cluster algorithm. Observing that the duration time between onion proxy and entry node of Tor network is usually fixed and long, the flow duration time is extracted as features. Moreover, the large number of fixed-length packets in the process of communication in Tor network is considered as another typical feature. Combined with the above features, they utilize the gravitational clustering algorithm as a classifier to realize the Tor traffic identification. However, the approach results in poor performance in classifying Tor and Http traffic, as the former is usually hidden behind the latter, which leads to similarity between them.

Deep learning methods. In addition to traditional machine learning methods, these methods are widely utilized to identify security protocols. Lopez-Martin *et al.* [140] present a new technique based on a combination of CNN and RNN that can be applied on IoT traffic to recognize the used services. In terms of feature extraction, they only consider the first 20 packets that are exchanged in a flow lifetime. 6 features are extracted from each packet header, i.e., source port, destination port, packet direction, bytes in payload, inter-arrival time and window size. They observe that compared with pure RNN model, RNN combined with a previous CNN model can achieve higher accuracy, as the location invariant patterns from the reshaped traffic can be extracted by CNNs.

Knowledge-based methods. These methods are used to identify protocols based on specific rules. The DPI method appears to obtain more accuracy in protocol recognition, which

is used to check the contents of the entire packet involving packet header and payload, and then match some pre-defined special strings to determine the specific application. For example, L7-filter [176], Libprotoident [177] and nDPI [78] are built based on DPI, where nDPI has the ability to process encryption protocols. It applies a decoder for SSL that extracts the host name, which can be leveraged for specific application identification.

C. Summary and Lessons Learned

In this section, we review existing studies that focus on encrypted traffic analysis applied in QoE metric measurement and protocol identification from the perspective of user experience. As shown in Table VI, 4 types of key factors affecting QoE are concluded, i.e., stalling, resolution, switch and bitrate. To measureg these factors, several statistical features are extracted from encrypted traffic, where the packet timing is the most commonly used feature, as it can reflect the speed of data transmission directly.

However, there are also several challenges related to network characterization. Firstly, as the requirements for real-time resource allocation and optimization mechanisms increase, such as the emergence of live broadcast applications, online deployment and updates of QoE prediction models are demanded. Secondly, the majority of aforementioned studies choose a subset of protocols in order to examine the effectiveness of approaches, which leads to potential scalability and adaptability issues. Moreover, since efficient data collection is significant, we suggest automating the process of data collection and processing, which enables the running of more massive measurement assignments and thus increases the robustness of prediction models.

VII. Privacy Leakage Detection

With the development of the Internet, large amounts of data are created and disseminated, which may contain personal private information. The user activities can be inferred from non-encrypted traffic effortlessly. Take web browsing for example, the website being visited can be confirmed through the HTTP packet header (e.g., host). While with the widespread encryption protocol, the packets are usually encrypted by TLS, making the HTTP header invisible. However, attackers can still obtain sensitive information from the encrypted traffic. In other words, despite the existence of encryption technologies, there is still a risk of privacy leakage through the channel of encrypted traffic analysis.

In this section, we will introduce three aspects of privacy leakage detection, which are website fingerprinting, application fingerprinting, and user action identification, respectively. Website fingerprinting is a kind of traffic analysis that attempts to learn about which website or webpage users are browsing. Similarly, application fingerprinting can be used to identify which application users are using. User action identification focuses on inferring the behaviors of users, e.g., sending an email on smartphones. A summary of approaches pertaining to privacy leakage detection are presented in Table VIII

It is worth mentioning that, from the aspect of defense, it helps administrators discover if there is a risk of privacy leakage, so they can protect user data by adding artificial noise [178]. However, from the aspect of the attack, the machine learning techniques for leakage detection might be repurposed as an adversarial tool to launch privacy attacks, e.g., re-identification attacks, inference and linkage attacks [179–182], which puts users at risk of privacy breaches even when only browsing on the Internet.

A. Website Fingerprinting

The popularity of the Internet has aroused people's attention to potential privacy threats associated with web browsing. Although encryption mechanisms like SSL/TLS or HTTPS establish encrypted tunnels to preserve communication contents, they are still endangered by website fingerprinting (WF) attacks, which leverage specific features of encrypted traffic, e.g., packet sizes, time and directions, to identify a website or webpage. Through the features, inadequate information about user identities, behaviors and preferences may be leaked.

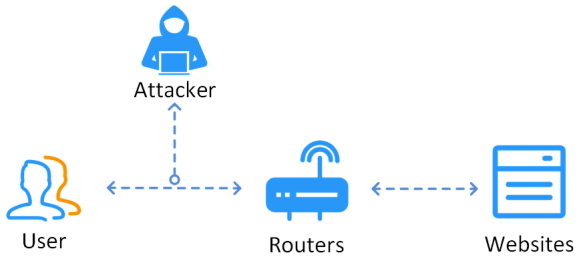


Fig. 7: Threat Model of Website Fingerprinting

The threat model of WF is shown in Fig. 7. First, the attacker utilizes sniffing tools such as Wireshark to monitor encrypted traffic generated by the victim. Then, the obtained traffic is segmented into flows or sessions, from which multiple features are extracted. After that, the attacker leverages classification techniques, e.g., machine learning algorithms, to identify from which website the traffic is generated. In this way, the web browsing information of the victim is leaked.

Traditional machine learning methods. The traditional machine learning classifiers, such as *k*-NN [100], SVM [106] and RF [128], are widely used in website fingerprinting. The effectiveness of different methods is usually evaluated in two scenarios, namely closed-world and open-world scenarios. In the closed-world assumption, users are assumed to access a small number of known websites, which simplifies the attack process. However, in practice, users can visit a larger amount of websites, thus it is unrealistic for the attacker to collect samples of all possible websites. The open-world assumption is more realistic, where only a fixed set of websites are assumed to be monitored by the attacker.

The popularity of Tor with anonymity urges researchers to propose novel WF attacks. Though Tor claims to provide anonymous browsing services for users through encryption, it cannot conceal the behavioral information of traffic, e.g., packet directions and packet inter-arrival time. Al-Naami *et*

TABLE VIII: A Conclusion For Studies Realted To Website & Application Fingerprinting And User Action Identification.

	Analysis		Dataset		Traffic Representation[1]							Method[2]				Evaluation[3]			
	Name	Year	Scale	EM	Level	Style	PC	PL	PT	PD	Others	Category	Classifier	Train	Predict	CW	OW	ET	TO
Website Fingerprinting	[79]	2010	2000 sites	OpenVPN/SSH	Session	SQ					✓	KB	-	Off	Off	✓	✓	✓	
	[89]	2014	11+ sites	VPN	Flow	SF		✓	✓		✓	ST	BayesNet	Off	Off	✓		✓	
	[106]	2011	1M pages	Tor/JAP	Session	SF	✓	✓	✓	✓		ML	SVM	Off	Off	✓	✓	✓	✓
	[107]	2016	300k pages	Tor	Session	SF		✓		✓	✓	ML	SVM	Off	Off	✓	✓	✓	✓
	[98]	2016	1000+ pages	Tor/HTTPS	Session	SF	✓	✓	✓		✓	ML	SVM/k-NN/RF	On	On	✓	✓	✓	✓
	[103]	2020	80k pages	Tor	Session	SF		✓	✓	✓		ML	SVM/k-NN	Off	Off	✓	✓	✓	
	[108]	2013	1000 sites	Tor	Session	SQ	✓	✓		✓	✓	ML	SVM	Off	Off	✓	✓	✓	
	[97]	2014	100 pages	Tor	Session	SF	✓	✓	✓	✓	✓	ML	k-NN	Off	Off	✓	✓	✓	✓
	[128]	2016	100k+ sites	Tor	Session	SF	✓		✓		✓	ML	RF/k-NN	Off	Off	✓	✓	✓	✓
	[101]	2017	33k+ traces	Tor	Session	SQ		✓	✓	✓		ML	k-NN	Off	Off	✓	✓	✓	
	[102]	2018	55k pages	Tor	Session	SF	✓	✓	✓	✓	✓	ML	k-NN	Off	Off	✓	✓	✓	
	[100]	2019	6k flows	SSL/TLS	Flow	SQ		✓				ML	k-NN	Off	On	✓		✓	✓
	[113]	2020	37k traces	SSL/TLS	Flow	SF	✓	✓				ML	k-NN/RF/DT	Off	Off	✓	✓	✓	✓
	[70]	2018	20k+ sites	Tor	Session	SQ				✓		DL	CNN	Off	Off	✓	✓	✓	✓
	[142]	2019	3 datasets	Tor	Session	SQ				✓		DL	CNN	Off	Off	✓	✓	✓	
Application Fingerprinting	[183]	2016	12 apps	SSL/TLS	Session	SF		✓				ST	Markov	Off	Off	✓		✓	
	[16]	2017	14 apps	SSL/TLS	Session	SF		✓				ST	Markov	Off	Off	✓		✓	✓
	[92]	2018	18 apps	SSL/TLS	Flow	SF		✓	✓			ST	Markov	Off	Off	✓		✓	
	[90]	2016	1595 apps	HTTPS	Session	SQ		✓				ST	Bayes	Off	On	✓		✓	
	[129]	2015	13 apps	WPA2	Flow	SF		✓	✓	✓		ML	RF	Off	On	✓		✓	
	[109]	2016	5 apps	HTTPS	Flow	SF		✓			✓	ML	RF	Off	Off	✓		✓	
	[130]	2018	110 apps	SSL/TLS	Flow	SF	✓	✓	✓	✓		ML	RF	Off	Off	✓		✓	
	[114]	2013	40 apps	HTTPS	Flow	SQ		✓				ML	DT	Off	Off	✓		✓	
	[71]	2021	1300 Dapps	SSL/TLS	Flow	GR		✓		✓	✓	DL	GNN	Off	Off	✓	✓	✓	✓
	[143]	2018	15 apps	HTTPS/SSH/SSL	Flow	SQ			✓		✓	DL	MLP/SAE/CNN	Off	On	✓		✓	✓
User Action Identification	[81]	2007	21 languages	SSL	Session	SF		✓				KB	-	Off	Off	✓		✓	✓
	[80]	2010	-	SSL	Flow	SQ		✓	✓			KB	-	Off	Off	✓		✓	
	[93]	2008	122 sentences	SSL	Flow	SQ		✓				ST	Markov	Off	Off	✓		✓	
	[86]	2014	-	SSH/HTTPS	Flow	SQ		✓				ST	Bayes	Off	Off	✓		✓	
	[132]	2015	3 apps	SSL/TLS	Flow	SF		✓	✓	✓	✓	ML	RF	Off	Off	✓		✓	
	[22]	2016	35 activities	SSL/TLS	Session	SF	✓	✓	✓	✓		ML	SVM	Off	Off	✓		✓	
	[184]	2017	7 usages	SSL/TLS	Flow	SF		✓	✓			ML	MVML	Off	Off	✓		✓	
	[133]	2018	16 actions	SSL/TLS	Flow	SF	✓	✓	✓	✓	✓	ML	RF	Off	Off	✓		✓	✓
	[110]	2017	2100 titles	HTTPS	Flow	SF		✓				ML	SVM	Off	Off	✓	✓	✓	
	[131]	2018	780 files	-	Flow	SF	✓	✓	✓	✓		ML	RF	Off	Off	✓		✓	
	[149]	2018	10 videos	HTTP	Flow	SF	✓	✓		✓		DL	CNN/LSTM/MLP	Off	On	✓		✓	
	[150]	2018	100 traces	WPA/WPA2-PSK	Flow	SQ		✓				DL	LSTM	Off	Off	✓		✓	

[1] In the *Traffic Representations* columns, *Level* is traffic representation level, *Style* is traffic representation style, where SF denotes statistical features, SQ denotes sequence and GR denotes graph, *PC* is packet content, *PL* is packet length, *PT* is packet timing, *PD* is packet direction.

[2] In the *Method* columns, *Train* is online training or offline training, *Predict* is online predicting or offline predicting.

[3] In the *Evaluation* columns, *CW* is closed world, *OW* is open world, *ET* is effectiveness, *TO* is time overhead.

al. [98] propose a WF attack by extracting features related to data dependencies occurring over sequential transmissions of packets, e.g., bursts, packet sizes and timestamp. To mitigate the impact of drifts between the current data and previously training data, the model needs to be re-trained periodically according to whether the accuracy drops below a certain threshold. Hence, there is an open issue of model fine tuning to overcome the data drift problem.

A superior representation of traffic plays a momentous role in WF. Wang *et al.* [108] propose a data collection scheme that uses the more fundamental Tor cells, which are data in units of 512 bytes, instead of TCP/IP packets as the meta-data. Since then, Tor cell has been certified and utilized in WF. Wang *et al.* [97] improve the effectiveness of the former in the open-world scenario by using k-NN. The fingerprints are constructed based on the features including bursts, packet sizes and packet ordering. However, it needs manually analysis to extract features that may contain significant information, to overcome this shortcoming, Panchenko *et al.* [107] propose a WF attack named CUMUL based on cumulated packet sizes and an SVM classifier. They leverage the cumulated sum of packet sizes as the traffic representation, from which a fixed number of features are derived. Specifically, they extract the features by sampling the piece-wise linear interpolant of the cumulated packet size sequence at n equidistant points. The feature sets proposed in [107] and [97] have proven to perform

well on website fingerprinting while they are less effective on webpage classification.

Webpage fingerprinting is a fine-grained variation of WF, where the attackers attempt to identify specific webpages belonging to the same website. The analysis of webpages is meaningful for privacy leakage detection due to the reason that the fine-grained analysis may reveal more sensitive information. For instance, the attacker can acquire information about which type of videos the victim prefers. However, the SSL/TLS traffic parameters of different webpages on the same websites are basically the same, making the classification methods based on SSL/TLS fingerprints less effective.

To achieve fine-grained WF, Shen *et al.* [100] focus on the classification of webpages from the same website. The cumulative length of a packet sequence is selected as the feature, which distinguishes between webpages. Only features extracted from the first 100 packets of a webpage loading trace are input into the k-NN classifier, balancing the accuracy and time complexity. As the cumulative packet length in bidirectional interaction between clients and servers describes the differences in the content transmitted on webpages, it is effective for fine-grained webpage fingerprinting. The authors in [113] extend the study in [100] by utilizing more distinguishing features and enlarging the experimental dataset. The regular patterns of the first few packets in the interaction process are leveraged to identify the webpages. They grasp distinctive features of webpages related to packet length in the uplink-dominant stage, including block features, sequence features and statistical features. Both high accuracy and low time overhead are achieved using the extracted features.

The selected features play an important role in WF, hence, there are several studies focusing on how to choose the distinctive features. For instance, Li *et al.* [102] select 3043 features used in the state-of-the-art WF attacks and empirically confirm that Tor leaks information about client connections. The mutual information is exploited to measure the amount of information that can be obtained through encrypted traffic, which illustrates that anonymous networks are still faced with eavesdropping. The above work quantifies the extent of information leakage, which brings inspiration in the aspect of feature selection.

Deep learning methods. Compared with traditional machine learning, deep learning requires more complex calculations and larger consumption, e.g., storage resources and time costs. As the input of the deep learning model, the samples classified need to be expressed in a regular form, e.g., a sequence of fixed lengths. Generally, deep learning methods weaken the interpretability of features, as feature extraction is done automatically by the model.

To protect the communication from encrypted traffic analysis, there are two defense strategies against WF attacks, which are WTF-PAD [185] and Walkie-Talkie [186], respectively. WTF-PAD is a defense method using adaptive padding that can increase padding when channel utilization is low. Walkie-talkie makes the network connection work in a half duplex mode as well as adds dummy packets and delays to mask the features of communications.

After that, robust WF methods have been proposed to maintain their performance on defended traffic. Sirinam *et al.* [70] propose a method named DeepFingerprinting based on CNNs. To be specific, the traffic is represented as sequences of packet directions (e.g., 1 for incoming packets and -1 for outgoing packets), which is used for automatic feature extraction by the CNN classifier. Even though the traffic statistical features are partly obscured by dummy packets, the remaining sequence features can still be learned by the CNN model. Gong *et al.* [187] observe that the front of traffic leaks the largest amount of information. They propose a WF defense method, named Front, to add dummy packets to the front of traffic according to Rayleigh distribution. The experiments show that it can reduce the effectiveness of DF [70].

To further improve the transferability of WF methods in different scenarios, the authors in [70] propose Triplet Fingerprinting [142], which is built on a machine learning technique named N-shot learning that requires a few training samples to identify a given class. While they do not address the problem of multi-tab web browsing [188], i.e., the users visit multiple websites continuously, where the traces of the websites may overlap. It remains an open issue of multi-trace splitting and classification.

In addition for WF attacks, deep learning, e.g., the generative adversarial net (GAN), also has applications in the privacy protection [179] to prevent attackers from inferring sensitive information from user browsing traffic. For example, as deep neural networks are vulnerable to adversarial examples, Nasr *et al.* [189] leverage adversarial perturbation to disturb the original traffic patterns. They design an adversarial network and a dummy packet inserting approach to create adversarial perturbations on live traffic, which is effective for againsting Var-CNN [190] and DF [70].

Recently, the one-page setting, a standard for evaluating WF defenses, is proposed by Wang [191]. The author argues that the evaluation of WF defense should not be set in a scenario where there are a large number of websites in the open world, which strongly favors the defender. He proposes that WF defense should be evaluated with only one monitored website and one non-monitored website. The experiment results show that under one-page setting, Decoy [192], Front [187] and Tamaraw [193] failed to defend against WF.

Knowledge-based methods. These methods are utilized in WF because of their convenience and efficiency. Specifically, calculating the similarities between samples to be classified and samples of known categories is one of the commonly used methods. Lu *et al.* [79] consider information about packet sizes and propose an approach to analyze the similarity between fingerprints using Levenshtein distance, which is effective for WF in the case of encryption and proxy channels. Considering that the Maximum Transmission Unit (MTU) downloading packets contain less information, only the size of the last packet transferring the remaining data in each data chunk is extracted as the fingerprint. The proposed method can guarantee the effectiveness of defending strategies such as traffic morphing and padding. While the extracted features depend heavily on packet size information, which is not available for Tor network.

B. Application Fingerprinting

With the profusion of mobile applications, the resulting network traffic increases dramatically. The existing studies usually regard the process of identifying applications through traffic analysis as application fingerprinting (AF). On the one hand, AF helps network administrators identify the applications that are visited by users in their networks, which can be used for network management. For example, through encrypted traffic analysis, the administrator can block access to forbidden applications for security control. On the other hand, AF leaks the users privacy such as their commonly used applications. For example, the usage of sensitive applications such as job-hunting and health testing applications may cause privacy exposure. In addition, grasping the information on sensitive applications visited by potential victims, attackers may provide them with false information and commit fraud such as phishing attacks.

Due to the widely used encryption mechanisms, traditional methods such as payload-based classification are no longer feasible. Multiple machine learning methods (e.g., RF [129]), deep learning methods (e.g., CNN [143]) and statistical methods (e.g., Markov model [16]) have been proposed for AF.

Traditional machine learning methods. Behavioral features such as sizes, lengths and directions are used in these methods, as alternatives to packet contents, which is invisible in encrypted traffic. Wang *et al.* [129] leverage the frame inter-arrival time, size and direction during bursts, which contain several frames aggregated densely within a period of time, as traffic features to categorize iOS and Android applications. Shen *et al.* [183] combine certificate packet length with second-order Markov Chain to identify applications. However, there are several occasions where the probability computed with the fingerprint of correct application is lower than that of the wrong application, as certificate packet lengths of the two applications are subject to the same cluster. To solve the issue, they extend the leveraged information to two types, i.e., the first application data length, and the certificate packet size in [16]. However, there are conditions that produce SSL/TLS flows without the certificate packets. To solve the problem, Liu *et al.* [92] leverage both message type sequences and packet length sequences as features, as there are overlaps in message type sequences between different applications. However, this method needs to load the entire flow before performing feature generation, weakening their capabilities for timely classification.

Capturing the traffic of specific targets contrapuntally has been the most commonly-employed strategy in practice. Nevertheless, it is not simple to distinguish and filter background traffic completely. Moreover, the mixture of target traffic and background traffic may weaken the effectiveness of the classification model. To solve the issue, Mongkolluksamee *et al.* [109] filter short-lived flows according to flow duration constraint to remove background traffic. Specifically, they construct graphlet consisting of five-tuple information and then utilize 59-dimension features extracted from graphlet and packet size distribution to classify monitored applications. However, it is not clear how scalable and robust their approach is since they consider only five applications. Moreover, the application fingerprints may be influenced by different devices, versions or time, which corresponds to the robustness and the stability of analysis methods. To investigate how to identify apps or how app fingerprints change, Taylor *et al.* [130] utilize a total of 54 statistical features to construct burst vectors, e.g., the variance, skew and kurtosis of packet sizes, and train the ambiguity detection classifier which involves a reinforcement learning strategy. They evaluate the robustness of their approach across various app versions, devices or changing time, the last of which is the least influential factor.

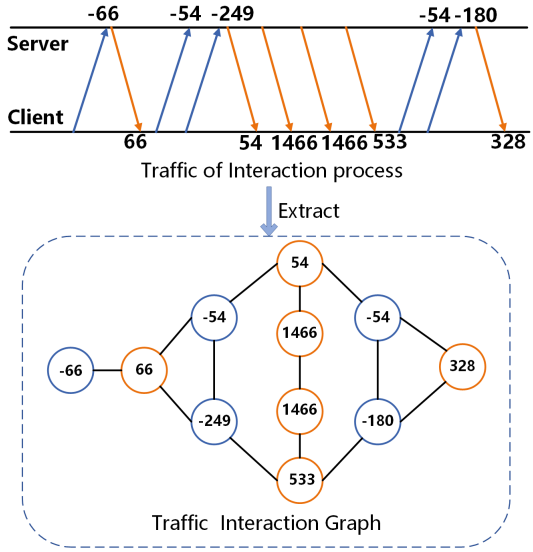


Fig. 8: Traffic Interaction Graph [71]

Deep learning methods. Deep learning automatically extracts and selects features, reducing the cost of manual feature extraction greatly. For example, decentralized applications (DApps) are increasingly developed and deployed on blockchain platforms. [71, 194] However, DApps deployed on the same platforms may share similar encryption settings, which reduces the discrimination of the generated traffic. To identify DApp accurately, Shen *et al.* [71] propose GraphDApp to extract traffic interaction graph using the method shown in Fig. 8 and utilize GNN for identifying decentralized applications that users visit. The traffic interaction graph reserves several dimensional features like packet length, direction and so on. Furthermore, their method is verified in classification of traditional applications and is applicable. While it needs to be updated regularly as the applications change.

C. User Action Identification

Apart from websites and applications, identifying fine-grained user actions from encrypted traffic is essential for network management. Various service providers research user behaviors or request types in order to manage services and improve QoS better. In addition, several studies analyze the specific traffic of applications, e.g., voice call, to detect the leakage of user privacy.

Traditional machine learning methods. These methods are widely used in user action identification. Despite the use of SSL/TLS, the traffic analysis approach is still an effective tool that an eavesdropper can leverage to undermine the privacy of mobile users. Conti *et al.* [132] propose a user action identification method based on clustering and RF. Packet header information (e.g., ports), packet sizes and packet directions are considered as features. The method takes advantage of the most distinctive flows belonging to a particular user action for classification. However, there are only 3 applications and a subset of actions related to those applications are analyzed. Thereafter, Saltaformaggio *et al.* [22] expand the application data set and propose a method to detect 35 types of behaviors in iOS and Android applications, such as reading news. Instead of extracting the fingerprints of an activity directly, they cut the traffic into segments, which represent the traffic's behaviors in a time window. Whether the traffic contains specific behaviors is considered as the fingerprints to classify activities. While the method is vulnerable to imitation attacks, i.e., replaying traffic that belongs to other applications to confuse the classifier.

Watching videos online by multiple users will generate a large amount of traffic, which may leak users privacy such as religious faith and sexual orientation. Encrypted traffic analysis is applied to video identification to prove the information leakage from video streaming. The packet size features extracted for other purposes, such as audio content analysis, are not reliable for video classification, as the payload size in video streaming is often set to be the maximum size. Dubin *et al.* [110] propose a method to classify the video titles on YouTube from encrypted traffic. They observe that the total number of bits at a peak, which is defined as a section of traffic where there is silence before and after, is an informative feature. Therefore, they construct a feature vector containing the bits of every peak in a video stream and make classification based on the similarity between a given sample and the known classes. Specifically, during the experiments, the noise traffic, e.g., non-YouTube packets and TCP re-transmission packets are filtered out at the state of preprocessing. While in a real network environment, the re-transmission event may change the feature distribution and even hinder accurate classification.

The widespread use of smart IoT devices gives rise to rampant concerns on user privacy. Smart devices such as Amazon Echo, which can interact with users via voice commands, may leak sensitive information such as contents spoken from traffic transmitting between the cloud service and itself. Jackson *et al.* [131] focus on the information leaked by smart speakers such as Amazon Echo. They collect the encrypted traffic sent from the cloud service to the Echo device, which associates with the response to the user request. Several statistical features extracted from TCP packets, such as the histogram of packet inter-arrival time, are exploited to identify the type of user requests, e.g., requests for weather and directions. It demonstrates the information leakage of user request type in smart speakers, but its generalizability across different networks, users, and devices still needs further investigation.

Deep learning methods. These methods are useful for identifying video contents or user actions. Li *et al.* [149] develop RNNs requiring less computational power and time than state-of-the-art classifiers to identify YouTube videos. It does not require complete TCP/IP flow information, which makes it suitable for online video identification. Nevertheless, the authors consider only 10 videos so it is unclear whether the method is effective in unveiling more videos.

To detect the existence of wireless cameras and infer user presence status, Ji *et al.* [150] utilize packet lengths and LSTM to identify whether there is a wireless camera in the target area. Furthermore, the count or size of video and audio frames of wireless camera may change with motions and voices of users, resulting in different bitrate. As such, they infer user presence by detecting the bitrate changes caused by human motions. However, slight movements of users may not change the state of camera, resulting in failure recognition, which makes the method less effective.

Knowledge-based methods. Various communication applications are arising to provide convenience to users. However, an encrypted VoIP conversation does not prevent attackers from eavesdropping completely. Knowledge-based methods can be leveraged to discover the sensitive information, e.g., language used [81], which leaks users privacy in the communication process. Dupasquier *et al.* [80] focus on recognizing the specific sentence during the conversation using Skype, which is a widely used VoIP application. The packet sizes are used as features to achieve isolated phoneme classification. Through manually separating sentences, the dynamic time warping (DTW) algorithm is used to calculate whether each sentence is the specific sentence previously set. However, due to the variety of sentences in natural language, it is impossible to build a DTW model for each sentence.

D. Summary and Lessons Learned

In this section, we review three types of encrypted traffic analysis related to privacy leakage detection, i.e., website fingerprinting, application fingerprinting, and user action identification. Discriminative feature extraction and appropriate classifier selection are two key points in privacy leakage detection, affecting the effectiveness of models significantly. Packet length and packet direction are two widely used features in disclosing traffic private information. From coarse-grained WF to fine-grained action analysis, more discriminative traffic representations, e.g., burst features, and more complex network structures, e.g., GNN, are used to dig out leaks.

Based on the limitations of existing approaches, several issues should be further investigated. First, the variety of dataset should be taken into consideration. For instance, the traffic used for WF is supposed to be collected from multiple web browsers (e.g., Firefox, Chrome and Internet Explorer) and multiple platforms (e.g., Windows, Linux and iOS), which can be leveraged to validate the generality of models. Second, in response to these privacy attacks, practical defenses have been proposed by adding dummy packets or delays into the traffic to mask the original features. Those countermeasures hinder the effectiveness of privacy attacks, which requires exploring traffic features surviving under these defenses.

TABLE IX: A Conclusion For Studies Related To Anomaly Detection.

	Analysis		Dataset		Feature[1]								Method[2]				Evaluation[3]		
	Ref	Year	Platform	Collection point/Dataset	Level	Style	PH	PC	PL	PT	PD	Others	Category	Classifier	Train	Predict	Uk	ET	TO
Malware detection	[82]	2014	Mobile	Gateway	Flow	SF		✓	✓	✓	✓		KB	-	Off	Off		✓	
	[83]	2018	IoT	Between device and hub	Flow	SF			✓		✓		KB	-	Off	On		✓	
	[115]	2017	Mobile	Gateway	Flow	SF	✓	✓	✓	✓	✓		ST/ML	DT/k-NN/Bayes	Off	Off	✓	✓	✓
	[154]	2015	Mobile	-	Session	SF	✓	✓	✓	✓	✓		ST/ML	Bayes/DT/RF/	Off	Off	✓	✓	
	[195]	2017	Mobile	Gateway	Flow	SF		✓		✓	✓		ML	RF/k-NN/DT	Off	Off		✓	
	[157]	2019	Mobile	Host/Gateway	Flow	SF		✓	✓	✓	✓		ML	XGBoost	Off	Off		✓	✓
	[116]	2018	Mobile	Gateway	Session	SF		✓	✓		✓		ML	DT	Off	Off		✓	
	[91]	2017	Mobile	-	Flow	SF		✓	✓	✓	✓		ML	Bayes	Off	Off	✓	✓	✓
	[134]	2020	IoT	Between device and hub	Flow	SF	✓		✓	✓	✓		ML	RF	Off	Off		✓	
	[144]	2018	Mobile	Stratosphere IPS project	Flow	SF		✓	✓	✓	✓		ML/DL	RF/CNN	Off	Off		✓	
	[151]	2017	Mobile	Cloud firewall	Flow	SF	✓		✓				DL	LSTM	Off	Off		✓	✓
	[145]	2020	Mobile	CICAndMal2017	Flow	SQ							DL	Autoencoder/CNN	Off	Off		✓	
	[146]	2020	Mobile	CICAndMal2017	Flow	SQ							DL	CNN	Off	Off		✓	
Network anomaly detection	[84]	2015	-	CAIDA 2007	Session	SF	✓					Entropy	KB	-	Off	Off		✓	
	[138]	2015	Simulation network	MIT-DARPA 1999	Flow	SF	✓		✓			Entropy	ML	k-Means	On	On	✓	✓	
	[196]	2015	Isolated network	RGCE	Flow	SF	✓	✓	✓	✓	✓		ML	DBSCAN	On	On	✓	✓	
	[117]	2018	Simulation network	UNSW-NB15	Flow	SF	✓	✓	✓	✓	✓		ML/DL	RF/DT/SVM/MLP	Off	Off		✓	✓
	[158]	2017	SDN	-	Flow	SF	✓	✓		✓			ML	Bagged Trees	Off	Off		✓	✓
	[135]	2020	Blockchain	-	Flow	SF			✓	✓	✓	Entropy	ML	RF	Off	Off		✓	
	[136]	2020	IoT	Gateway	Flow	SF	✓	✓					ML	RF	Off	On	✓	✓	
	[69]	2016	Isolated network	RGCE	Flow	SF	✓	✓	✓			Duration	ML/DL	SOM/SAE	Off	Off	✓	✓	
	[197]	2020	Campus network	CTU-13	Flow	SF	✓	✓	✓	✓	✓	Duration	DL	ANN	Off	Off		✓	
	[152]	2019	Simulation network	ISCX	Flow	SQ							DL	CNN/LSTM/SAE	Off	Off		✓	

[1] In the *Traffic Representations* columns, *Level* is traffic representation level, *Style* is traffic representation style, where SF denotes statistical features and SQ denotes sequence, *PH* is packet header, *PC* is packet content, *PL* is packet length, *PT* is packet timing, *PD* is packet direction.

[2] In the *Method* columns, *Train* is online training or offline training, *Predict* is online predicting or offline predicting.

[3] In the *Evaluation* columns, *Uk* is unknown attack, *ET* is effectiveness, *TO* is time overhead.

VIII. Attack Detection

With the rapid evolution of the Internet, the complexity of network topology and the scale of devices has increased accordingly, which brings an outbreak trend of anomalous behaviors. The increasing popularity of mobile phones and IoT devices attracts malware developers who earn profit from the malicious applications implanted into victims' devices. Moreover, there are numerous network attacks on various platforms, e.g., campus networks, enterprise networks, IoT networks and blockchain networks. Given the significant growth of anomalous behaviors, there is a pressing need for timely and effective anomaly detection.

The traditional anomaly detection methods commonly scan packet contents to find malicious patterns based on the predetermined signatures, e.g., the unique part of the malware which is unlikely found in any benign traffic [198]. However, due to traffic encryption, payload-based anomaly detection methods are less effective. Thus, novel traffic representations and anomaly detection methods suitable for encryption traffic need to be proposed. As described in Table IX, we mainly focus on the applications of encrypted traffic analysis in malware detection and network anomaly detection.

A. Malware Detection

Malware is software designed to damage devices or networks. After being implanted in the target device, the malware will run executable code or scripts to damage it. For example, malware like WannaCry and Petya disrupted business operations by exposing vulnerabilities in software, which leads to immense financial loss. The rampant malware makes it necessary for intelligent intrusion detection systems.

Traffic analysis methods are largely used in malware detection, which attempts to leverage unique features reflected in traffic to detect the behavior of the target software during execution. While the encryption leveraged in malware communication hinders payload-based feature extraction, which motivates researchers to find effective fingerprints that can be used for encryption traffic.

Traditional machine learning methods. Machine learning has become the most commonly used intelligent data analysis technology. A large amount of studies apply machine learning techniques to malware detection. We divide these studies into the application of machine learning methods in mobile malware detection and IoT malware detection.

With the development of wireless communication technology [182, 199], smartphones have become indispensable, which are closely related to sensitive information exchange.

However, the software installed on mobile phones may bring potential dangers to users. Smartphones store a large amount of private information, thereby, they are the main targets of cyber-attacks [200], which makes it urgent to build an efficient malware detection system to protect privacy.

Lashkari *et al.* [195] propose an Android malware detection model to protect mobile device users from malicious applications. They leverage 3 feature selection algorithms, i.e., information gain [201], cfs subset evaluation [202], and SVM, to select the common features from the feature set. Finally, the selected features, e.g., maximum and minimum packet length, are extracted from traffic and then are fed into 5 classifiers, e.g., k-NN and DT. They compare the precision, recall, accuracy and false positive of the 5 classifiers and observe that the DT model performs well in two tasks, i.e., distinguishing malware from benign applications and classifying malware into general malware and adware (software that generates revenue by generating advertisements automatically). To improve the performance of a single machine learning model, ensemble learning is used in malware detection. Rahmat *et al.* [157] apply bagging and boosting algorithms to create ensemble models. They divide the dataset into smaller subsets and each of the subsets is then provided to train a base learner. The outputs obtained by these learners are voted to obtain the final result. They compare 5 ensemble learners (e.g., AdaBoost, XGBoost) as well as 4 single learners (e.g., Naïve Bayes, SVM) and observe that the XGBoost model achieves the highest accuracy.

The above methods require a large number of resources for feature calculation and classifier training. However, mobile phones have limited computing power and storage, which necessitates a lighter detection method to improve user experience. Offloading the calculations from mobile to remote servers can be used to solve the challenge. Following this idea, Wang *et al.* [116] utilize a mirror technology to transfer traffic to the server, which is used for traffic filtering, analyzing and detecting. To detect malware, they train a DT model with 6 statistical features including uploading or downloading bytes, uploading or downloading packet number and average byte of uploading or downloading packets. However, this method performs poorly in unknown types of malware detection, as it relies on existing features, which may not appear in unknown malware.

Given the various types of malware, there is a pressing demand to propose methods to detect the malware family that is not included in the training set, which requires extracting the features that are prevalent in malware traffic. Garg *et al.* [115] extract representative malware features, e.g., time properties and packet numbers, then feed them into machine learning classifiers. They compare 5 classifiers including DT, logistic regression, k-NN, Bayes as well as RF, and observe that the k-NN and RF achieve the highest and stablest accuracy. To examine the ability of unknown malware detection, the leave-one-out method is used, i.e., a type of malware is left out as the test set and the model is trained with the rest of the malware traffic. The results show that if there is similar malware in the training set, the detection of unknown malware is effective. However, it has low identification rates when there is no similar malware available in the trained model. Therefore, it is an open issue of unknown malware detection.

In addition to mobile phones, smart home IoT devices are also vulnerable to malware. With the popularity of smart home platforms supporting third-party application development, people pay more and more attention to the security and privacy loopholes of smart home IoT devices. Intrinsic design flaws of Samsung-owned SmartThings that lead to significant over privilege in SmartApps are discovered [203], which promotes the development of IoT malware detection.

Malware that controls IoT devices may behave inconsistently with its declared working patterns. Gu *et al* [134] propose an IoT malware detection method by analyzing the sequence of traffic packets to discover whether the state of the IoT device is consistent with the state declared by the software. They extract IoT context using natural language processing (NLP) to represent the actions the user expected and generate wireless context inferred from the wireless communication traffic to describe the actual actions. By comparing the differences of the contexts, malware can be identified. However, the detection process is time consuming due to the process of NLP analysis, which limits its ability to detect malware in a real-time manner.

Moreover, the fabricated ad request is another malicious traffic, which leads to losses for advertisers. "Click farm" is one of the most common ways to fabricate the visits (e.g., the number of times an app has been downloaded), which leverages multiple mobile devices to simulate normal user operations. To discover these bot devices, Sun [204] propose a mobile ad fraud detection method based on invalid traffic analysis. Firstly, they develop a classifier to distinguish bot devices from the normal user by leveraging 11 features, e.g., the entropy of ad slot IDs. Secondly, devices are clustered based on app usage patterns. Finally, the devices in a cluster are related by majority voting of all devices in the same cluster.

Deep learning methods. As the diversity and variability of traffic features, Malware detection is considered as a complex problem, which is difficult to be solved by traditional machine learning methods. Therefore, deep learning methods are introduced to solve two problems, i.e., automatic feature extraction and effective feature learning.

Compared with traditional machine learning methods which require a lot of prior knowledge to select input features, deep learning methods can automatically extract them from raw traffic. Feng *et al.* [145] propose a two-layer deep learning model for Android malware detection. The traffic analysis is based on a cascaded model of CNN and AutoEncoder, which takes the 2-D images as input. Thus the malware detection problem is translated to image classification problem. This method achieves high accuracy in binary classification (2-label) and category classification (4-label), while it will lead to low accuracy for malicious family classification, as the images are similar between malicious families in the same category.

As the structure of deep neural network makes it has a stronger ability to fit complex functions, it is more suitable to learn traffic features comprehensively. For example, traffic is composed of sequential packets and adjacent packets are usually closely related to each other, which makes LSTM suit-

able for the traffic analysis, as its ability of handling sequential input and dealing with long-term dependencies. Following this idea, Prasse *et al.* [151] propose a client malware detection method based on the LSTM network. Compared with RF classifier, the method they proposed performs better.

B. Network Anomaly Detection

As the Internet has become the primary universal communication infrastructure, it is also subject to a variety and an increasing number of attacks. Network attacks such as Denial of Service (DoS), User to Root (U2R), Remote to Local (R2L) and Probing Attack increase rapidly, which pushes cyberspace security becoming an important topic. Traffic analysis methods are widely used to build intrusion detection systems, while encryption technology block accesses to the contents of data packets, which brings challenges to attack detection. Therefor, the Internet needs more effective protection measures suitable for encrypted traffic. As summarized in Table IX, in this section, we mainly introduce the encrypted traffic analysis used in network anomaly detection.

Traditional machine learning methods. Machine learning methods are largely used in the network anomaly detection. As shown in Fig. 9 , we can further divide the network anomaly detection based on machine learning methods into anomaly-based detection, classification-based detection and hybrid detection according to the prior knowledge. For anomaly-based detection, only normal traffic is leveraged in model building, which is usually constructed with unsupervised learning methods. For classification-based detection, both normal and abnormal traffic are leveraged to build a classifier with supervised learning methods. For hybrid detection, it is the combination of anomaly-based and classification-based methods.

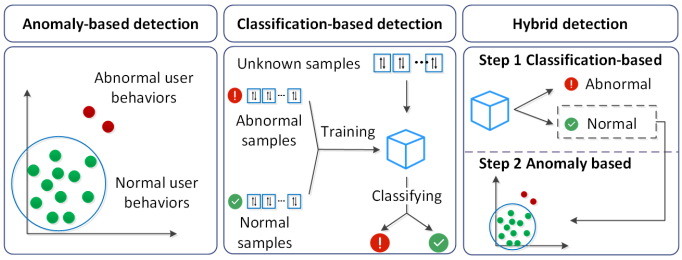


Fig. 9: Three types of network anomaly detection methods based on machine learning

Anomaly-based detection methods construct models of normal user behaviors leveraging features extracted from them. On the basis of the models, behaviors that deviate from the normal patterns are classified as abnormal.

Qin *et al.* [138] leverage the entropy of destination addresses, destination ports, source addresses, packet sizes and flow durations to model normal users' patterns using k-means algorithm. Through detecting deviations from the built models, they can judge whether a DDoS attack has occurred. Similarly, Zolotukhin *et al.* [196] build a normal user behavior model using DBSCAN algorithm with 9 features, e.g., inter-arrival time, extracted from traffic. If the target traffic deviates from the normal model by greater than the threshold, it is judged as a DoS attack traffic. The main difference between the above methods is that the former builds several normal user behavior models, which considers multiple types of user behaviors, while the latter builds only one, which regards all outliers as abnormal values.

Anomaly-based methods are usually realized by clustering algorithms, which have the advantages of updating the built models online. Moreover, This type of method augments the ability to detect unknown anomalies. However, if the attacker can mimic the behavior of a normal human user, traffic related to the attack may be clustered into the normal class.

The modeling process of the anomaly-based method rarely considers anomalous traffic, but it may contain key discriminative information. Due to this reason, several studies consider the problem of network anomaly detection as a classification problem [117]. They combine normal and abnormal traffic to train a classifier to determine whether the target object is network anomaly traffic.

In order to build the detection system suited to SDN, Ajaeiya *et al.* [158] propose a method which leverages the built-in statistics collected by the OpenFlow switch at 1-second interval to detect network attacks. The statistics are aggregated to 9 features, e.g., standard deviation of packets and standard deviation of bytes, which are used to build a classifier using bagged trees. The system runs over the control plan and is therefore transparent to users, which is an advantage of SDN. In general, the special structure of SDN introduces the ability of traffic collection, which brings convenience for network attack detection system construction.

In blockchain networks, special types of network attack detection can be regarded as a binary classification problem [205]. As an emerging infrastructure, blockchain has been applied in many fields such as healthcare, IoT, energy and manufacturing [206]. However, it still has many security problems, which threaten network security. Eclipse attack is an attack that may cause serious consequences to the blockchain network, which isolates the victimized blockchain node from the normal network by occupying the routing table of it. To detect the eclipse attack in Ethereum, Xu *et al.* [135] leverage encrypted traffic analysis to detect whether the target node is being attacked. They utilize information entropy, packet size, packet frequency and connection time to describe the eclipse attack features and use RF as the classifier.

However, the classification-based detection method has an inherent limitation that it is difficult to deal with unknown types of attacks, as the classification process is based on features extracted from existing malicious behaviors, which may be changed as the malware evolves.

To further improve the ability of network anomaly detection, Hybrid detection methods which combine anomaly-based detection and classification-based detection are proposed. Generally, the hybrid detection method is shown as a two-step method. The first step is classification-based detection and the second step is anomaly-based detection. Samples classified as "normal" in step 1 will be input into the model in step 2 for abnormal or outlier detection. IoTArgos [136] combines

supervised and unsupervised learning algorithms to discover suspicious behaviors. They make the observation that the 11 features, e.g., remote ports and packets per 5 min, are effective for network detection. To be specific, supervised machine learning algorithm is used for filtering a subset of attack traffic, while unsupervised machine learning algorithm is exploited for discovering samples misclassified as 'normal' at the previous step.

Compared with the anomaly-based detection and classification-based detection, the hybrid detection method achieves both high classification accuracy of known attacks and detection capability of unknown attacks.

Deep learning methods. Due to the strong feature extraction ability, it is applied in network anomaly detection.

On the one hand, deep learning methods are able to extract features from raw traffic. Zeng *et al.* [152] propose an end-to-end intrusion detection framework based on deep learning. Three deep learning models including CNN, LSTM and SAE are used for anomaly detection. Compared with traditional machine learning methods, the deep learning algorithm requires less manual intervention, which provides greater automation.

On the other hand, the multi-layer neural network has the ability to combine shallow features to form higher-level features that are more complex and abstract, which is suitable to discover the covert attack behavior. Following this idea, Zolotukhin *et al.* [69] combine both clustering algorithm and SAE algorithm to build a DDoS attack detection model. For each time interval, 8 types of features, e.g., percentage of packets with different TCP flags, are extracted as the input of clustering and SAE model. A normal user behavior model is built using clustering algorithm, which divides the behavior deviated from it as DoS attack. SAE is used to detect the attack that is able to mimic the browsing behavior of a regular human user, which cannot be discovered by the clustering algorithm.

Knowledge-based methods. Generally, these methods depend on artificial rules to determine whether it is network attack traffic. The rules are usually based on ports, protocol headers and artificial features. Jisa *et al.* [84] calculate the fast entropy of flow count for each connection as the fingerprint, which needs only packet header information. They build a rule with an adaptive threshold for DDoS attack detection. However, the proposed detection method relies on flow count, which makes it less effective for classifying flash crowds (massive legitimate packets arrive in a short time) [207] and DDoS attacks.

C. Summary and Lessons Learned

In this section, we introduce existing studies that focus on encrypted traffic analysis applied in anomaly detection, including malware detection and network anomaly detection. The traffic patterns are distinctive between normal and abnormal user behaviors, e.g., DDoS attack generates more illegal connections compared with normal actions, from which the anomaly can be detected. As for unknown types, the clustering algorithm is introduced to solve the problem, where the traffic deviating from the normal cluster is classified as abnormal. However, its performance largely relies on the effectiveness of the extracted features, which makes it difficult to detect malicious behaviors that mimic normal users.

We summarize that high overhead, self-adaptive and zero-day attack discoveries are three main dilemmas of anomaly detection based on encrypted traffic analysis. Firstly, for the purpose of protecting resources, malicious behaviors are expected to be detected as soon as possible, which requires timely feature extraction approaches and efficient classification models. Secondly, due to the dynamics and diversity of malware and network anomaly, the detection models are expected to update the strategies (e.g., extracted features and decision thresholds) as illegitimate traffic changes mask in a short while. Moreover, new malware and network anomaly are surging. Detection systems that are appropriate for zero-day attack discovering, e.g., anomaly-based methods, are more suitable for the realistic network environments in terms of unknown malicious behavior detecting.

IX. Challenges and Future Research Directions

Although researchers have made substantial achievements in developing encrypted traffic analysis methods, there remain significant challenges around dataset construction, traffic representation, analysis model building, and potential countermeasures. At the same time, unprecedented study opportunities are also provided to develop innovative approaches to tackle these challenges.

A. Traffic Dataset Construction

Network traffic dataset is a crucial component for traffic analysis, as high-quality datasets can play an important role in training, e.g., helping to train accurate and robust classifiers for supervised learning, and in validation, e.g., helping to evaluate the performance of traffic analysis methods, no matter whatever technique is used. The requirements in constructing high-quality encrypted traffic datasets are two-fold.

First, datasets should provide a good amount of variety to cover diverse scenarios in reality. For instance, in supervised learning, to construct accurate website fingerprinting classifiers, an ideal dataset for training should consist of instances generated by visiting numerous websites through diverse web browsers (e.g., Chrome, Firefox, IE, Safari) on various types of devices (e.g., laptops, PCs, smartphones, and pads) that are equipped with all major OS types (e.g., Windows, Linux, iOS, Android) and get access to the Internet via different approaches (e.g., fixed network, 3G/4G/5G, and WiFi). Without adequate training data, even the classifier with the best performance would be rendered useless. Second, to quantitatively measure the effectiveness of a certain traffic analysis method, such as checking the correctness of the results from machine learning classifiers, we should be able to obtain factual data as ascertainable through direct observation, which is known as the ground truth.

It is quite challenging to obtain datasets with sufficient diversity and undisputed ground truth. We refer to a straightforward way that is commonly used in the literature as snowball method. The basic idea of this method is to simulate a vasty of

TABLE X: Summary of challenges and future research directions

Challenge	Details	Future directions
Traffic Dataset Construction	• Diversification • Ground truth	• Snowball method • Gold panning method
Traffic Representation	• Effectiveness • Robustness • Time efficiency • Interpretability	• Diging into the interaction process • Stream representations • Representation construction and number of packets required • Interpretable machine learning
Analysis Model Building	• Effectiveness • Generalizability • Transferability	• Considering the open-world and defended scenarios • Networking heterogeneity, parameter variations and temporal drift • Dataset transferability and task transferability
Countermeasures	• Effectiveness • Overhead	• Generative adversarial networks

user actions, such as simulating web visiting behaviors of real users, and then collect the resulting traffic traces. It usually starts with a small set of user actions, and then enlarges the set to cover a variety of combinations of software, hardware, and environmental variations. It can obtain data samples with accurate ground truth, but requires significant manual efforts. An alternative way is referred to as gold panning method, which collects traffic from network devices in real-world environments, e.g., recording traffic traversing through the egress router of a campus network, as real-world traffic has sufficient diversity. Since the target traffic traces are usually mixed with a much larger amount of background traces, time-consuming efforts are needed to filter out untargeted traces, like the gold panning process that separates gold from other materials. A major obstacle to this approach is the lack of ground truth for most encrypted traffic traces.

To break through the dilemma between accuracy and diversity, auto-labeling tools are highly desirable to automatically collect accurate and adequate data samples for a given analysis purpose. Since ensuring sufficient diversity can be time-consuming and costly in terms of human efforts, crowdsourcing emerges as a promising approach to reduce the complexity, where multiple users or volunteers are involved to complete the same data collection and labeling tasks together. There are several issues to be further addressed, such as operational standards to measure data quality, incentives to attract more participants, and privacy protection strategies to secure participants sensitive information. Additional efforts can also be put on benchmark datasets in different research domains, which can greatly facilitate a fair and comprehensive comparison of different approaches.

B. Traffic Representation

Traffic representation can be of great importance to extract multi-dimensional information from encrypted traffic. Informative representations can extract the most distinctive features, which helps improve the accuracy of both supervised and unsupervised machine learning models. Existing studies have proposed several representations at different levels of granularity. For flow-level representation, statistic features [68] are usually fed into traditional machine learning classifiers, while the sequence of directional packets, image-based abstraction [152], graph-based abstraction [71], frequency domain [208] are proposed to work with deep learning classifiers. For session-level representation, the sequences of packet direction or timing [70] are used. Recently, raw bitmap representation [209] combined with automated machine learning is proposed to reduce reliance upon feature selection, hence it is suitable for multiple traffic analysis tasks.

It is still a challenging task to propose an appropriate representation of encrypted traffic. To achieve this goal, several issues should be carefully considered and addressed.

Effectiveness. To facilitate the classification or clustering of different classes of encrypted traffic, an effective representation should be discriminative enough to make each class easily separated from one another. More specifically, a powerful representation should make a traffic instance similar to those in the same class while distinctive from those in a different class. Existing representations usually abstract encrypted traffic from one or multiple aspects, such as packet order, timing, direction and frequency [75]. Except for abstracting these superficial features, future studies can dig into the interaction process between a pair of communication components (e.g., client-server interactions) and attach the traffic with more semantic knowledge of the application-layer data in transmission. This can provide us with more insights into the fundamental reasons that lead to traffic differences among different classes. In addition, appropriate evaluation metrics are needed to quantitatively measure the effectiveness of different representations.

Robustness. Traffic representation should have the ability to resist change of encrypted traffic without significantly adapting its initial construction. Potential changes can take many forms, such as protocol upgrading, content updates, and network condition changes. Statistical representations are usually sensitive to these changes, as a small magnitude of changes would render the original representations inefficient. This can be demonstrated by the quantitative measure of information leakage of different statistical features, where the features contribute mostly to website fingerprinting vary greatly when the traffic is perturbed. In contrast, the stream representations are more robust to such changes, as they construct traffic abstractions from the original packet streams and thereby are

not severely affected by potential changes. The current forms of stream representation (e.g., sequence, graph, and image) usually borrow ideas from other research domains, such as speech signal processing, natural language processing, and computer vision. More powerful and robust representations are expected to be proposed to reflect the inherent characteristics of encrypted network traffic.

Time efficiency. It is desirable to construct traffic representation in a time-efficient manner so as to accelerate the model training process or make decisions rapidly for on-the-fly traffic instances. Time efficiency can be explained in two aspects. The first one, which is commonly employed in the literature, is time consumption on construing representation for a batch of encrypted traffic instances. Statistical representations usually require complex computation of multi-dimensional features, such as entropy and frequency spectrum. Stream representations vary a lot in terms of time consumption, depending on the specific form of a representation. For instance, sequence-based and graph-based representations are quite easy to obtain from the original encrypted traffic, whereas image-based representation can result in a dramatic increase of time overhead for reforming packet sequence information into the image-like structure. The second aspect is usually ignored in existing studies, which can be roughly interpreted as the number of packets required for representation construction. Since packets are arrived at a monitoring point in sequence, the smaller the number of packets used, the less the time spent on representation construction. Several statistical features are able to be extracted by the end of the session, which cost a lot of time [120]. Therefore, the number of packets used is crucial for online training or testing. Statistical representations require complete flows or packet sequences, thus resulting in poor time efficiency, while typical stream representations (e.g., sequence- or graph-based ones) can be obtained with a flexible number of packets. Improving time efficiency of representation construction is an interesting topic. The challenge is carefully balance the trade-offs between effectiveness and time efficiency, i.e., achieving discriminative representation using as few features and packets as possible.

Interpretability. As the machine learning methods, especially deep learning techniques, have been widely used for encrypted traffic analysis, interpretability becomes a challenging issue, which means the degree to which a human can understand the cause of a decision. The need for interpretability arises from an eagerness to know why a machine learning method ends with such predictions. In other words, getting correct predictions only partially solves the problem. For instance, in a QoE metric estimation task, besides accurate predication of a well-trained machine learning method, network operators are also eager to know the major reasons that result in poor QoE metrics (e.g., low video resolution on user side), which can be interpreted from typical network quality indicators (e.g., RTT, packet loss rate, and inter-packet gap). With these indicators, the operators can launch certain optimization strategies on network devices so as to improve QoE perceived by end users. Statistical representations owe the advantage of interpretability, because the features can be used to reason the prediction of classifiers. While stream representations are not interpretable enough as the features are automatically extracted and learned as in a black-box scenario. Recent advances in interpretable machine learning can be employed to improve the interpretability of encrypted traffic analysis methods.

C. Analysis Model Building

Traffic encryption uncovers payload segments of packets, which reduce the information that can be exploited for analysis significantly. Thus, most existing work resorts to machine learning techniques to build analysis models. As mentioned above, the traffic analysis problem is usually treated as a supervised or unsupervised classification problem, where the classifiers can be roughly grouped into two categories.

In the first category, the classifiers usually leverage traditional machine learning models (e.g., k-NN, SVM, RF, and DT), which are carefully trained with empirically crafted features. The classifiers in the second category employ deep learning models (e.g., CNN, MLP, and LSTM) that can automatically learn effective features from raw input data and thus avoid the complicated feature engineering process. An emerging trend appears to combine deep learning models and traditional machine learning techniques together, e.g., effective features are first extracted using deep learning models and then used to train traditional machine learning classifiers.

In order to improve the performance of a classifier, several challenging issues should be well addressed.

Effectiveness. The fundamental goal of a classifier is to achieve high effectiveness, which can be measured by a series of metrics, such as accuracy, precision, recall, etc. Most existing work focuses on evaluating the effectiveness of a classifier in the closed-world setting [100], where the traffic is assumed to be generated from a limited number of targeted sources (e.g., websites). A classifier with high accuracy in the closed-world scenario, however, loses efficacy in the more realistic open-world scenario, as the encrypted traffic can be generated from a much larger number of untargeted sources than the targeted ones. In other words, the targeted traffic may take only a small fraction of the total amount of traffic in the open-world setting, and the classifier should also be able to distinguish between the targeted and untargeted traffic.

In order to evade traffic analysis tools, several defenses have been proposed to reduce the information leaked out from the encrypted traffic. Defenses negatively affect the effectiveness of classifiers, as less discriminative features can be extracted from defended traffic. Existing studies have confirmed that the accuracy of classifiers drops significantly on defended traffic. An interesting problem is to investigate how to maintain the effectiveness of classifiers in distinguishing between the targeted and untargeted traffic when all the traffic is protected.

Generalizability. A robust classifier should remain effective under different networks and environmental conditions. Many existing studies assume that the pre-trained classifiers are applied in scenarios having similar or even the same conditions as in the training process. This assumption gives the classifier an unrealistic advantage for making predictions,

as the conditions for testing would be different from those for training. We summarize the differences as following:

- Networking heterogeneity. Network encrypted traffic can be generated from users or devices with heterogeneous network conditions, including fixed network and wireless network (e.g., WiFi and 3G/4/G/5G). The locations where network traffic is collected can also be different, e.g., at the locations near to end users (e.g., the access point), both upstream and downstream traffic can be collected, whereas at the locations far away from end users (e.g., the backbone routers) only unidirectional traffic can be collected due to routing asymmetry in the real world.
- Parameter variations. Network encrypted traffic can be generated with different hardware and software parameters. The range of hardware covers PCs, laptops, smartphones, IoT devices, wearable devices, network servers and devices, while the range of software parameters covers OS types, browser brands and versions, protocol versions (e.g., TLS and Tor-browser-bundle).
- Temporal drift. The sources for generating encrypted traffic would change over time, e.g., the layout and content of a webpage can vary frequently, or the upgrade of data transmission protocols such as Dynamic Adaptive Streaming over HTTP (DASH), making the accuracy of a well-trained classifier decay after a certain period.

Transferability. In order to reduce the complexity and resource consumption on frequently re-training a classifier, it is highly desirable that the classifier can be flexibly transferred to different datasets and tasks. Currently, most supervised classifiers are trained with a fixed number of labels, where the classifiers attempt to learn the optimal mapping from labelled samples (traffic traces or features) to their corresponding labels. Then, in the prediction phase, the classifier outputs the probability vector for a given unlabeled sample. In practice, more flexibility is needed to design an out-of-the-box classifier.

- Dataset transferability. The label set in the training dataset would change dynamically. For instance, the adversary conducting website fingerprinting attacks may wish to enlarge or shrink the set of targeted websites by adding or removing the corresponding labels. Another example can be found in the estimation of video resolutions, where a coarse-grained label set (e.g., high, medium, and low resolutions) is replaced by a fine-grained label set consisting of specific resolutions (e.g., 144P, 360P, 480P, 720P, and 1080P).
- Task transferability. Most classifiers are trained for a specific task, which results in poor transferability among tasks. For instance, when the classifiers for application recognition are directly applied for user action recognition, the accuracy would drop significantly, as the features of different tasks vary a lot. As a result, the shift of traffic analysis goals inevitably leads to classifier re-training.

The current advances in transfer learning provide promising opportunities to improve the transferability of classifiers, which can also ameliorate the generalization issues.

D. Countermeasures

In order to protect against traffic analysis methods, many countermeasures have been proposed over the years. Existing solutions mainly focus on two aspects: 1) protecting user-side information leakage against fingerprinting attacks, and 2) obfuscating malicious traffic to evade anomaly traffic detectors. Next, we briefly describe the achievements of current solutions and discuss the challenges.

Website Fingerprinting Defenses. To defend against passive WF attacks, several methods have been proposed to make the targeted traffic traces less discriminative. To achieve this goal, they typically employ two approaches: inserting dummy packets, and/or deferring the sending of certain packets. Wang et al. [187] classify existing defenses into three types: traffic obfuscation, confusion, and regularization. For ease of understanding, we can roughly summarize the basic ideas behind these defenses into two categories.

The first category aims at making traffic traces as similar as possible, by inserting dummy packets or enlarging inter-packet delays. The defended traffic traces from a targeted site become indistinguishable from those from another one (e.g., Walkie-talkie [186] or Decoy [106]) or multiple sites (e.g., Supersequence). This idea is analogue to the k-anonymity strategy in protecting sensitive location information. The second category aims to make traffic traces as stochastic as possible, by inserting dummy packets following a certain distribution (e.g., a fixed rate distribution in WTF-PAD [185], or the Rayleigh distribution in Front [187]). With the adoption of such defenses, the resulting traffic traces from the same site become apparently different, making the attackers difficult to extract consistent and discriminative features.

Anomaly Detection Escape. To evade anomaly traffic detection, several methods have been proposed to mislead the classifiers by traffic obfuscation. The basic idea behind these methods is to disguise malicious traffic as normal traffic, by changing the features of malicious traffic. Such as intermediate DDoS attack, bots generate random browser-like requests to make the attack behavior can not be distinguished from the regular human traffic [69].

Next, we discuss the challenges to design efficient and effective countermeasures.

The effectiveness and overhead should be taken into consideration when designing countermeasures. The effectiveness is usually measured as the accuracy of a specific classifier on traffic traces when the countermeasures are applied. Since the goal of changing traffic features is usually achieved by adding dummy packets or deferring packet delivery, the overhead is commonly measured by the bandwidth overhead and time overhead, where the former denotes the fraction of bandwidth increase compared with the original traffic traces, while the latter denotes the fraction of time extension for completing the packet transmission.

There usually exists a conflict between these two factors. For instance, to reduce the overhead introduced by potential countermeasures, dummy packets should be added as little as possible, and at the same time, packets should maintain their original delivery timetables. This, however, would have limited impacts on changing the original traffic features, resulting in a

poor defense effect on classifiers. Thus, to design appropriate countermeasures for traffic analysis models, a better balance should be achieved between effectiveness and overhead.

The recent advances in GANs can be helpful to improve the effectiveness of defenses. GANs have proved to be effective in many fields such as face recognition and acoustical signal processing. By crafting adversarial examples, the perturbed samples can successfully cheat a deep learning classifier. It is not straightforward to apply GAN techniques in the domain of encrypted traffic analysis, due to the following barriers:

- Unforeseeable traffic traces. The traffic traces are generated in a real-time manner, the defender can hardly learn the whole traces in advance. To address this problem, historical traces can be used to learn a relatively constant pattern as a basic prediction of unforeseeable traces. Since many factors have influences on traces, the gap between real and predicted traces can always exist and would negatively impact the effectiveness of defenses.
- Limited manipulation methods. Unlike the diversified ways to craft adversarial images, the methods to manipulate traffic traces are limited to adding dummy packets or deferring packet delivery. In other words, the defender cannot delete or speed up packets in the original traffic traces, which narrows the space to craft effective adversarial traffic traces.
- Limited prior knowledge of classifiers. Most classifiers conduct traffic analysis in a passive manner, where the victim knows little prior knowledge of these classifiers. As a result, the defenses are blinded to the features that can be employed by potential attacks. In such a scenario, the specific classifiers can be treated as a black-box and the defenses are expected to have good defense effects no matter which classifier is used.

X. Conclusion

This paper surveys significant studies in the field of multi-target and multi-method encrypted traffic analysis. Before starting the introduction of the specific studies, we review the network encryption technologies and introduce the goals of encrypted traffic analysis. In particular, we summarize the workflow including traffic collection, traffic representation, traffic analysis method, and performance evaluation.

In terms of classification goals, we review four application scenarios including network asset identification, network characterization, privacy leakage detection and anomaly detection. Network asset identification section focuses studies of device fingerprinting and OS identification. By probing network assets, administrators can keep abreast of the dynamic changes, which provides connivance for device management. Then, we review the studies on privacy leakage detection including WF, AF and fine-grained user action identification. By detecting sensitive information leaked from website browsing and application using, malicious eavesdroppers may commit criminal activities such as financial fraud. Next, the network characterization section focuses on QoE metric measurement and protocol recognition, which helps service providers to grasp the network environment and adjust service strategies timely. As for the anomaly detection section, we introduce two types of malicious behaviors, which are malware and network attacks, respectively. Novel representations and detection methods for encrypted traffic are introduced. Finally, we discuss challenges and future directions, which provide new ideas for developing innovative approaches.

In conclusion, encrypted traffic analysis plays an essential role in the field of network security protection and personalized service guarantee. However, the analysis is a mixed blessing which breaks the security of the network and probes user privacy to a certain extent. Hence, there are also several studies contributing to countermeasures against encrypted traffic analysis. Due to this reason, determining how to properly use encrypted traffic analysis is a problem worthy of consideration. To conclude, with the rapid development of the network and related technologies, encrypted traffic analysis will be a vibrant field and continue to flourish.

References

[1] Eric Rescorla. *SSL and TLS: designing and building secure systems*, volume 1. Addison-Wesley Reading, 2001.

[2] Transparencyreport. https://transparencyreport.google.com/https/overview, 2022.

[3] Anna L Buczak and Erhan Guven. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys and Tutorials*, 18(2):1153–1176, 2016.

[4] Xuyang Jing, Zheng Yan, and Witold Pedrycz. Security data collection and data analytics in the internet: a survey. *IEEE Communications Surveys & Tutorials*, 21(1):586–618, 2018.

[5] Gilberto Fernandes, Joel JPC Rodrigues, Luiz Fernando Carvalho, Jalal F Al-Muhtadi, and Mario Lemes Proença. A comprehensive survey on network anomaly detection. *Telecommunication Systems*, 70(3):447–489, 2019.

[6] Donghwoon Kwon, Hyunjoo Kim, Jinoh Kim, Sang C Suh, Ikkyun Kim, and Kuinam J Kim. A survey of deep learning-based network anomaly detection. *Cluster Computing*, pages 1–13, 2017.

[7] Petr Velan, Milan Čermák, Pavel Čeleda, and Martin Drašar. A survey of methods for encrypted traffic classification and analysis. *International Journal of Network Management*, 25(5):355–374, 2015.

[8] Shahbaz Rezaei and Xin Liu. Deep learning for encrypted traffic classification: An overview. *IEEE communications magazine*, 57(5):76–81, 2019.

[9] Mauro Conti, Qian Qian Li, Alberto Maragno, and Riccardo Spolaor. The dark side (-channel) of mobile devices: A survey on network traffic analysis. *IEEE Communications Surveys & Tutorials*, 20(4):2658–2713, 2018.

[10] Fannia Pacheco, Ernesto Exposito, Mathieu Gineste, Cedric Baudoin, and Jose Aguilar. Towards the deployment of machine learning solutions in network traffic

classification: a systematic survey. *IEEE Communications Surveys & Tutorials*, 21(2):1988–2014, 2018.

[11] Anat Bremler-Barr, Yotam Harchol, David Hay, and Yaron Koral. Deep packet inspection as a service. In *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*, pages 271–282, 2014.

[12] Giorgos Dimopoulos, Pere Barlet-Ros, and Josep Sanjuas-Cuxart. Analysis of youtube user experience from passive measurements. In *Proceedings of the 9th International Conference on Network and Service Management (CNSM 2013)*, pages 260–267, 2013.

[13] Raimund Schatz, Tobias Hoßfeld, and Pedro Casas. Passive youtube qoe monitoring for isps. In *2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pages 358–364. IEEE, 2012.

[14] Meng Shen, Jinpeng Zhang, Siqi Chen, Yiting Liu, and Liehuang Zhu. Machine learning classification on traffic of secondary encryption. In *2019 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, 2019.

[15] Meng Shen, Yiting Liu, Liehuang Zhu, Ke Xu, Xiaojiang Du, and Nadra Guizani. Optimizing feature selection for efficient encrypted traffic classification: A systematic approach. *IEEE Network*, 34(4):20–27, 2020.

[16] Meng Shen, Mingwei Wei, Liehuang Zhu, and Mingzhong Wang. Classification of encrypted traffic with second-order markov chains and application attribute bigrams. *IEEE Transactions on Information Forensics & Security*, 12(8):1830–1843, 2017.

[17] Mario Bkassiny, Yang Li, and Sudharman K Jayaweera. A survey on machine-learning techniques in cognitive radios. *IEEE Communications Surveys & Tutorials*, 15(3):1136–1159, 2012.

[18] Zhengwei Wang, Qi She, and Tomas E Ward. Generative adversarial networks in computer vision: A survey and taxonomy. *ACM Computing Surveys (CSUR)*, 54(2):1–38, 2021.

[19] Qing Wu, Yungang Liu, Qiang Li, Shaoli Jin, and Fengzhong Li. The application of deep learning in computer vision. In *2017 Chinese Automation Congress (CAC)*, pages 6522–6527. IEEE, 2017.

[20] Laizhong Cui, Shu Yang, Fei Chen, Zhong Ming, Nan Lu, and Jing Qin. A survey on application of machine learning for internet of things. *Int. J. Mach. Learn. Cybern.*, 9(8):1399–1417, 2018.

[21] Susan Athey and Guido W. Imbens. Machine learning methods that economists should know about. *Annual Review of Economics*, 11(1):685–725, 2019.

[22] Brendan Saltaformaggio, Hongjun Choi, Kristen Johnson, Yonghwi Kwon, Qi Zhang, Xiangyu Zhang, Dongyan Xu, and John Qian. Eavesdropping on fine-grained user activities within smartphone apps over encrypted network traffic. In *10th USENIX Workshop on Offensive Technologies (WOOT 16)*, Austin, TX, August 2016. USENIX Association.

[23] Weiting Zhang, Dong Yang, Youzhi Xu, Xuefeng Huang, Jun Zhang, and Mikael Gidlund. Deephealth: A self-attention based method for instant intelligent predictive maintenance in industrial internet of things. *IEEE Transactions on Industrial Informatics*, 17(8):5461–5473, 2020.

[24] Jiayi Zhang, Hongyang Du, Qiang Sun, Bo Ai, and Derrick Wing Kwan Ng. Physical layer security enhancement with reconfigurable intelligent surface-aided networks. *IEEE Transactions on Information Forensics and Security*, 16:3480–3495, 2021.

[25] Meng Shen, Hao Lu, Fei Wang, Huisen Liu, and Liehuang Zhu. Secure and efficient blockchain-assisted authentication for edge-integrated internet-of-vehicles. *IEEE Transactions on Vehicular Technology*, pages 1–13, 2022.

[26] Kajaree Das and Rabi Narayan Behera. A survey on machine learning: concept, algorithms and applications. *International Journal of Innovative Research in Computer and Communication Engineering*, 5(2):1301–1309, 2017.

[27] Yuben Qu, Chao Dong, Jianchao Zheng, Haipeng Dai, Fan Wu, Song Guo, and Alagan Anpalagan. Empowering edge intelligence by air-ground integrated federated learning. *IEEE Network*, 35(5):34–41, 2021.

[28] Zehui Xiong, Yang Zhang, Dusit Niyato, Ruilong Deng, Ping Wang, and Li-Chun Wang. Deep reinforcement learning for mobile 5g and beyond: Fundamentals, applications, and challenges. *IEEE Vehicular Technology Magazine*, 14(2):44–52, 2019.

[29] Yi Liu, Xingliang Yuan, Zehui Xiong, Jiawen Kang, Xiaofei Wang, and Dusit Niyato. Federated learning for 6g communications: Challenges, methods, and future directions. *China Communications*, 17(9):105–118, 2020.

[30] Minrui Xu, Jialiang Peng, B. B Gupta, Jiawen Kang, Zehui Xiong, Zhenni Li, and Ahmed A. Abd El-Latif. Multi-agent federated reinforcement learning for secure incentive mechanism in intelligent cyber-physical systems. *IEEE Internet of Things Journal*, pages 1–1, 2021.

[31] Junfei Qiu, Qihui Wu, Guoru Ding, Yuhua Xu, and Shuo Feng. A survey of machine learning for big data processing. *EURASIP Journal on Advances in Signal Processing*, 2016(1):1–16, 2016.

[32] Miles N Wernick, Yongyi Yang, Jovan G Brankov, Grigori Yourganov, and Stephen C Strother. Machine learning in medical imaging. *IEEE signal processing magazine*, 27(4):25–38, 2010.

[33] Pariwat Ongsulee. Artificial intelligence, machine learning and deep learning. In *2017 15th international conference on ICT and knowledge engineering (ICT&KE)*, pages 1–6. IEEE, 2017.

[34] Jafar Alzubi, Anand Nayyar, and Akshi Kumar. Machine learning from theory to algorithms: an overview. In *Journal of physics: conference series*, volume 1142, page 012012. IOP Publishing, 2018.

[35] Mohammed Ali Al-Garadi, Amr Mohamed, Abdulla Khalid Al-Ali, Xiaojiang Du, Ihsan Ali, and

Mohsen Guizani. A survey of machine and deep learning methods for internet of things (iot) security. *IEEE Communications Surveys & Tutorials*, 22(3):1646–1685, 2020.

[36] Thanasis Kotsiopoulos, Panagiotis Sarigiannidis, Dimosthenis Ioannidis, and Dimitrios Tzovaras. Machine learning and deep learning in smart manufacturing: The smart grid paradigm. *Computer Science Review*, 40:100341, 2021.

[37] Pramila P Shinde and Seema Shah. A review of machine learning and deep learning applications. In *2018 Fourth international conference on computing communication control and automation (ICCUBEA)*, pages 1–6. IEEE, 2018.

[38] Yang Xin, Lingshuang Kong, Zhi Liu, Yuling Chen, Yanmiao Li, Hongliang Zhu, Mingcheng Gao, Haixia Hou, and Chunhua Wang. Machine learning and deep learning methods for cybersecurity. *Ieee access*, 6:35365–35381, 2018.

[39] Irina Rish et al. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46, 2001.

[40] John G Kemeny and J Laurie Snell. *Markov chains*, volume 6. Springer-Verlag, New York, 1976.

[41] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

[42] William S Noble. What is a support vector machine? *Nature biotechnology*, 24(12):1565–1567, 2006.

[43] S Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3):660–674, 1991.

[44] Gérard Biau and Erwan Scornet. A random forest guided tour. *Test*, 25(2):197–227, 2016.

[45] Rui Xu and Donald Wunsch. Survey of clustering algorithms. *IEEE Transactions on neural networks*, 16(3):645–678, 2005.

[46] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[47] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[48] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[49] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[50] Sheila Frankel. *Demystifying the IPsec Puzzle*. Artech House, 2001.

[51] Sheila Frankel, Karen Kent, Ryan Lewkowski, Angela D Orebaugh, Ronald W Ritchey, and Steven R Sharma. Guide to ipsec vpns:. 2005.

[52] Petr Velan, Milan Čermák, Pavel Čeleda, and Martin Drašar. A survey of methods for encrypted traffic classification and analysis. *International Journal of Network Management*, 25(5):355–374, 2015.

[53] Eric Rescorla and Tim Dierks. The transport layer security (tls) protocol version 1.3. 2018.

[54] Tatu Ylonen, Chris Lonvick, et al. The secure shell (ssh) protocol architecture, 2006.

[55] Mittal S Bhiogade. Secure socket layer. In *Computer Science and Information Technology Education Conference*, pages 85–90. Citeseer, 2002.

[56] Adam Langley, Alistair Riddoch, Alyssa Wilk, Antonio Vicente, Charles Krasic, Dan Zhang, Fan Yang, Fedor Kouranov, Ian Swett, Janardhan Iyengar, et al. The quic transport protocol: Design and internet-scale deployment. In *Proceedings of the conference of the ACM special interest group on data communication*, pages 183–196, 2017.

[57] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. Technical report, Naval Research Lab Washington DC, 2004.

[58] Multi-site connectivity capability package v1.1. https://www.nsa.gov/Resources/Commercial-Solutions-for-Classified-Program/Capability-Packages/.

[59] Rajendra Singh Panwar and Krishna M Sivalingam. Implementation of wrap around mechanism for system level simulation of lte cellular networks in ns3. In *2017 IEEE 18th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–9. IEEE, 2017.

[60] Jacob H Cox, Joaquin Chung, Sean Donovan, Jared Ivey, Russell J Clark, George Riley, and Henry L Owen. Advancing software-defined networks: A survey. *IEEE Access*, 5:25487–25526, 2017.

[61] Laizhong Cui, F Richard Yu, and Qiao Yan. When big data meets software-defined networking: Sdn for big data and big data for sdn. *IEEE network*, 30(1):58–65, 2016.

[62] Michał P Karpowicz and Piotr Arabas. Preliminary results on the linux libpcap model identification. In *2015 20th International Conference on Methods and Models in Automation and Robotics (MMAR)*, pages 1056–1061. IEEE, 2015.

[63] Piyush Goyal and Anurag Goyal. Comparative study of two most popular packet sniffing tools-tcpdump and wireshark. In *2017 9th International Conference on Computational Intelligence and Communication Networks (CICN)*, pages 77–81. IEEE, 2017.

[64] Laura Chappell. *Wireshark network analysis*. PODBOOKS. COM, LLC, 2012.

[65] Borja Merino. *Instant traffic analysis with Tshark howto*. Packt Publishing Ltd, 2013.

[66] Rick Hofstede, Pavel Čeleda, Brian Trammell, Idilio Drago, Ramin Sadre, Anna Sperotto, and Aiko Pras. Flow monitoring explained: From packet capture to data analysis with netflow and ipfix. *IEEE Communications Surveys & Tutorials*, 16(4):2037–2064, 2014.

[67] Nicola Bonelli, Andrea DiPietro, Stefano Giordano, Gregorio Procissi, and Fabio Vitucci. Towards smarter probes: In-network traffic capturing and processing. In *Trustworthy Internet*, pages 289–301. Springer, 2011.

[68] Hiroki Kawai, Shingo Ata, Nobuyuki Nakamura, and Ikuo Oka. Identification of communication devices from analysis of traffic patterns. In *2017 13th International Conference on Network and Service Management (CNSM)*, pages 1–5. IEEE, 2017.

[69] Mikhail Zolotukhin, Timo Hämäläinen, Tero Kokkonen, and Jarmo Siltanen. Increasing web service availability by detecting application-layer ddos attacks in encrypted traffic. In *2016 23rd International Conference on Telecommunications (ICT)*, pages 1–6. IEEE, 2016.

[70] Payap Sirinam, Mohsen Imani, Marc Juarez, and Matthew Wright. Deep fingerprinting: Undermining website fingerprinting defenses with deep learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 1928–1943. ACM, 2018.

[71] Meng Shen, Jinpeng Zhang, Liehuang Zhu, Ke Xu, and Xiaojiang Du. Accurate decentralized application identification via encrypted traffic analysis using graph neural networks. *IEEE Transactions on Information Forensics and Security*, 16:2367–2380, 2021.

[72] Tal Shapira and Yuval Shavitt. Flowpic: A generic representation for encrypted traffic classification and applications identification. *IEEE Transactions on Network and Service Management*, 18(2):1218–1232, 2021.

[73] Ke Gao, Cherita Corbett, and Raheem Beyah. A passive approach to wireless device fingerprinting. In *2010 IEEE/IFIP International Conference on Dependable Systems & Networks (DSN)*, pages 383–392. IEEE, 2010.

[74] Martin Husák, Milan Čermák, Tomáš Jirsík, and Pavel Čeleda. Https traffic analysis and client identification using passive ssl/tls fingerprinting. *EURASIP Journal on Information Security*, 2016(1):6, 2016.

[75] Nicholas Ruffing, Ye Zhu, Rudy Libertini, Yong Guan, and Riccardo Bettati. Smartphone reconnaissance: Operating system identification. In *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pages 1086–1091. IEEE, 2016.

[76] Jiaxi Gu, Jiliang Wang, Zhiwen Yu, and Kele Shen. Walls have ears: Traffic-based side-channel attack in video streaming. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pages 1538–1546. IEEE, 2018.

[77] Guang Cheng and Song Wang. Traffic classification based on port connection pattern. In *2011 International Conference on Computer Science and Service System (CSSS)*, pages 914–917. IEEE, 2011.

[78] Luca Deri, Maurizio Martinelli, Tomasz Bujlow, and Alfredo Cardigliano. ndpi: Open-source high-speed deep packet inspection. In *2014 International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 617–622. IEEE, 2014.

[79] Liming Lu, Ee-Chien Chang, and Mun Choon Chan. Website fingerprinting and identification using ordered feature sequences. In *European Symposium on Research in Computer Security*, pages 199–214. Springer, 2010.

[80] Stefan Burschka, Kieran Mclaughlin, and Sakir Sezer. Analysis of information leakage from encrypted skype conversations. *International Journal of Information Security*, 9(5):313–325, 2010.

[81] Charles V Wright, Lucas Ballard, Fabian Monrose, and Gerald M Masson. Language identification of encrypted voip traffic: Alejandra y roberto or alice and bob? In *USENIX Security Symposium*, volume 3, pages 43–54, 2007.

[82] Anshul Arora, Shree Garg, and Sateesh K. Peddoju. Malware detection using network traffic analysis in android based mobile devices. In *2014 Eighth International Conference on Next Generation Mobile Apps, Services and Technologies*, pages 66–71, 2014.

[83] Wei Zhang, Yan Meng, Yugeng Liu, Xiaokuan Zhang, Yinqian Zhang, and Haojin Zhu. Homonit: Monitoring smart home apps from encrypted traffic. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 1074–1088. ACM, 2018.

[84] Jisa David and Ciza Thomas. Ddos attack detection using fast entropy approach on flow-based network traffic. *Procedia Computer Science*, 50:30–36, 2015.

[85] Lingjing Yu, Tao Liu, Zhaoyu Zhou, Yujia Zhu, Qingyun Liu, and Jianlong Tan. Wdmti: Wireless device manufacturer and type identification using hierarchical dirichlet process. In *2018 IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pages 19–27. IEEE, 2018.

[86] Scott E Coull and Kevin P Dyer. Traffic analysis of encrypted messaging services: Apple imessage and beyond. *ACM SIGCOMM Computer Communication Review*, 44(5):5–11, 2014.

[87] Yi-Chao Chen, Yong Liao, Mario Baldi, Sung-Ju Lee, and Lili Qiu. Os fingerprinting and tethering detection in mobile networks. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, pages 173–180, 2014.

[88] Mashael AlSabah, Kevin Bauer, and Ian Goldberg. Enhancing tor's performance using real-time traffic classification. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 73–84. ACM, 2012.

[89] Yan Shi and Subir Biswas. Website fingerprinting using traffic analysis of dynamic webpages. In *2014 IEEE Global Communications Conference*, pages 557–563. IEEE, 2014.

[90] Hasan Faik Alan and Jasleen Kaur. Can android applications be identified using only TCP/IP headers of their launch time traffic? In *ACM Conference on Security & Privacy in Wireless and Mobile Networks*, pages 61–66, 2016.

[91] Anshul Arora and Sateesh K Peddoju. Minimizing network traffic features for android mobile malware detection. In *Proceedings of the 18th International*

Conference on Distributed Computing and Networking, pages 1–10, 2017.

[92] Chang Liu, Zigang Cao, Gang Xiong, Gaopeng Gou, Siu-Ming Yiu, and Longtao He. MaMPF: Encrypted traffic classification based on multi-attribute markov probability fingerprints. In *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, pages 1–10. IEEE, 2018.

[93] Charles V Wright, Lucas Ballard, Scott E Coull, Fabian Monrose, and Gerald M Masson. Spot me if you can: Uncovering spoken phrases in encrypted voip conversations. pages 35–49, 2008.

[94] Amin Shahraki, Mahmoud Abbasi, Amir Taherkordi, and Anca Delia Jurcut. A comparative study on online machine learning techniques for network traffic streams analysis. *Computer Networks*, 207:108836, 2022.

[95] Abbas Acar, Hossein Fereidooni, Tigist Abera, Amit Kumar Sikder, Markus Miettinen, Hidayet Aksu, Mauro Conti, Ahmad-Reza Sadeghi, and Selcuk Uluagac. Peek-a-boo: I see your smart home activities, even encrypted! In *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pages 207–218, 2020.

[96] Yu Ning Dong, Jia Jie Zhao, and Jiong Jin. Novel feature selection and classification of internet video traffic based on a hierarchical scheme. *Computer Networks*, 119:102–111, 2017.

[97] Tao Wang, Xiang Cai, Rishab Nithyanand, Rob Johnson, and Ian Goldberg. Effective attacks and provable defenses for website fingerprinting. pages 143–157, San Diego, CA, 2014. USENIX Association.

[98] Khaled Al-Naami, Swarup Chandra, Ahmad Mustafa, Latifur Khan, Zhiqiang Lin, Kevin Hamlen, and Bhavani Thuraisingham. Adaptive encrypted traffic fingerprinting with bi-directional dependence. In *Conference on Computer Security Applications*, pages 177–188, 2016.

[99] Xun Gong, Negar Kiyavash, and Nikita Borisov. Fingerprinting websites using remote traffic analysis. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 684–686. ACM, 2010.

[100] Meng Shen, Yiting Liu, Siqi Chen, Liehuang Zhu, and Yuchao Zhang. Webpage fingerprinting using only packet length information. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pages 1–6, 2019.

[101] Rebekah Overdorf, Mark Juarez, Gunes Acar, Rachel Greenstadt, and Claudia Diaz. How unique is your. onion? an analysis of the fingerprintability of tor onion services. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2021–2036, 2017.

[102] Shuai Li, Huajun Guo, and Nicholas Hopper. Measuring information leakage in website fingerprinting attacks and defenses. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 1977–1992. ACM, 2018.

[103] Tao Wang. High precision open-world website fingerprinting. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 152–167. IEEE, 2020.

[104] Rajib Ranjan Maiti, Sandra Siby, Ragav Sridharan, and Nils Ole Tippenhauer. Link-layer device type classification on encrypted wireless traffic with cots radios. In *European Symposium on Research in Computer Security*, pages 247–264. Springer, 2017.

[105] Jonathan Muehlstein, Yehonatan Zion, Maor Bahumi, Itay Kirshenboim, Ran Dubin, Amit Dvir, and Ofir Pele. Analyzing https encrypted traffic to identify user's operating system, browser and application. In *2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pages 1–6. IEEE, 2017.

[106] Andriy Panchenko, Lukas Niessen, Andreas Zinnen, and Thomas Engel. Website fingerprinting in onion routing based anonymization networks. In *ACM Workshop on Privacy in the Electronic Society*, pages 103–114, 2011.

[107] Andriy Panchenko, Fabian Lanze, Jan Pennekamp, Thomas Engel, Andreas Zinnen, Martin Henze, and Klaus Wehrle. Website fingerprinting at internet scale. In *NDSS*, 2016.

[108] Tao Wang and Ian Goldberg. Improved website fingerprinting on tor. In *Proceedings of the 12th ACM workshop on Workshop on privacy in the electronic society*, pages 201–212. ACM, 2013.

[109] Sophon Mongkolluksamee, Vasaka Visoottiviseth, and Kensuke Fukuda. Combining communication patterns & traffic patterns to enhance mobile traffic identification performance. *Journal of Information Processing*, 24(2):247–254, 2016.

[110] Dubin Ran, Amit Dvir, Ofir Pele, and Ofer Hadar. I know what you saw last minute - encrypted http adaptive video streaming title classification. *IEEE Transactions on Information Forensics and Security*, 12(12):3039–3049, 2017.

[111] Taher Al-Shehari and Farrukh Shahzad. Improving operating system fingerprinting using machine learning techniques. *International Journal of Computer Theory and Engineering*, 6(1):57, 2014.

[112] Diogo Barradas, Nuno Santos, and Luis Rodrigues. Effective detection of multimedia protocol tunneling using machine learning. pages 169–185, 2018.

[113] Meng Shen, Yiting Liu, Liehuang Zhu, Xiaojiang Du, and Jiankun Hu. Fine-grained webpage fingerprinting using only packet length information of encrypted traffic. *IEEE Transactions on Information Forensics and Security*, 16:2046–2059, 2021.

[114] Zafar Ayyub Qazi, Jeongkeun Lee, Tao Jin, Gowtham Bellala, Manfred R Arndt, and Guevara Noubir. Application-awareness in SDN. *acm special interest group on data communication*, 43(4):487–488, 2013.

[115] Shree Garg, Sateesh K Peddoju, and Anil K Sarje. Network-based detection of android malicious apps. *International Journal of Information Security*, 16(4):385–400, 2017.

[116] Shanshan Wang, Zhenxiang Chen, Qiben Yan, Bo Yang, Lizhi Peng, and Zhongtian Jia. A mobile malware detection method using behavior features in network traffic. *Journal of Network and Computer Applications*, 133:15–25, 2019.

[117] Fares Meghdouri, Tanja Zseby, and Félix Iglesias. Analysis of lightweight feature vectors for attack detection in network traffic. *Applied Sciences*, 8(11):2196, 2018.

[118] Arunan Sivanathan, Daniel Sherratt, Hassan Habibi Gharakheili, Adam Radford, Chamith Wijenayake, Arun Vishwanath, and Vijay Sivaraman. Characterizing and classifying iot traffic in smart cities and campuses. In *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 559–564. IEEE, 2017.

[119] Markus Miettinen, Samuel Marchal, Ibbad Hafeez, N Asokan, Ahmad-Reza Sadeghi, and Sasu Tarkoma. Iot sentinel: Automated device-type identification for security enforcement in iot. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 2177–2184. IEEE, 2017.

[120] Mustafizur R Shahid, Gregory Blanc, Zonghua Zhang, and Hervé Debar. Iot devices recognition through network traffic analysis. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 5187–5192. IEEE, 2018.

[121] Jiaqi Bao, Bechir Hamdaoui, and Weng-Keen Wong. Iot device type identification using hybrid deep learning approach for increased iot security. In *2020 International Wireless Communications and Mobile Computing (IWCMC)*, pages 565–570. IEEE, 2020.

[122] Blake Anderson and David McGrew. Os fingerprinting: New techniques and a study of information gain and obfuscation. In *2017 IEEE Conference on Communications and Network Security (CNS)*, pages 1–9. IEEE, 2017.

[123] Giorgos Dimopoulos, Ilias Leontiadis, Pere Barlet-Ros, and Konstantina Papagiannaki. Measuring video qoe from encrypted traffic. In *Proceedings of the 2016 Internet Measurement Conference*, pages 513–526, 2016.

[124] Irena Oršolić, Petra Rebernjak, Mirko Sužnjević, and Lea Skorin-Kapov. In-network QoE and KPI monitoring of mobile youtube traffic: Insights for encrypted ios flows. In *2018 14th International Conference on Network and Service Management (CNSM)*, pages 233–239. IEEE, 2018.

[125] Wubin Pan, Gaung Cheng, Hua Wu, and Yongning Tang. Towards qoe assessment of encrypted youtube adaptive video streaming in mobile networks. In *2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS)*, pages 1–6. IEEE, 2016.

[126] Irena Orsolic, Dario Pevec, Mirko Suznjevic, and Lea Skorin-Kapov. Youtube qoe estimation based on the analysis of encrypted network traffic using machine learning. In *2016 IEEE Globecom Workshops (GC Wkshps)*, pages 1–6. IEEE, 2016.

[127] Michael Seufert, Pedro Casas, Nikolas Wehner, Li Gang, and Kuang Li. Stream-based machine learning for real-time qoe analysis of encrypted video streaming traffic. In *2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, pages 76–81. IEEE, 2019.

[128] Jamie Hayes and George Danezis. k-fingerprinting: A robust scalable website fingerprinting technique. In *USENIX Security Symposium*, pages 1187–1203, 2016.

[129] Qinglong Wang, Amir Yahyavi, Bettina Kemme, and Wenbo He. I know what you did on your smartphone: Inferring app usage over encrypted data traffic. In *2015 IEEE Conference on Communications and Network Security (CNS)*, pages 433–441. IEEE, 2015.

[130] Vincent F Taylor, Riccardo Spolaor, Mauro Conti, and Ivan Martinovic. Robust smartphone app identification via encrypted network traffic analysis. *IEEE Transactions on Information Forensics and Security*, 13(1):63–78, 2018.

[131] Ryan Blake Jackson and Tracy Camp. Amazon echo security: Machine learning to classify encrypted traffic. In *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–10. IEEE, 2018.

[132] Mauro Conti, Luigi V. Mancini, Riccardo Spolaor, and Nino Vincenzo Verde. Can't you hear me knocking: Identification of user actions on android apps via traffic analysis. In *ACM Conference on Data and Application Security and Privacy*, pages 297–304, 2015.

[133] Feipeng Yan, Ming Xu, Tong Qiao, Ting Wu, Xue Yang, Ning Zheng, and Kim-Kwang Raymond Choo. Identifying wechat red packets and fund transfers via analyzing encrypted network traffic. In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 1426–1432. IEEE, 2018.

[134] Tianbo Gu, Zheng Fang, Allaukik Abhishek, Hao Fu, Pengfei Hu, and Prasant Mohapatra. Iotgaze: Iot security enforcement via wireless context analysis. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 884–893. IEEE, 2020.

[135] Guangquan Xu, Bingjiang Guo, Chunhua Su, Xi Zheng, Kaitai Liang, Duncan S Wong, and Hao Wang. Am i eclipsed? a smart detector of eclipse attacks for ethereum. *Computers & Security*, 88:101604, 2020.

[136] Yinxin Wan, Kuai Xu, Guoliang Xue, and Feng Wang. Iotargos: A multi-layer security monitoring system for internet-of-things in smart homes. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 874–883. IEEE, 2020.

[137] Gianluca Maiolini, Andrea Baiocchi, Alfonso Iacovazzi, and Antonello Rizzi. Real time identification of ssh encrypted application flows by using cluster analysis techniques. In *International Conference on Research in Networking*, pages 182–194. Springer, 2009.

[138] Xi Qin, Tongge Xu, and Chao Wang. Ddos attack detection using flow entropy and clustering technique. In *2015 11th International Conference on Computational*

Intelligence and Security (CIS), pages 412–415. IEEE, 2015.

[139] Meng Shen, Jinpeng Zhang, Ke Xu, Liehuang Zhu, Jiangchuan Liu, and Xiaojiang Du. Deepqoe: Real-time measurement of video qoe from encrypted traffic with deep learning. In *2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS)*, pages 1–10, 2020.

[140] Manuel Lopez-Martin, Belen Carro, Antonio Sanchez-Esguevillas, and Jaime Lloret. Network traffic classifier with convolutional and recurrent neural networks for internet of things. *IEEE Access*, 5:18042–18050, 2017.

[141] Zhitang Chen, Ke He, Jian Li, and Yanhui Geng. Seq2img: A sequence-to-image based approach towards ip traffic classification using convolutional neural networks. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 1271–1276. IEEE, 2017.

[142] Payap Sirinam, Nate Mathews, Mohammad Saidur Rahman, and Matthew Wright. Triplet fingerprinting: More practical and portable website fingerprinting with n-shot learning. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1131–1148, 2019.

[143] Pan Wang, Feng Ye, Xuejiao Chen, and Yi Qian. Datanet: Deep learning based encrypted network traffic classification in SDN home gateway. *IEEE Access*, 6:55380–55391, 2018.

[144] Minsoo Yeo, Y Koo, Y Yoon, T Hwang, J Ryu, J Song, and Cheolsoo Park. Flow-based malware detection using convolutional neural network. In *2018 International Conference on Information Networking (ICOIN)*, pages 910–913. IEEE, 2018.

[145] Jiayin Feng, Limin Shen, Zhen Chen, Yuying Wang, and Hui Li. A two-layer deep learning method for android malware detection using network traffic. *IEEE Access*, 8:125786–125796, 2020.

[146] Peng Yujie, Niu Weina, Zhang Xiaosong, Zhou Jie, Hao Wu, and Chen Ruidong. End-to-end android malware classification based on pure traffic images. In *2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, pages 240–245. IEEE, 2020.

[147] Meng Shen, Zhenbo Gao, Liehuang Zhu, and Ke Xu. Efficient fine-grained website fingerprinting via encrypted traffic analysis with deep learning. In *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*, pages 1–10, 2021.

[148] Shaveta Dargan, Munish Kumar, Maruthi Rohit Ayyagari, and Gulshan Kumar. A survey of deep learning and its applications: a new paradigm to machine learning. *Archives of Computational Methods in Engineering*, 27(4):1071–1092, 2020.

[149] Ying Li, Yi Huang, Richard Xu, Suranga Seneviratne, Kanchana Thilakarathna, Adriel Cheng, Darren Webb, and Guillaume Jourjon. Deep content: Unveiling video streaming content from encrypted WiFi traffic. In *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*, pages 1–8. IEEE, 2018.

[150] Xiaoyu Ji, Yushi Cheng, Wenyuan Xu, and Xinyan Zhou. User presence inference via encrypted traffic of wireless camera in smart homes. *Security and Communication Networks*, 2018, 2018.

[151] Paul Prasse, Lukáš Machlica, Tomáš Pevnỳ, Jiří Havelka, and Tobias Scheffer. Malware detection by analysing network traffic with neural networks. In *2017 IEEE Security and Privacy Workshops (SPW)*, pages 205–210. IEEE, 2017.

[152] Yi Zeng, Huaxi Gu, Wenting Wei, and Yantao Guo. $deep - full - range$: A deep learning based network encrypted traffic classification and intrusion detection framework. *IEEE Access*, 7:45182–45190, 2019.

[153] Xinlei Fan, Gaopeng Gou, Cuicui Kang, Junzheng Shi, and Gang Xiong. Identify os from encrypted traffic with tcp/ip stack fingerprinting. In *2019 IEEE 38th International Performance Computing and Communications Conference (IPCCC)*, pages 1–7. IEEE, 2019.

[154] Dmitri Bekerman, Bracha Shapira, Lior Rokach, and Ariel Bar. Unknown malware detection using network traffic classification. In *2015 IEEE Conference on Communications and Network Security (CNS)*, pages 134–142. IEEE, 2015.

[155] Petr Matoušek, Ondřej Ryšavỳ, Matěj Grégr, and Martin Vymlátil. Towards identification of operating systems from the internet traffic: Ipfix monitoring with fingerprinting and clustering. In *2014 5th International Conference on Data Communication Networking (DCNET)*, pages 1–7. IEEE, 2014.

[156] Nizar Msadek, Ridha Soua, and Thomas Engel. Iot device fingerprinting: Machine learning based encrypted traffic analysis. In *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–8. IEEE, 2019.

[157] Safia Rahmat, Quamar Niyaz, Akshay Mathur, Weiqing Sun, and Ahmad Y Javaid. Network traffic-based hybrid malware detection for smartphone and traditional networked systems. In *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pages 0322–0328. IEEE, 2019.

[158] Georgi A Ajaeiya, Nareg Adalian, Imad H Elhajj, Ayman Kayssi, and Ali Chehab. Flow-based intrusion detection system for sdn. In *2017 IEEE Symposium on Computers and Communications (ISCC)*, pages 787–793. IEEE, 2017.

[159] Bruhadeshwar Bezawada, Maalvika Bachani, Jordan Peterson, Hossein Shirazi, Indrakshi Ray, and Indrajit Ray. Behavioral fingerprinting of iot devices. New York, NY, USA, 2018. Association for Computing Machinery.

[160] Sandhya Aneja, Nagender Aneja, and Md Shohidul Islam. Iot device fingerprint using deep learning. In *2018 IEEE International Conference on Internet of Things and Intelligence System (IOTAIS)*, pages 174–179. IEEE, 2018.

[161] Sakthi Vignesh Radhakrishnan, A Selcuk Uluagac, and Raheem Beyah. Gtid: A technique for physical deviceanddevice type fingerprinting. *IEEE Transactions*

on Dependable and Secure Computing, 12(5):519–532, 2014.

[162] Shodan Search Engine. https://www.shodan.io/, 2021. [Online; accessed 16-May-2021].

[163] FOFA. https://fofa.so/, 2021. [Online; accessed 16-May-2021].

[164] ZoomEye - Cyberspace Search Engine. https://www.zoomeye.org/, 2021. [Online; accessed 16-May-2021].

[165] Gregor Maier, Fabian Schneider, and Anja Feldmann. A first look at mobile hand-held device traffic. In *International Conference on Passive and Active Network Measurement*, pages 161–170. Springer, 2010.

[166] Yubo Song, Qiang Huang, Junjie Yang, Ming Fan, Aiqun Hu, and Yu Jiang. Iot device fingerprinting for relieving pressure in the access control. In *Proceedings of the ACM Turing Celebration Conference-China*, pages 1–8, 2019.

[167] p0f v3. https://lcamtuf.coredump.cx/p0f3/.

[168] Jianbo Du, F Richard Yu, Guangyue Lu, Junxuan Wang, Jing Jiang, and Xiaoli Chu. Mec-assisted immersive vr video streaming over terahertz wireless networks: A deep reinforcement learning approach. *IEEE Internet of Things Journal*, 7(10):9517–9529, 2020.

[169] Jan Beznazwy and Amir Houmansadr. How china detects and blocks shadowsocks. In *Proceedings of the ACM Internet Measurement Conference*, pages 111–124, 2020.

[170] Michael Oche, Rafidah Md Noor, and Christopher Chembe. Multivariate statistical approach for estimating qoe of real-time multimedia applications in vehicular its network. *Computer Communications*, 104:88–107, 2017.

[171] James Nightingale, Pablo Salva-Garcia, Jose M Alcaraz Calero, and Qi Wang. 5g-qoe: Qoe modelling for ultra-hd video streaming in 5g networks. *IEEE Transactions on Broadcasting*, 64(2):621–634, 2018.

[172] Hua Wu, Xin Li, Guang Cheng, and Xiaoyan Hu. Monitoring video resolution of adaptive encrypted video traffic based on http/2 features. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 1–6. IEEE, 2021.

[173] Jianbo Du, F Richard Yu, Xiaoli Chu, Jie Feng, and Guangyue Lu. Computation offloading and resource allocation in vehicular networks based on dual-side cost minimization. *IEEE Transactions on Vehicular Technology*, 68(2):1079–1092, 2018.

[174] Zhihong Rao, Weina Niu, XiaoSong Zhang, and Hongwei Li. Tor anonymous traffic identification based on gravitational clustering. *Peer-to-Peer Networking and Applications*, 11(3):592–601, 2018.

[175] Wenbo Feng, Zheng Hong, Lifa Wu, Menglin Fu, Yihao Li, and Peihong Lin. Network protocol recognition based on convolutional neural network. *China Communications*, 17(4):125–139, 2020.

[176] Tomasz Bujlow, Tahir Riaz, and Jens Myrup Pedersen. A method for classification of network traffic based on c5. 0 machine learning algorithm. In *2012 international conference on computing, networking and communications (ICNC)*, pages 237–241. IEEE, 2012.

[177] Shane Alcock and Richard Nelson. Libprotoident: traffic classification using lightweight packet inspection. *WAND Network Research Group, Tech. Rep*, 2012.

[178] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15:3454–3469, 2020.

[179] Bo Liu, Ming Ding, Sina Shaham, Wenny Rahayu, Farhad Farokhi, and Zihuai Lin. When machine learning meets privacy: A survey and outlook. *ACM Computing Surveys (CSUR)*, 54(2):1–36, 2021.

[180] Emiliano De Cristofaro. A critical overview of privacy in machine learning. *IEEE Security and Privacy*, 19(4):19–27, 2021.

[181] Harry Chandra Tanuwidjaja, Rakyong Choi, and Kwangjo Kim. A survey on deep learning techniques for privacy-preserving. In *International Conference on Machine Learning for Cyber Security*, pages 29–46. Springer, 2019.

[182] Hao Dong, Cunqing Hua, Lingya Liu, and Wenchao Xu. Towards integrated terrestrial-satellite network via intelligent reflecting surface. In *ICC 2021-IEEE International Conference on Communications*, pages 1–6. IEEE, 2021.

[183] Meng Shen, Mingwei Wei, Liehuang Zhu, Mingzhong Wang, and Fuliang Li. Certificate-aware encrypted traffic classification using second-order markov chain. In *2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS)*, pages 1–10. IEEE, 2016.

[184] Yanjie Fu, Junming Liu, Xiaolin Li, Xinjiang Lu, Jingci Ming, Chu Guan, and Hui Xiong. Service usage analysis in mobile messaging apps: A multi-label multi-view perspective. In *IEEE International Conference on Data Mining*, pages 877–882, 2017.

[185] Marc Juarez, Mohsen Imani, Mike Perry, Claudia Diaz, and Matthew K Wright. Toward an efficient website fingerprinting defense. *European Symposium on Research in Computer Security*, pages 27–46, 2016.

[186] Tao Wang and Ian Goldberg. Walkie-talkie: An efficient defense against passive website fingerprinting attacks. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 1375–1390, Vancouver, BC, August 2017. USENIX Association.

[187] Jiajun Gong and Tao Wang. Zero-delay lightweight defenses against website fingerprinting. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 717–734, 2020.

[188] Qilei Yin, Zhuotao Liu, Qi Li, Tao Wang, Qian Wang, Chao Shen, and Yixiao Xu. Automated multi-tab website fingerprinting attack. *IEEE Transactions on Dependable and Secure Computing*, pages 1–1, 2021.

[189] Milad Nasr, Alireza Bahramali, and Amir Houmansadr. Defeating dnn-based traffic analysis systems in real-time with blind adversarial perturbations. In Michael

Bailey and Rachel Greenstadt, editors, *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pages 2705–2722. USENIX Association, 2021.

[190] Sanjit Bhat, David Lu, Albert Kwon, and Srinivas Devadas. Var-cnn and dynaflow: Improved attacks and defenses for website fingerprinting. *CoRR*, abs/1802.10215, 2018.

[191] Tao Wang. The one-page setting: A higher standard for evaluating website fingerprinting defenses. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 2794–2806, New York, NY, USA, 2021. Association for Computing Machinery.

[192] Andriy Panchenko, Lukas Niessen, Andreas Zinnen, and Thomas Engel. Website fingerprinting in onion routing based anonymization networks. In Yan Chen and Jaideep Vaidya, editors, *Proceedings of the 10th annual ACM workshop on Privacy in the electronic society, WPES 2011, Chicago, IL, USA, October 17, 2011*, pages 103–114. ACM, 2011.

[193] Xiang Cai, Rishab Nithyanand, Tao Wang, Rob Johnson, and Ian Goldberg. A systematic approach to developing and evaluating website fingerprinting defenses. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*, pages 227–238. ACM, 2014.

[194] Meng Shen, Jinpeng Zhang, Liehuang Zhu, Ke Xu, Xiaojiang Du, and Yiting Liu. Encrypted traffic classification of decentralized applications on ethereum using feature fusion. In *2019 IEEE/ACM 27th International Symposium on Quality of Service (IWQoS)*, pages 1–10. IEEE, 2019.

[195] Arash Habibi Lashkari, Andi Fitriah A Kadir, Hugo Gonzalez, Kenneth Fon Mbah, and Ali A Ghorbani. Towards a network-based framework for android malware detection and characterization. In *2017 15th Annual conference on privacy, security and trust (PST)*, pages 233–23309. IEEE, 2017.

[196] Mikhail Zolotukhin, Timo Hämäläinen, Tero Kokkonen, Antti Niemelä, and Jarmo Siltanen. Data mining approach for detection of ddos attacks utilizing ssl/tls protocol. In *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*, pages 274–285. Springer, 2015.

[197] Abdulghani Ali Ahmed, Waheb A Jabbar, Ali Safaa Sadiq, and Hiran Patel. Deep learning-based classification model for botnet attack detection. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–10, 2020.

[198] Guoning Hu and Deepak Venugopal. A malware signature extraction and detection method applied to mobile networks. In *2007 IEEE International Performance, Computing, and Communications Conference*, pages 19–26. IEEE, 2007.

[199] Ying He, Zheng Zhang, F Richard Yu, Nan Zhao, Hongxi Yin, Victor CM Leung, and Yanhua Zhang. Deep-reinforcement-learning-based optimization for cache-enabled opportunistic interference alignment wireless networks. *IEEE Transactions on Vehicular Technology*, 66(11):10433–10445, 2017.

[200] Yisroel Mirsky, Asaf Shabtai, Bracha Shapira, Yuval Elovici, and Lior Rokach. Anomaly detection for smartphone data streams. *Pervasive and Mobile Computing*, 35:83–107, 2017.

[201] Infogainattributeeval. https://weka.sourceforge.io/doc.dev/weka/attributeSelection/InfoGainAttributeEval.html.

[202] Cfssubseteval. https://weka.sourceforge.io/doc.dev/weka/attributeSelection/CfsSubsetEval.html.

[203] Earlence Fernandes, Jaeyeon Jung, and Atul Prakash. Security analysis of emerging smart home applications. In *2016 IEEE symposium on security and privacy (SP)*, pages 636–654. IEEE, 2016.

[204] Suibin Sun, Le Yu, Xiaokuan Zhang, Minhui Xue, Ren Zhou, Haojin Zhu, Shuang Hao, and Xiaodong Lin. Understanding and detecting mobile ad fraud through the lens of invalid traffic. In *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021*, pages 287–303. ACM, 2021.

[205] Ke Ye, Meng Shen, Zhenbo Gao, and Liehuang Zhu. Real-time detection of cryptocurrency mining behavior. *International Conference on Blockchain and Trustworthy Systems*, 2022.

[206] Yiming Liu, F Richard Yu, Xi Li, Hong Ji, and Victor CM Leung. Blockchain and machine learning for communications and networking systems. *IEEE Communications Surveys & Tutorials*, 22(2):1392–1431, 2020.

[207] Xuyang Jing, Zheng Yan, and Witold Pedrycz. Security data collection and data analytics in the internet: A survey. *IEEE Communications Surveys & Tutorials*, 21(1):586–618, 2018.

[208] Chuanpu Fu, Qi Li, Meng Shen, and Ke Xu. Realtime robust malicious traffic detection via frequency domain analysis. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 3431–3446, 2021.

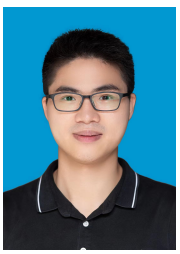
[209] Jordan Holland, Paul Schmitt, Nick Feamster, and Prateek Mittal. New directions in automated traffic analysis. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 3366–3383, 2021.

Meng Shen (M'14) is a Professor at Beijing Institute of Technology, Beijing, China. He received the B.Eng degree from Shandong University, Jinan, China in 2009, and the Ph.D. degree from Tsinghua University, Beijing, China in 2014, both in computer science. His research interests include data privacy and security, blockchain applications, and encrypted traffic classification. He has authored over 50 papers in top-level journals and conferences, such as ACM SIGCOMM, IEEE JSAC, and IEEE TIFS. He has guest edited special issues on emerging technologies for data security and privacy in IEEE Network and IEEE Internet-of-Things Journal. He received the Best Paper Runner-Up Award at IEEE IPCCC 2014 and IEEE/ACM IWQoS 2020. Dr. Shen was selected by the Beijing Nova Program 2020 and was the winner of the ACM SIGCOMM China Rising Star Award 2019. He is a member of the IEEE.

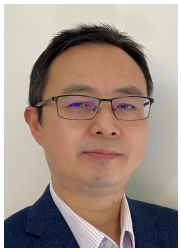
Ke Ye received the B.Eng degree in software engineering from Shandong University, Weihai, China in 2020. Currently she is a master student in the Department of Computer Science, Beijing Institute of Technology. Her research interests include Anonymity Networks and Traffic Analysis.

Xingtong Liu received the B.Eng degree in information security from Hunan University, Changsha, China in 2020. Currently she is a master student in the Department of Computer Science, Beijing Institute of Technology. Her research interests include Anonymity Networks and Traffic Analysis.

Liehuang Zhu (M'16) is a Professor in the Department of Computer Science at Beijing Institute of Technology. He is selected into the Program for New Century Excellent Talents in University from Ministry of Education, P.R. China. His research interests include Internet of Things, Cloud Computing Security, Internet and Mobile Security.



Jiawen Kang (M'18) received the Ph.D. degree from the Guangdong University of Technology, China in 2018. He was a postdoc at Nanyang Technological University, Singapore from 2018 to 2021. He currently is a professor at Guangdong University of Technology, China. His research interests mainly focus on blockchain, security, and privacy protection in wireless communications and networking.



Shui Yu (IEEE SM'12) obtained his PhD from Deakin University, Australia, in 2004. He currently is a Professor of School of Computer Science, University of Technology Sydney, Australia. Dr Yu's research interest includes Big Data, Security and Privacy, Networking, and Mathematical Modelling. He has published four monographs and edited two books, more than 400 technical papers, including top journals and top conferences, such as IEEE TPDS, TC, TIFS, TMC, TKDE, TETC, ToN, and INFOCOM. His h-index is 66. Dr Yu initiated the research field of networking for big data in 2013, and his research outputs have been widely adopted by industrial systems, such as Amazon cloud security. He is currently serving a number of prestigious editorial boards, including IEEE Communications Surveys and Tutorials (Area Editor), IEEE Communications Magazine, IEEE Internet of Things Journal, and so on. He served as a Distinguished Lecturer of IEEE Communications Society (2018-2021). He is a Distinguished Visitor of IEEE Computer Society, a voting member of IEEE ComSoc Educational Services board, and an elected member of Board of Governor of IEEE Vehicular Technology Society.

Qi Li (Senior Member, IEEE) received the Ph.D. degree from Tsinghua University. He is currently an Associate Professor with the Institute for Network Sciences and Cyberspace, Tsinghua University. He has worked with ETH Zurich and the University of Texas at San Antonio. His research interests include network and system security, particularly in Internet and cloud security, mobile security and big data security. He is currently an Editorial Board Member of the IEEE TDSC and ACM DTRAP.

Ke Xu received his Ph.D. from the Department of Computer Science & Technology of Tsinghua University, Beijing, China, where he serves as a full professor. He has published more than 200 technical papers and holds 11 US patents in the research areas of next-generation Internet, blockchain systems, Internet of Things (IoT), and network security. He is a member of ACM and senior member of IEEE. He has guest-edited several special issues in IEEE and Springer Journals. He is an editor of IEEE IoT Journal. He is also the Steering Committee Chair of IEEE/ACM IWQoS.