Mathematical Biosciences and Engineering

MBE, 20 (2): 2183–2202.
DOI: 10.3934/mbe.2023101
Received: 13 September 2022
Revised: 03 November 2022
Accepted: 09 November 2022
Published: 16 November 2022

http://www.aimspress.com/journal/MBE

Survey

A survey of methods for encrypted network traffic fingerprinting

Sunghyun Yu and Yoojae Won*

Department of Computer Science and Engineering, Chungnam National University, Daejeon 34134, Korea

*** Correspondence:** Email: yjwon@cnu.ac.kr; Tel: +82-42-821-6294; Fax: +82-42-822-4997.

Abstract: Privacy protection in computer communication is gaining attention because plaintext transmission without encryption can be eavesdropped on and intercepted. Accordingly, the use of encrypted communication protocols is on the rise, along with the number of cyberattacks exploiting them. Decryption is essential for preventing attacks, but it risks privacy infringement and incurs additional costs. Network fingerprinting techniques are among the best alternatives, but existing techniques are based on information from the TCP/IP stack. They are expected to be less effective because cloud-based and software-defined networks have ambiguous boundaries, and network configurations not dependent on existing IP address schemes increase. Herein, we investigate and analyze the Transport Layer Security (TLS) fingerprinting technique, a technology that can analyze and classify encrypted traffic without decryption while addressing the problems of existing network fingerprinting techniques. Background knowledge and analysis information for each TLS fingerprinting technique is presented herein. We discuss the pros and cons of two groups of techniques, fingerprint collection and artificial intelligence (AI)-based. Regarding fingerprint collection techniques, separate discussions on handshake messages ClientHello/ServerHello, statistics of handshake state transitions, and client responses are provided. For AI-based techniques, discussions on statistical, time series, and graph techniques according to feature engineering are presented. In addition, we discuss hybrid and miscellaneous techniques that combine fingerprint collection with AI techniques. Based on these discussions, we identify the need for a step-by-step analysis and control study of cryptographic traffic to effectively use each technique and present a blueprint.

Keywords: fingerprinting; TLS; encrypted network traffic; privacy-preserving; cybersecurity; classification; identification

1. Introduction

Various studies are being conducted on privacy protection in the field of computer communication [1–3]. It is known that when plaintext is transmitted without encryption, it can be eavesdropped on and intercepted. Hence, encrypted communication protocols are being used to protect privacy by encrypting transmission data. In 2017, Gartner predicted that more than 80% of web traffic can be encrypted [4]. According to the statistics reported as of August 2022, more than 80% of web pages were loaded as Hypertext Transfer Protocol Secure (HTTPS) over Transport Layer Security (TLS) protocols [5].

Encryption communication protocols protect transmission data using encryption algorithms. Clients and servers negotiate the encryption methods and share encryption keys before transmitting the data. After the negotiation, the data are encrypted and transmitted. This method ensures that the confidentiality of the data is maintained by making it impossible to know the content, even if a third-party attempts to eavesdrop and intercept the data. However, encryption communication protocols can be used for malicious purposes, thus posing cybersecurity issues. Cisco expected that by 2021, more than 70% of web malware would encrypt traffic, whereas 60% of organizations would fail to detect web malware traffic [6].

Attackers threaten cybersecurity by exploiting the fact that third parties are incapable of accessing the contents of transmission data when encryption communication protocols are used, as shown in Figure 1.

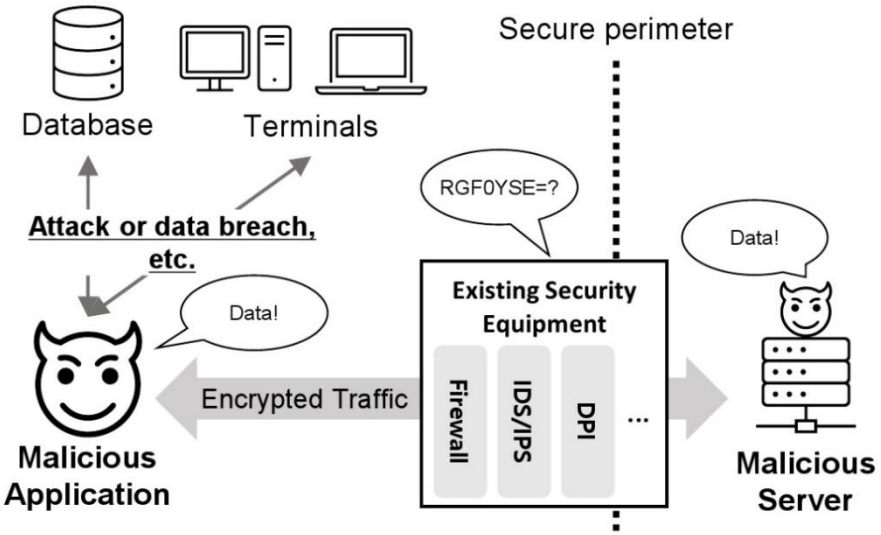


Figure 1. Example of encrypted traffic abuse by malicious applications.

In general, the network security equipment is installed in the actual network path or replicates communication packets during traffic inspection. However, the data being transmitted cannot be decrypted if the traffic is encrypted. Therefore, the network security equipment cannot check the contents and take necessary steps based on analyses. Hence, attackers exploit the limited visibility of network security equipment to conduct cyberattacks such as leakage of internal information assets or sending of triggers to malicious codes to penetrate the internal network. It is, therefore, difficult to specify the attempt and scope of the attacks that use encrypted traffic, posing a significant threat.

To manage the risk of encryption communication protocol abuse, decryption must be performed to

ensure the visibility of encrypted traffic data and for performing inspections. However, decrypting data for inspection risks privacy infringement while incurring additional computing resources and decryption time. In addition, the time delay between the client and server due to decryption and inspection increases, which can reduce the quality of the user experience. To overcome these disadvantages, it is necessary to classify and analyze the encrypted traffic without using a decryption process. To this end, network fingerprinting technology is one of the most promising candidates.

Traditional network fingerprinting techniques have been used to perform OS identification using header information in the TCP/IP stack or identify network topology using IP address information [7]. The procedure depends on the existing network environment configuration, on-premises server, and network configuration. However, with the advent of modern cloud computing and software-defined network technologies, network fingerprinting is expected to be less effective as the boundaries of the network become blurred and the configuration of services that are not dependent on traditional network address schemes increases. In addition, traffic-type classifiers utilizing traditional network fingerprinting operate based on the TCP/IP packet length [8]. The padding of encryption algorithms used in cryptographic communication protocols is expected to act as noise obstructing the input of traffic-type classifiers, thus reducing classifier accuracy. TLS fingerprinting is emerging as a method to compensate for these shortcomings.

TLS fingerprinting uses the handshake and header information of the TLS stack and encrypted application data to identify client applications and web servers and identify and classify the types of content sent. Instead of relying on IP addresses, it generates fingerprints by extracting specific information (such as encryption algorithm chutes and extensions) characterized by a TLS handshake. It also targets encrypted packets to extract features and generate classifiers to respond to encrypted traffic. This technology can be used alone and in conjunction with technology and other network fingerprinting technologies in the existing TCP/IP stack to improve accuracy. Cybersecurity, therefore, requires a broad understanding of TLS fingerprinting to respond to changing networks.

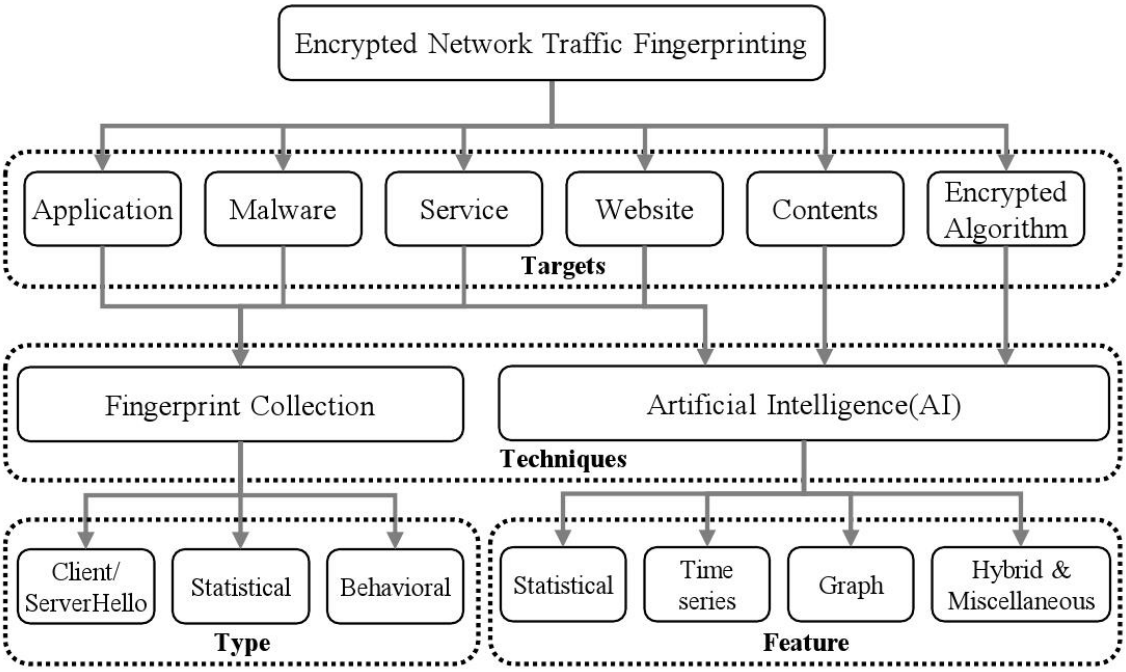


Figure 2. Taxonomy for encrypted network traffic fingerprinting.

This study provides a broad overview of encrypted network traffic fingerprinting techniques that analyze encrypted traffic without decryption. Fingerprinting is used to classify and identify client applications or servers. Fingerprints are generated through non-encrypted data from cryptographic negotiation (handshake) messages. They are then compared against fingerprint databases, or machine learning, artificial intelligence (AI), and statistical techniques are applied to a large number of datasets to generate identifiers and classifiers.

The contributions of this study are divided into two categories. First, encrypted network traffic fingerprinting techniques are investigated, and several fingerprint generation and accuracy improvement methods are described. Second, the investigation results are analyzed by organizing the identifier and classifier generation techniques. They are categorized and described based on the method and features used for identification and classification. Lastly, journals, conferences, research papers, and internet documents are surveyed. The findings are classified and described using the taxonomy shown in Figure 2.

The manuscript consists of six sections. Section 2 explains the background information required for understanding the fingerprinting techniques. Section 3 describes encrypted traffic fingerprint techniques and studies based on them. Section 4 discusses identifier and classifier generation techniques. The last section concludes the paper by presenting the results and future scope of the current research.

1.1. Related survey

Our work provides extensive information on TLS fingerprinting gained through investigation and analysis. Prior to the investigation of detailed techniques, we discuss them in relation to a survey study of techniques targeting encrypted traffic [9–12]. Survey studies have been performed for classification [9,10], detection [12], and analysis [11] as problem domains. Among them, Refs. [9–11] focus on techniques using machine learning. In addition, we refer to relevant studies [13–16] for explanations of techniques using privacy, graph networks, time series data, etc., discussed in the paper. We summarize the related survey papers below, with Table 1 providing a comparison of the problem domains, methods, and protocols of the survey papers described.

Velan et al. [9] discussed classification and analysis techniques for Internet Protocol Security, TLS, Secure Shell Protocol, BitTorrent, and Skype protocol traffic. The classification technique divided the information extraction steps into non-encrypted initialization steps and the encrypted data transfer steps into payload- and feature-based methods. Payload-based methods include techniques that perform pattern matching on payload, which is the transmission of data or operations based on information such as the payload size, port number, and IP address that can be obtained from packets without processing. Feature-based methods extract features from encrypted communication patterns and utilize maps, unsupervised methods, hybrid machine learning, and basic statistical analysis. Pacheco et al. [10] presented a study that systematically described techniques using machine learning for the classification of encrypted network traffic. Basic machine learning required for traffic classification and the representative workflows of traffic classification techniques using machine learning are discussed. Based on the workflow, an overview of the data collection, feature engineering, algorithm selection, and model deployment methods by category is provided.

Oh et al. [11] discussed methods used for analyzing malicious network traffic encrypted with TLS available at the Security Operation Center. Machine-learning-based algorithms and encryption traffic analysis techniques using middleboxes were mainly described. An overview of how TLS interception

can be performed over middlebox without a secret key or through a machine learning pipeline, for passive inspection of TLS-encrypted traffic to perform analysis by sniffing traffic and extracting features or performing malware detection using TLS flow fingerprinting was provided.

Papadogiannaki and Ioannidis [12] described cryptographic network analysis applications, technologies, and countermeasures in four use cases. These use cases consist of analytics, security, user privacy, and middleboxes. Analytics identifies protocols and users, security detects malicious traffic, user privacy detects data leakage and fingerprinting, and middlebox performs a deep packet inspection for man-in-the-middle attacks. Each case provides an overview of the application, technology, and response measures adopted. The datasets used in each study are also mentioned.

However, the researchers did not specify or describe TLS fingerprinting, which can be utilized additionally or alone in the technology stack of network fingerprinting. Existing survey papers on TLS fingerprinting provide only partial information on its use as a cryptographic traffic analysis technique, insufficient for a broad understanding of TLS fingerprinting. This paper aims to provide a broad perspective on TLS fingerprinting techniques without being specific to the purpose, technique, or data type.

Table 1. Comparison of present study with various related literature.

Survey	Problem domains	Method	Protocols
[9]	Classification	Focus on ML-based	Various
[10]	Classification	ML-based	Various
[11]	Detection	Focus on ML-based	TLS
[12]	Analysis	Various	Various
Present study	Fingerprinting	Various	TLS

2. Background

2.1. TLS

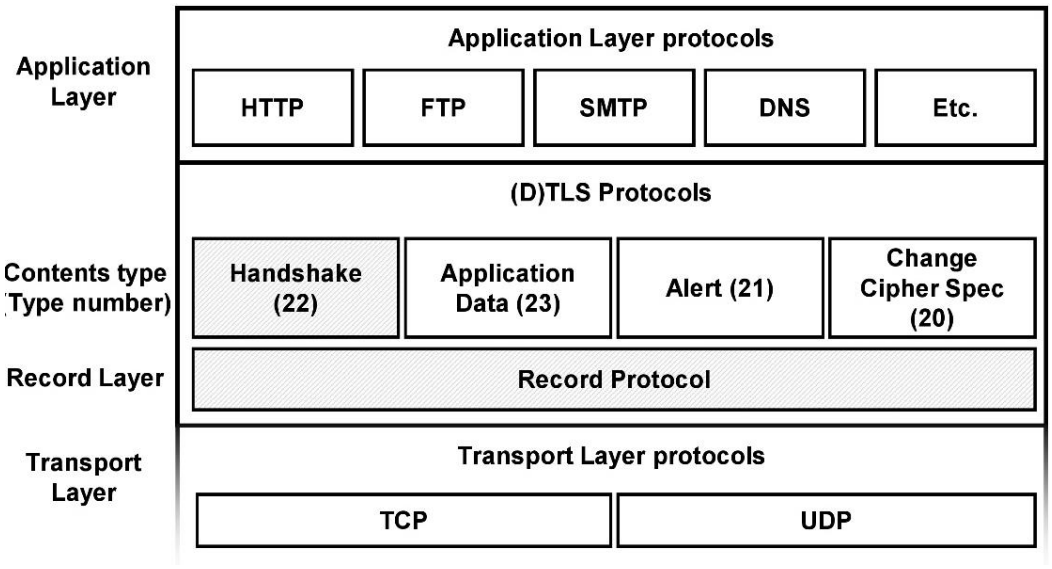


Figure 3. TLS structure.

TLS is an encryption communication protocol designed to prevent eavesdropping, tampering, and message forgery of client-server data by providing end-to-end encryption. TLS 1.0, based on Secure Socket Layer (SSL) 3.0, achieved communication privacy over the internet [17]. When TLS 1.3 was announced in March 2021, TLS 1.0 and TLS 1.1 were discontinued by the Internet Engineering Task Force (IETF) [18]. Transmission data are protected by authenticating servers and clients using a certificate and public key (asymmetric key) algorithms and performing encryption algorithm negotiation and key exchange for transmission data encryption to encrypt data after negotiation [19–22]. The TLS structure is illustrated in Figure 3. It protects the data transmitted from application layer protocols (HTTP and FTP) to transport layer protocols (TCP and UDP payload). Traffic between communication peers is protected through record protocols, and handshakes sharing encryption specifications and keys for transmission data protection are conducted.

2.1.1. TLS handshake

Key exchange, server parameter setting, and authentication are the three stages of a handshake, as shown in Figure 4. The Client/ServerHello messages sent during the key exchange phase are not encrypted. A handshake encryption key is shared to protect the messages during this phase. The handshake encryption key is used until an application encryption key is shared in the server parameters and authentication phases. The application data are encrypted using N number of application encryption keys generated through key sharing. The Client/ServerHello messages, which are non-encrypted data, are used to generate fingerprints using cipher suites and extensions.

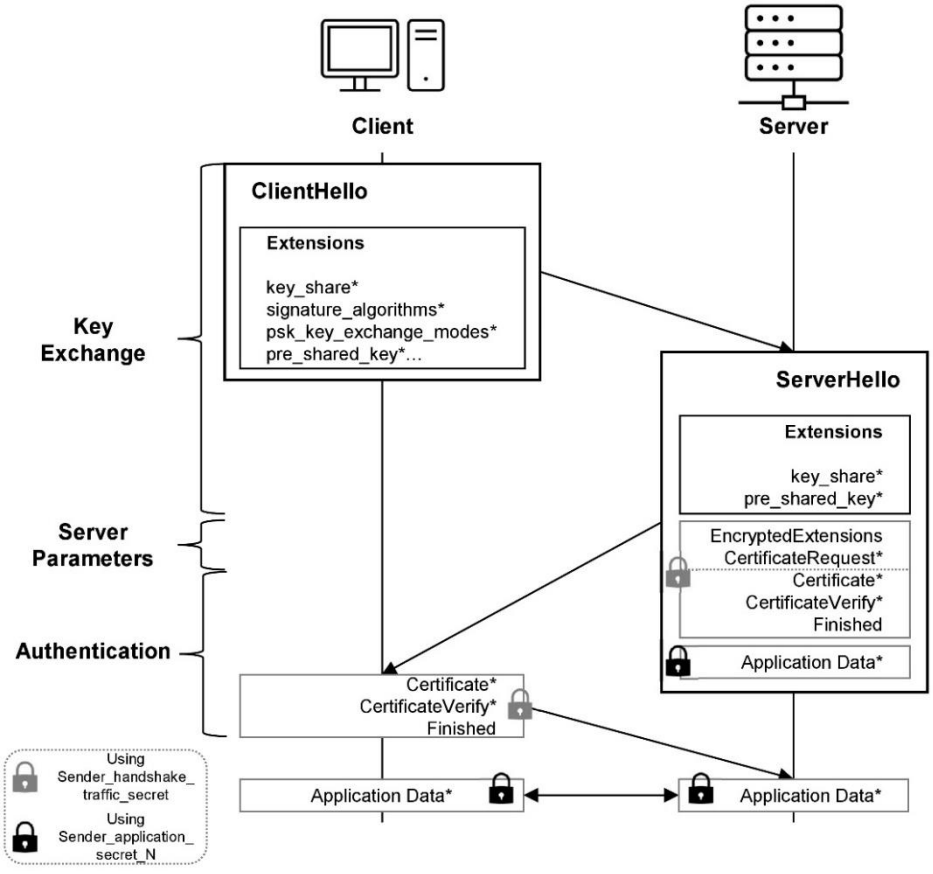


Figure 4. Basic full TLS handshake.

2.2. Markov chain

A Markov chain is a stochastic probability model with a Markov property in which the probability of a particular event depends only on the state attained in the previous event [23]. A stochastic process is a set of probability variables whose state changes stochastically over a certain period. It can be termed as a collection of values that observe the state of an object over time. The probability process can be categorized into discrete and continuous time depending on the time of observation. A Markov chain generally refers to the discrete-time Markov process [24]. If X_{n+1} is a specific state and X_n is a historical state, then the Markov chain can be expressed as

$$\lim_{x \to a^+} f(x) = \pm\infty P(X_{n+1} = j \mid X_n = i, X_{n-1} = i_{n-1}, \ldots, X_0 = i_0) = P(X_{n+1} = j \mid X_n = i). \quad (1)$$

If $P(X_{n+1} = j \mid X_n = i)$, which is a pair of (i, j), is expressed as p_{ij}, then a Markov chain with m states can be encoded as in Eq (2). The matrix is a two-dimensional transition probability matrix, where i, j, and p_{ij} represent the row, column, and transition probability elements of the matrix. Since the sum of all elements in each row is the sum of the probabilities of transition to a particular state, it is always 1.

$$\begin{pmatrix} p_{11} & \cdots & p_{1m} \\ \vdots & \ddots & \vdots \\ p_{m1} & \cdots & p_{mm} \end{pmatrix} \quad (2)$$

A Markov chain can also be represented by a state transition graph using a set of states and a transition probability matrix. For example, Figure 5 illustrates a Markov chain with $m = 2$ and $p_{ij} = 4$

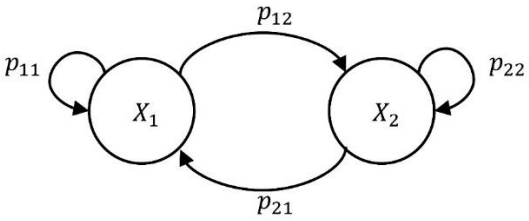


Figure 5. Example of state transition graph for a Markov chain.

Markov processes and chains are mathematical techniques for analyzing state changes and state characteristics by approaching complex probability processes with simple assumptions using a set of states and transition probability matrices [25,26]. The state transition graph of a Markov chain can be used to generate fingerprints based on changes in the message type and state of encrypted communication traffic.

2.3. AI

AI is a field of study based on the speculation that machines can be used to simulate all aspects of learning and other features of intelligence with accuracy as its principle [27]. AI, which aims to simulate intelligent human behavior, is widely used in the field of data engineering, power demand forecasting, etc. [28]. It aims for an accurate prediction in the form of machine learning systems.

Machine learning extracts features from a large number of data and then learns them into artificial neural networks to perform classification and regression on future inputs [29]. With the development of deep learning techniques that combine human neural networks with deep neural networks, research in various fields and use cases have emerged [30]. It is used to analyze encrypted traffic, such as extracting features from encrypted communication traffic, classifying positive/malicious software traffic, or identifying client applications and services.

3. Fingerprinting based on fingerprint collection

A technique based on fingerprint collection generates fingerprints for encrypted traffic, stores them in a database, and performs fingerprinting by complete or approximate matching. Fingerprints are generated based on Client/ServerHello messages, statistical techniques, and behavior such as responding to a request message from a particular sequence. They are represented by strings, data formats (XML, JSON), state transition graphs, etc. Fingerprints that are forged, altered, or unregistered have the disadvantage of poor accuracy, as the technique relies on fingerprint information. Research on improving accuracy using statistics and AI techniques is being conducted.

3.1. Technique based on ClientHello/ServerHello

This technique generates fingerprints by extracting values from ClientHello/ServerHello and messages sent and received during the key exchange phase. It is primarily a technique that identifies the processes of a client and the services provided by a server and compares them to a prebuilt fingerprint database used for fingerprinting.

Table 2. Comparison of fingerprinting technique based on ClientHello field.

ClientHello field	[31]	[32]	[33]	[34]	[35]
SSL/TLS record version			√		√
SSL/TLS version		√	√	√	√
Cipher suites length			√		
Cipher suites	√	√	√	√	√
Extensions		√	√	√	√
Data of extensions					√
Flag		√			
Compression length			√		
Compression			√		
Server name*			√		
Elliptic curves*				√	
Elliptic curve point formats*				√	

Ristic [31] proposed a technique that distinguished client processes, taking advantage of the fact that the list of supported encryption algorithms on the server differed depending on the client using the SSL. Subsequently, several studies have proposed fingerprint generation methods using various ClientHello fields based on cipher suits [32–35]. The fields used are presented in Table 2.

Cipher suites are commonly used for fingerprinting while extensions are used with SSL fingerprinting for p0f [32]. Brotherston [33] presented a demonstration of identifying real-world

processes by adding various fields. Althouse [34] discussed JA3 fingerprint, which uses extension and elliptic curve algorithm information for fingerprint generation to simplify the use field and for discrimination. Fingerprint generation codes and methods were shared as open sources to broaden the base. JA3 fingerprint is also used as pulse information in Open Threat Exchange, a threat intelligence-sharing system. An example of a JA3 fingerprint obtained using the ClientHello message is shown in Figure 6. The network flow data capturing and aliasing package named Joy by Cisco uses TLS fingerprints [35]. Similar to the JA3 fingerprint, these fingerprints do not extract the contents of a specific extension for fingerprint usage, but the data of the extension can be included in the fingerprint. Although this technique has the advantages of fast identification speed and ease of use, fingerprints are generated only in the ClientHello messages. Therefore, the technique has a disadvantage in that several processes using the same ClientHello messages may belong to one fingerprint.

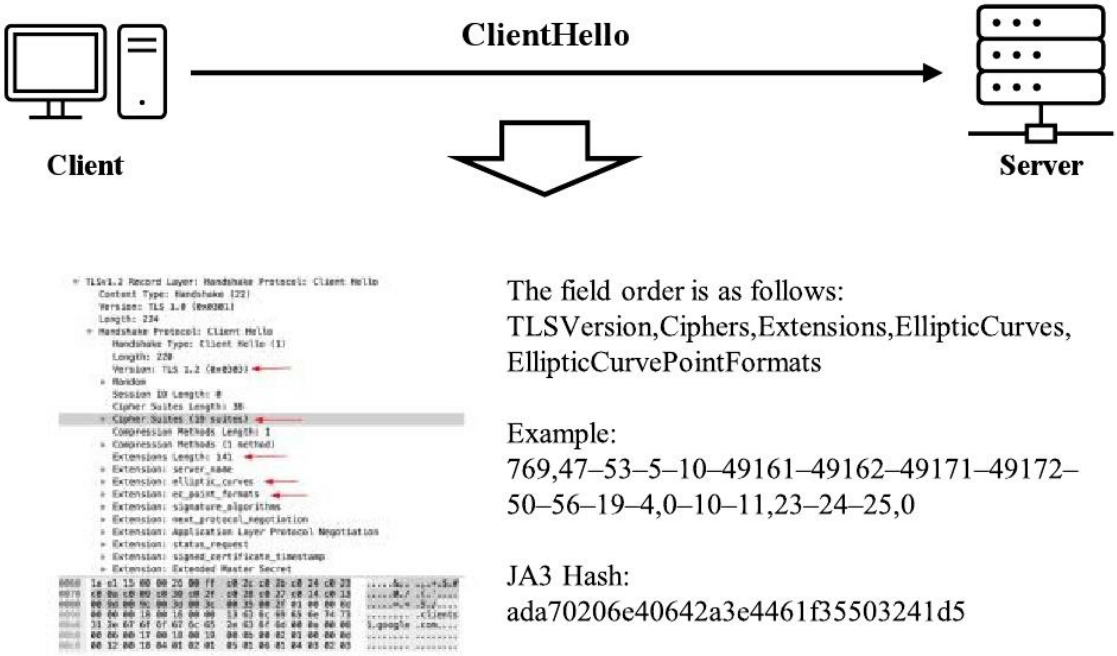


Figure 6. Example of JA3 fingerprint using ClientHello message.

Accuracy can be improved by utilizing additional techniques, such as machine learning, statistical techniques, and usage of ServerHello information, to compensate for the shortcomings. This is done because the ClientHello of the client may remain the same but the ServerHello of the servers, which are connected, may differ for different applications. JA3S uses ServerHello to generate server fingerprints, while Joy uses the cipher suit selected by the server [35,36]. JA3S with JA3 fingerprints provides additional information on connections. Thus, a classification between connections with ClientHello may be performed. Joy allows the classification between connections, but it uses different fingerprinting information such as different protocols and Operating Systems.

To improve ClientHello/ServerHello-based techniques, Anderson and McGrew [37] used knowledge bases combined with end host and network data to identify the direction of trends in enterprise TLS applications. It also enhanced the understanding of application behavior. Anderson and McGrew [38] presented a method to improve the identification and classification accuracy of ClientHello-based TLS fingerprints using destination context and pre-collected knowledge bases. For this study, a similar fingerprint group was selected through approximate matching, using the

Levenshtein distance algorithm, when no exact match for the fingerprint information was found. The matching probability was then computed using the weighted naive Bayes model learned with the destination context and knowledge bases. The process was identified as most likely.

3.2. Statistical-based

Korczynésk and Duda [39] collected all handshake connections that occurred when accessing a particular service (Paypal, Twitter, Dropbox, etc.). Probabilities for the TLS protocol versions and the handshake message occurrences were derived and classified as Markov chains with parameters. Liu et al. [40] generated features by using length Markov models in addition to message-type Markov models [39]. Classification using machine-learning-based classifiers was performed. Liu et al. [41] proposed a method to improve the classification accuracy of application traffic by performing fingerprinting using cipher suite distribution as multi-attributions in addition to the length Markov models [40]. Classification accuracy was derived for applications such as MaMPF and improved accuracy was demonstrated [40]. An example of a statistical fingerprint obtained using a Markov chain and state transition graph is shown in Figure 7.

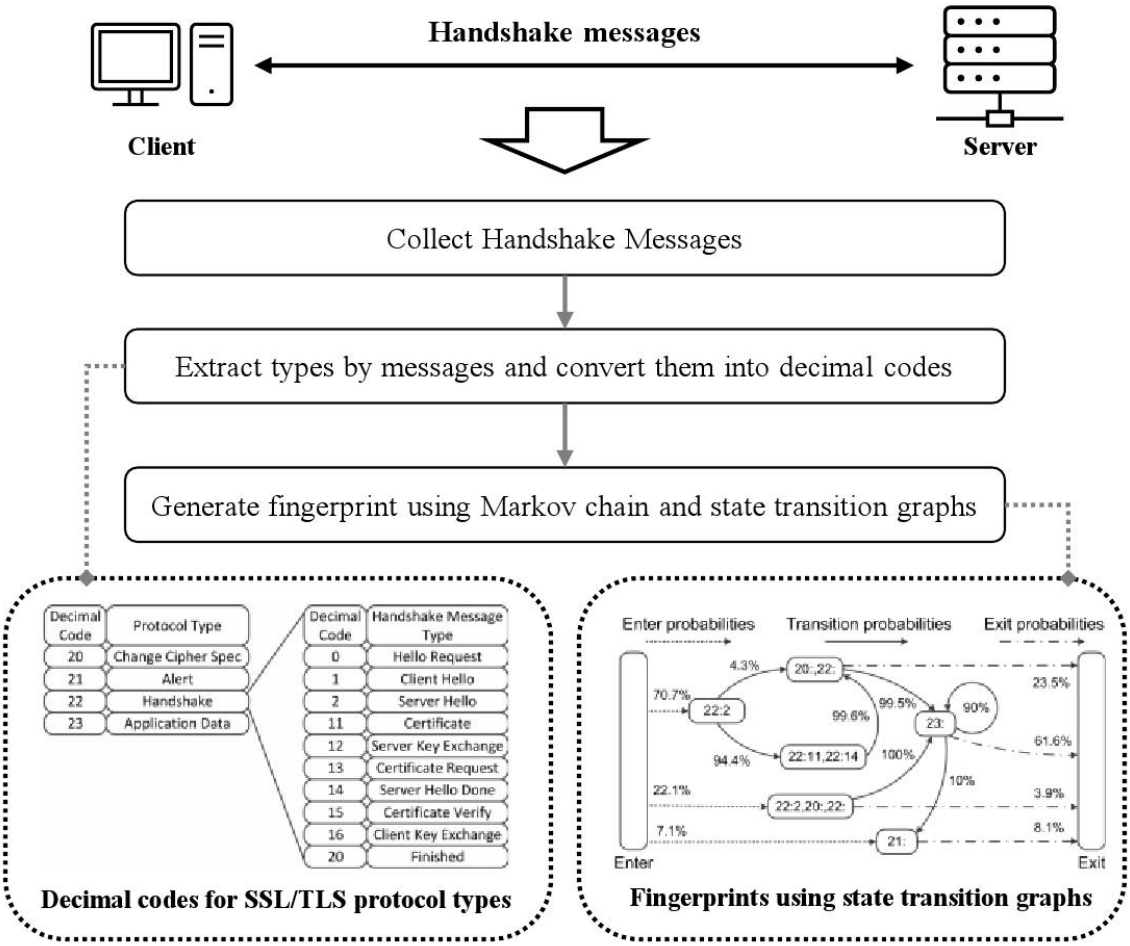


Figure 7. Example of the statistical fingerprint using a Markov chain and state transition graph.

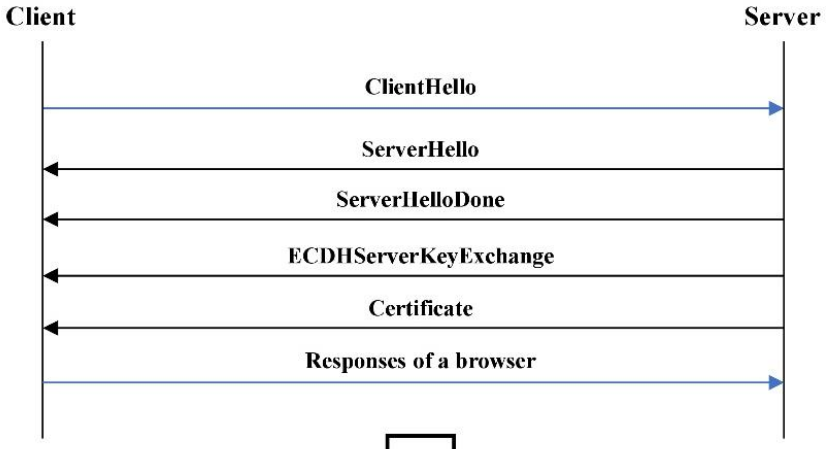
3.2.1. Enhancements of statistical-based techniques

Chao [42] conducted a study to identify malicious encrypted traffic. The fingerprint was generated

by adding a TCP handshake, SSL/TLS message type, and TCP four-way wave as fingerprint elements to the SSL/TLS handshake state transition-based fingerprint [39]. Encrypted traffic was also identified by generating a feature based on a 2-order Markov chain derived from the generated fingerprint and utilizing it for machine learning.

Zhao et al. [43] classified and identified encrypted traffic by replicating and analyzing traffic characteristics for pre-filtered traffic with header and handshake. Fingerprints based on a hidden Markov model were extracted to classify and identify applications. For this study, classification was performed on web applications, Real-time transport protocol, voice over internet protocol, and video-audio streaming media traffic.

3.3. Behavioral-based



Name	Response Value
RM1	'CLIENT_HELLO'
RM2	'AlertMessage'
AM	'UNEXPECTED_Message'
EH	' '
EPH	'Connection reset by peer: socket write error'

Figure 8. Example of behavioral fingerprint using a combination of sequences.

In 2019, Garn et al. [44] identified a web browser, during a client application, by sending a test set consisting of a specific combination of sequences from the server during the TLS handshake process. When the browser sends a ClientHello message to the server, the server, configured with the proposed framework, sends a specific combination of server-response messages. The browser transmits a response message to the server message. During this handshake process, the browser fingerprints the message sent by the client as a feature vector. In 2022, Garn et al. [45] discussed a two-step method for identifying a browser, and the ClientHello-based fingerprinting was performed while connecting to the browser. The browser was identified using the old method when no unique matching results were found [44]. An example of a behavioral fingerprint using a combination of sequences is shown in Figure 8.

4. Artificial-intelligence-based fingerprinting

AI-based techniques extract features from encrypted traffic and learn artificial neural networks to perform fingerprinting. Feature extraction utilizes statistical, time series, graph, and hybrid (or miscellaneous) techniques. Statistical analysis extracts features using statistical techniques on data obtained from the traffic, while the time series technique extracts feature through meaningful information arranged in chronological order among data obtained from the traffic. A graph extracts feature using a traffic-tracking graph. A hybrid technique uses several complex techniques and other features. In particular, the technique can identify the transmitted content and the algorithm used for encrypting transmission data.

4.1. Statistical-based

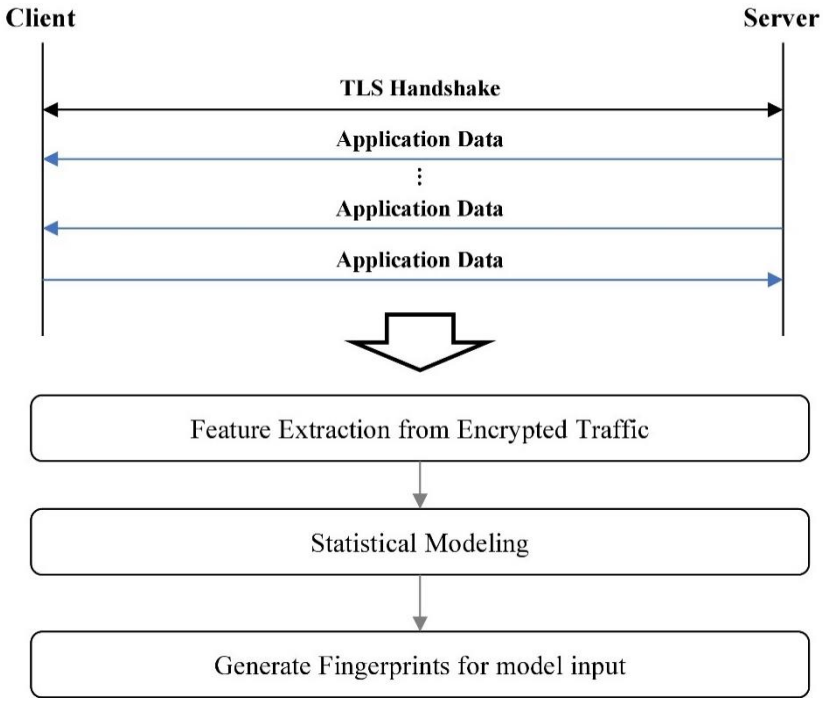


Figure 9. Example of AI-based fingerprinting using statistical-based features.

In 2017, Dubin et al. [46] proposed a method using machine learning for extracting features from video streaming traffic and for classifying the video titles uploaded to video platforms. The bit-per-peak feature, derived from the peak value in the download speed pattern during video streaming, is used. The video content transmitted through encrypted traffic was classified by a third party. Yang et al. [47] proposed a method specifying which images were viewed on performing fingerprinting with Markov chain for fragments of encrypted video traffic in addition to the previous method [46]. The state change diagram of the fragment sequence, modeled for video streaming with a specific title for the video, was uploaded as a Markov chain on YouTube, a video platform operator. Thereafter, the title of the image, viewed through the machine learning model learned with the modeled information, was a YouTube image.

Al-Naami et al. [48] extracted the burst size and time of length, downlink, and uplink from the header of the packet and configured a feature set called bi-directional dependent fingerprinting to

study the machine learning model by performing a two-way application on a mobile website.

In 2021, Kanda and Hashimoto [49] proposed a technique for identifying encryption algorithms and libraries on encrypted payloads to compensate for the shortcomings of randomizing or modifying the handshake message parameters to bypass TLS fingerprinting. Based on the test features of NIST SP 800-22 "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications," features were extracted, and the TLS version, cipher suite, and encryption algorithm were predicted [50]. An example of AI-based behavioral fingerprinting using statistical features is shown in Figure 9.

4.2. Time-series-based

Böttinger et al. [51] used the length of The TLS record protocol as a feature to train a machine learning model to generate a classifier. The classifier classified the file format to be transmitted. The Classification was performed on application (ELF), voice (MP3), document (PDF), and image (JPG, WEBM) format files. In real scenarios, it was reported that fragment offset, uncertain compressor state, and symmetric block cipher padding acted as noise.

Zhang et al. [52] proposed features that could be used in techniques for fingerprinting websites encrypted with SSL/TLS and applied the features to a Deep Forest model. The proposed local request and response sequence feature was a value derived by grouping the size and packet number of in/outgoing packets according to various criteria. The developed model was then compared with other models and f1 score.

4.3. Graph-based

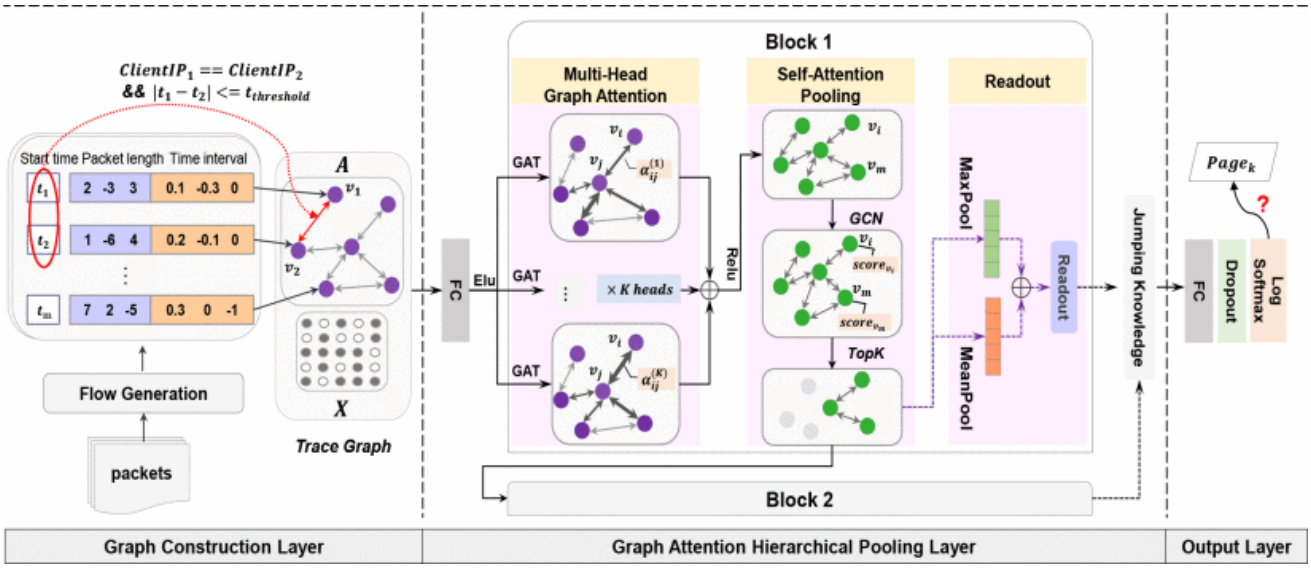


Figure 10. System overview of the GAP-WF [53].

Lu et al. [53] used traffic-tracking graphs to fingerprint websites. The traffic-tracking graph was generated by dividing a packet into flows, by referring to a collection of packets with the same IP, and by using the packet length, time interval, directional sequence, and timestamp of the first packet. Subsequently, website classification was performed using the proposed graph-based machine learning model. A system overview of the graph attention pooling network-website fingerprinting (GAP-WF)

is shown in Figure 10.

4.4. Hybrid & miscellaneous

Richter et al. [54] presented four methods using HMAC size changes based on the selected cipher suite and TLS fragmentation. Five application layer protocols (namely, Hypertext Transfer Protocol, Simple Mail Transfer Protocol, Internet Relay Chat, Post Office Protocol3, and Internet Message Access Protocol) using Bayesian classifiers were classified. The TLS fragmentation, analyzed traffic structures, packet length, and inter-arrival time used as features are shown in Figure 11. Each method was made unique by using different feature extractions.

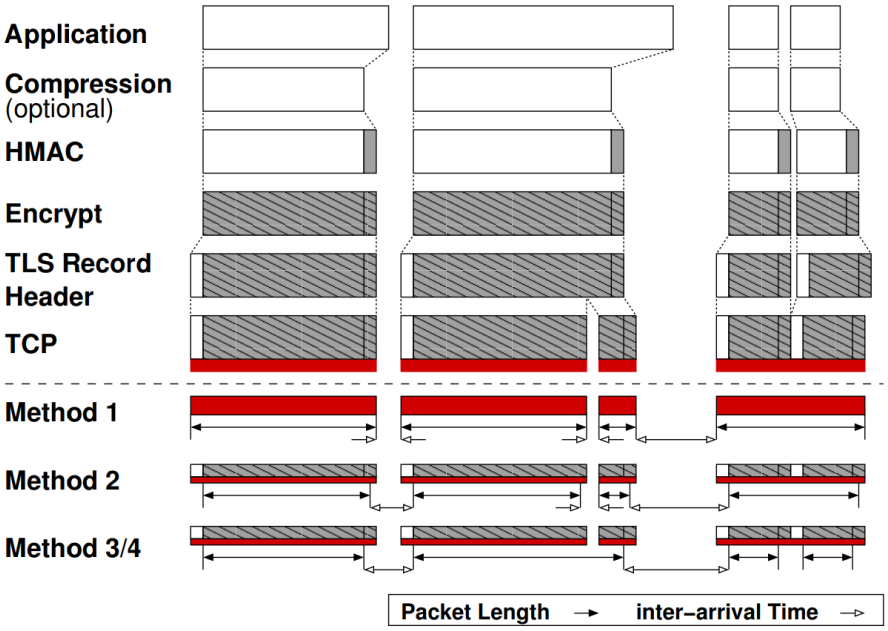


Figure 11. Differences in the use of TLS fragmentation and network traffic statistics by methods [54].

Anderson et al. [55–57] attempted to increase the accuracy of malware classification using various features. A dataset with a TLS version, offered cipher suits, TLS extension, selected cipher suite, client key length, sequence of record length, time, and type as features were constructed, and malware detection was performed [55]. Malicious traffic can be classified using a sequence of packet lengths (SPLTs), time, byte distribution (BD), with TLS, HTTP, and DNS metadata as features [56]. In the case of metadata for each protocol, data were extracted from the TLS handshake message, HTTP header, and DNS response without decryption. Malware detection was performed by learning the L1-logistic regression model. Malicious traffic can also be classified using SPLT, BD, and TLS metadata, and a custom feature server certificate was self-signed [57].

5. Discussion

Based on the techniques discussed so far, a step-by-step TLS fingerprinting technique is presented in Figure 12.

First, in encrypted traffic fingerprinting, the TLS packet between the client and server is analyzed and executed. If the packet to be analyzed is classified by time, it can be divided into the key exchange,

server parameter and authentication, and application data phases, which are termed Phase 1, 2, and 3, respectively, for convenience. Phase 1 is understood as a connection establishment attempt phase, Phase 2 is understood as a connection establishment phase, and Phase 3 is understood as a connected phase.

In Phase 1, pre-connection analysis is performed based on the Client/ServerHello fingerprints. In Phase 2, analysis is performed during the connection setup (handshake) using the metadata of the entire handshake phase. In Phase 3, analysis is performed while connected. Phase 1 has the advantage of quick analysis and pre-detection, but it is vulnerable to handshake modulation and is relatively less accurate when the analysis is performed using limited information. Phase 2 needs more information to perform high-accuracy detections and may still be vulnerable to handshake modulation. It has the disadvantage of being incapable of detecting content (e.g., insider information leakage) transmitted after being disconnected. In Phase 3, the information extracted from the application is used to detect the transmitted content based on the connection information that was analyzed until Phase 2.

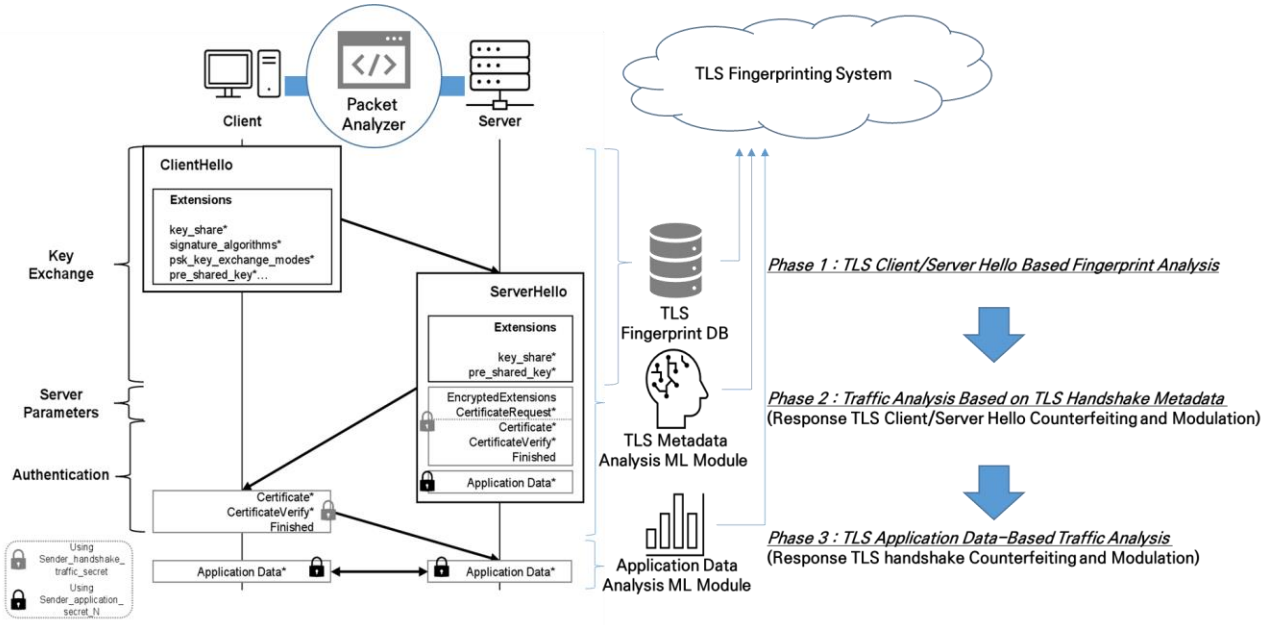


Figure 12. Future work of encrypted network traffic fingerprinting.

Detailed fingerprinting can be performed using a greater number of phases, but a significant time and computing resources are consumed as more information is used. Therefore, applying a step-by-step technology according to the policy will increase the efficiency of the fingerprinting process. For example, the application phases may be configured differently depending on the importance and type of asset. For systems where identification of "who approaches" is important, the client should be analyzed using Phases 1 and 2, and for important assets that can cause serious damage in case of leakage, all three Phases should be used to identify content and detect insider information leakage.

6. Conclusions

In this study, encrypted traffic fingerprinting is analyzed by dividing it into fingerprint collection

and AI techniques. Several advantages and disadvantages for each technique are identified.

Fingerprint collection techniques have the advantage of easy system construction when the fingerprint generation methods are clear. In addition, they are identified and classified through a fingerprint database comparison, which consumes less time and computing resources. In these techniques, pre-detection is possible because identification and classification can proceed during the handshake process. However, their accuracy is poor when fingerprints are not registered in the fingerprint database or when the client and server falsify and modulate the information used for fingerprint generation.

AI can perform calculations using various features and can infer encrypted and transmitted contents along with the encryption algorithms used. This is advantageous because it can identify leaked contents and report whether the information was leaked during detection. However, detection in advance is difficult because it can target the application layer or collect information on the entire connected traffic and then extract features and read the results.

As shown in Figure 12, a study capable of performing fingerprinting step-by-step should be conducted in the future to offset the advantages and disadvantages of each technique.

Acknowledgments

This work was supported by an Institute for Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korean government (MSIT) (No.2022-0-01200, Training Key Talents in Industrial Convergence Security).

Conflict of interest

The authors declare no conflict of interest.

References

1. T. W. Kim, A. E. Azzaoui, B. Koh, J. Kim, J. H. Park, A secret sharing-based distributed cloud system for privacy protection, *Hum. Centric Comput. Inf. Sci.*, **12** (2022). https://doi.org/10.22967/HCIS.2022.12.020
2. C. Blundo, C. De Maio, M. Parente, L. Siniscalchi, Targeted advertising that protects the privacy of social networks users, *Hum. Centric Comput. Inf. Sci.*, **11** **(**2021), 18. https://doi.org/10.22967/HCIS.2021.11.018
3. C. Jia, C. Jia, L. Kong, W. Lin, L. Qi, Privacy-aware retrieval of electronic medical records by fuzzy keyword search, *Hum. Centric Comput. Inf. Sci.*, **12** **(**2022). https://doi.org/10.22967/HCIS.2022.12.041
4. L. Orans, A. Hils, J. D'Hoinne, E. Ahlm, Gartner, Predicts 2017: Network and Gateway Security, 2016.
5. Let's encrypt stats. Available from: https://letsencrypt.org/stats/
6. Cisco encrypted traffic analytics white paper. Available from: https://www.cisco.com/c/en/us/solutions/collateral/enterprise-networks/enterprise-network-security/nb-09-encrytd-traf-anlytcs-wp-cte-en.pdf
7. F. Veysset, O. Courtay, O. Heen, New tool and technique for remote operating system fingerprinting, *Intranode Softw. Technol.*, **4** (2002).

8. L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, K. Salamatian, Traffic classification on the fly, in: *ACM Sigcomm Comput. Commun. Rev.*, **36** (2006), 23–26. https://doi.org/10.1145/1129582.1129589
9. P. Velan, M. Čermák, P. Čeleda, M. Drašar, A survey of methods for encrypted traffic classification and analysis, *Int. J. Netw. Manag.*, **25** (2015), 355–374. https://doi.org/10.1002/nem.1901
10. F. Pacheco, E. Exposito, M. Gineste, C. Baudoin, J. Aguilar, Towards the deployment of machine learning solutions in network traffic classification: A systematic survey, *IEEE Commun. Surv. Tutor.*, **23** (2018), 1988–2014. https://doi.org/10.1109/COMST.2018.2883147
11. C. Oh, J. Ha, H. Roh, A survey on TLS-encrypted malware network traffic analysis applicable to security operations centers, *Appl. Sci.*, **12** (2021), 155. https://doi.org/10.3390/app12010155
12. E. Papadogiannaki, S. Ioannidis, A survey on encrypted network traffic analysis applications, techniques, and countermeasures, *ACM Comput. Surv.*, **54** (2021), 1–35. https://doi.org/10.1145/3457904
13. H. Gao, W. Huang, T. Liu, Y. Yin, Y. Li, PPO_2: Location privacy-oriented task offloading to edge computing using reinforcement learning for intelligent autonomous transport systems, *IEEE Trans. Intell. Transport. Syst.*, (2022), 1–14. https://doi.org/10.1109/TITS.2022.3169421
14. Z. Zhang, Y. Li, H. Dong, H. Gao, Y. Jin, W. Wang, Spectral-based directed graph network for malware detection, spectral-based directed graph network for malware detection, *IEEE Trans. Netw. Sci. Eng.*, **8** (2021), 957–970. https://doi.org/10.1109/TNSE.2020.3024557
15. H. Gao, B. Qiu, R. J. Duran Barroso, W. Hussain, Y. Xu, X. Wang, TSMAE: A novel anomaly detection approach for Internet of things time series data using memory-augmented autoencoder, *IEEE Trans. Netw. Sci. Eng.*, (2022), 1–1. https://doi.org/10.1109/TNSE.2022.3163144
16. P. Li, X. Wang, H. Gao, X. Xu, M. Iqbal, K. Dahal, A dynamic and scalable user-centric route planning algorithm based on polychromatic sets theory, *IEEE Trans. Intell. Transp. Syst.*, **23** (2022), 2762–2772. https://doi.org/10.1109/TITS.2021.3085026
17. A. Freier, P. Karlton, P. Kocher, The secure sockets layer (SSL) protocol version 3.0, *The Internet Engineering Task Force, IETF*. https://www.rfc-editor.org/rfc/rfc6101.html
18. K. Moriarty, S. Farrell, Deprecating, TLS 1.0 and TLS 1.1, *Internet Engineering Task Force, IETF*. https://www.hjp.at/doc/rfc/rfc8996.html
19. R. L. Rivest, A. Shamir, L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, *Commun. ACM.*, **21** (1978), 120–126. https://doi.org/10.1145/359340.359342
20. D. Johnson, A. Menezes, S. Vanstone, The elliptic curve digital signature algorithm (ECDSA), *Int. J. Inf. Sec.*, **1** (2001), 36–63. https://doi.org/10.1007/s102070100002
21. S. Josefsson, I. Liusvaara, Edwards-curve digital signature algorithm (EdDSA), *Internet Engineering Task Force, IETF*, (2017), No. rfc8032.
22. J. Bradley, B. Campbell, T. Lodderstedt, N. Sakimura, OAuth 2.0 mutual-TLS client authentication and certificate-bound access tokens, *Internet Engineering Task Force, IETF*, (2020), Rep. rfc8705.
23. K. L. Chung, *Markov chains*, Springer-Verlag, 1967. https://doi.org/10.1007/978-3-642-49686-8
24. D. Bertsekas, J. N. Tsitsiklis, *Introduction to probability*, Athena Scientific, 2008.
25. S. R. Eddy, What is a hidden Markov model?, *Nat. Biotechnol.*, **22** (2004), 1315–1316. https://doi.org/10.1038/nbt1004-1315

26. L. Gong, X. Gong, Y. Liang, B. Zhang, A. Y. Yang, Life prediction of hydraulic concrete based on grey residual markov model, *J. Inf. Process. Syst.*, **18** (2022), 457–469. https://doi.org/10.3745/JIPS.04.0247
27. S. Dick, Artificial intelligence, *Harv. Data Sci. Rev.*, **1** (2019). https://doi.org/10.1162/99608f92.92fe150c
28. H. Yoon, S. Jeong, Electric power demand prediction using deep learning model with temperature data, *KIPS transactions on software and data engineering*, **11** (2022), 307–314. https://doi.org/10.3745/KTSDE.2022.11.7.307
29. H. Wang, Z. Lei, X. Zhang, B. Zhou, J. Peng, Machine learning basics, *Deep Learn*, (2016), 98–164.
30. M. I. Jordan, T. M. Mitchell, Machine learning: Trends, perspectives, and prospects, *Science*, **349** (2015), 255–260. https://doi.org/10.1126/science.aaa8415
31. I. Ristic, HTTP client fingerprinting using SSL handshake analysis, 2009. Available from: http://www.ssllabs.com/projects/client-fingerprinting
32. M. Majkowski, SSL fingerprinting for p0f, 2012. Available from: https://idea.popcount.org/2012-06-17-ssl-fingerprinting-for-p0f
33. L. Brotherston, GitHub, FingerprinTLS, 2015. Available from: http://github.com/LeeBrotherston/tlsfingerprinting
34. J. Althouse, S. Engineering, J.A. Open Sourcing, 2017. Available from: http://engineering.salesforce.com/open-sourcing-ja3-92c9e53c3c41
35. TLS Fingerprinting addendum, Joy: A package for capturing and analyzing network Data features, 2019. Available from: https://github.com/cisco/joy
36. J. Althouse, T.L.S. Saleforce Engineering, Fingerprinting with JA3 and JA3S, 2019. Available from: http://engineering.salesforce.com/tls-fingerprinting-with-ja3-and-ja3s
37. B. Anderson, D. McGrew, TLS beyond the browser: Combining end host and network data to understand application behavior, in: *Proceedings of the Internet Measurement Conference*, (2019), 379–392. https://doi.org/10.1145/3355369.3355601
38. B. Anderson, D. McGrew, Accurate TLS fingerprinting using destination context and knowledge bases, (2020), preprint. https://doi.org/10.48550/arXiv.2009.01939
39. M. Korczyński, A. Duda, Markov chain fingerprinting to classify encrypted traffic, in: *IEEE Conference on Computer Communications, IEEE Publications Infocom*, IEEE Publications, (2014), 781–789. https://doi.org/10.1109/INFOCOM.2014.6848005
40. C. Liu, Z. Cao, G. Xiong, G. Gou, S. M. Yiu, L. He, MaMPF: Encrypted traffic classification based on multi-attribute markov probability fingerprints, in: *26th International Symposium on Quality of Service (IWQoS), IEEE Publications/ACM*, IEEE Publications, (2018), 1–10. https://doi.org/10.1109/IWQoS.2018.8624124
41. C. Liu, G. Xiong, G. Gou, S. M. Yiu, Z. Li, Z. Tian, Classifying encrypted traffic using adaptive fingerprints with multi-level attributes, *World Wide Web*, **24** (2021), 2071–2097. https://doi.org/10.1007/s11280-021-00940-0
42. D. Chao, A fingerprint enhancement and second-order Markov chain based malicious encrypted traffic identification scheme, in: *Proceedings of the 2020 6th International Conference on Computing and Artificial Intelligence*, (2020), 328–333. https://doi.org/10.1145/3404555.3404590

43. Y. Zhao, Y. N. Yang, K. Wu, Y. Hao, H. Su, Q. Zhao, A classification and identification technology of TLS encrypted traffic applications, in: I.E.E.E. IV International (Ed.) *Conference on Big Data and Artificial Intelligence (BDAI)*, IEEE Publications, (2021), 160–164. https://doi.org/10.1109/BDAI52447.2021.9515274
44. B. Garn, D. E. Simos, S. Zauner, R. Kuhn, R. Kacker, Browser fingerprinting using combinatorial sequence testing, in: *Proceedings of the 6th Annual Symposium on Hot Topics in the Science of Security*, (2019), 1–9. https://doi.org/10.1145/3314058.3314062
45. B. Garn, S. Zauner, D. E. Simos, M. Leithner, R. Kuhn, R. Kacker, A Two-Step TLS-Based Browser fingerprinting approach using combinatorial sequences, *Comput. Secur. J.*, **114** (2022), 102575. https://doi.org/10.1016/j.cose.2021.102575
46. R. Dubin, A. Dvir, O. Pele, O. Hadar, I know what you saw last minute—Encrypted http adaptive video streaming title classification, *IEEE Trans. Inf. Forensics Secur.*, **12** (2017), 3039–3049. https://doi.org/10.1109/TIFS.2017.2730819
47. L. Yang, S. Fu, Y. Luo, J. Shi, Markov probability fingerprints: A method for identifying encrypted video traffic, in: *16th International Conference on Mobility, Sensing and Networking (MSN), IEEE Publications*, (2020), 283–290. https://doi.org/10.1109/MSN50589.2020.00055
48. K. Al-Naami, S. Chandra, A. Mustafa, L. Khan, Z. Lin, K. Hamlen, et al., Adaptive encrypted traffic fingerprinting with bi-directional dependence, in: *Proceedings of the 32nd Annual Conference on Computer Security Applications*, (2016), 177–188. https://doi.org/10.1145/2991079.2991123
49. A. Kanda, M. Hashimoto, Identification of TLS communications using randomness testing, in: *2021 IEEE Publications 45th Annual Computers, software, and Applications Conference (COMPSAC)*, 1099–1106. https://doi.org/10.1109/COMPSAC51774.2021.00150
50. L. E. Bassham III, A. L. Rukhin, J. Soto, J. R. Nechvatal, M. E. Smid, E. B. Barker, et al., *A statistical test suite for random and pseudorandom number generators for cryptographic applications*, National Institute of Standards & Technology, 800–822, 2010.
51. K. Böttinger, D. Schuster, C. Eckert, Detecting fingerprinted data in TLS traffic, in: *Proceedings of the 10th ACM Symposium on Information, Comput. Commun. Security*, (2015), 633–638. https://doi.org/10.1145/2714576.2714595
52. Z. Zhang, C. Kang, G. Xiong, Z. Li, Deep forest with LRRS feature for fine-grained website fingerprinting with encrypted SSL/TLS, in: *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, (2019), 851–860. https://doi.org/10.1145/3357384.3357993
53. J. Lu, G. Gou, M. Su, D. Song, C. Liu, C. Yang, et al., GAP-WF: Graph attention pooling network for fine-grained SSL/TLS Website fingerprinting, in: *International Joint Conference on Neural Networks (IJCNN)*, IEEE, (2021), 1–8. https://doi.org/10.1109/IJCNN52387.2021.9533543
54. C. Richter, M. Finsterbusch, J. A. Müller, K. Hänßgen, Classification of TLS applications, in: *Proceedings of the 9th International Conference on Internet Monitoring and Protection, ICIMP*, (2014), 1–6.
55. B. Anderson, Classifying encrypted traffic with TLS-aware telemetry, in: *CERT, FloCon2016*, (2016).

56. B. Anderson, D. McGrew, Identifying encrypted malware traffic with contextual flow data, in: *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security*, (2016), 35–46. https://doi.org/10.1145/2996758.2996768
57. B. Anderson, S. Paul, D. McGrew, Deciphering malware's use of TLS (without decryption), *J. Comput. Virol. Hacking Tech.*, **14** (2018), 195–211. https://doi.org/10.1007/s11416-017-0306-6

AIMS Press