Studies on Inspecting Encrypted Data: Trends and Challenges

Jongkil Kim[1*]

[1*]Assistant Professor, Department of Cyber Security, Ewha Womans University, Seoul, Republic of Korea. jongkil@ewha.ac.kr, Orcid: https://orcid.org/0000-0001-5755-108X

Received: December 26, 2022; Accepted: January 27, 2023; Published: March 30, 2023

Abstract

According to Google's transparency report, 99% of the pages loaded on the Chrome browser are over HTTPS. Encrypting the data traffic becomes almost the default requirement for not only secret data transmission but also users' privacy. However, encrypting data traffic makes us face a new threat. Particularly, conventional middlebox systems such as network intrusion detection systems and other network monitoring systems cannot effectively detect malicious activities over encrypted traffic. Due to this, building middlebox systems that can monitor the encrypted traffic become an actively researched topic. Many different middlebox systems that are based on different techniques are introduced, recently. In this paper, we present the trends and challenges of deep packet inspection (DPI) over encrypted data. Particularly, we triage them into non privacy-preserving techniques and privacy-preserving techniques. Then, we explain which inspection techniques they are based on and compared their properties. Moreover, we provide the challenge that those techniques have for future work.

Keywords: Deep Packet Inspection, Searchable Encryption, Transport Layer Security.

1 Introduction

On the Internet, the demand to preserve data privacy is growing. Encrypting the data traffic becomes almost the default requirement for data transmission. According to Google's transparency report ("HTTPS encryption on the web. https://transparencyreport.google.com/https/overview?hl= en. ,"), 99% of the pages loaded on the Chrome browser are over HTTPS (Hypertext Transfer Secure) and 95% of data traffic across Google is encrypted. HTTPS is based on an end-to-end protocol called TLS (Transport layer security) which is one of the widely used security protocols providing authentication and encryption. End-to-end encryption takes an essential role to keep a user's privacy as only the user and a server that the user connects to can decrypt the data. End-to-end encryption is helpful for enhancing the privacy of users. However, on the other side, encrypting data traffic makes adopting network-based intrusion detection systems (NIDS) difficult (Kumar, Hanumanthappa, & Suresh Kumar, 2012; Papadogiannaki, Tsirantonakis, & Ioannidis, 2022). The network device called a middlebox is located in the middle of the network of organizations to monitor incoming and outgoing network traffic if they have malicious patterns. For example, network-based intrusion detection systems (NIDS) or intrusion protection systems (NIPS) are located between users and web servers and prevent users to download malware. NIDS and NIPS are the most popular middle-box solutions. They monitor data traffic using known attack patterns, also called detection signatures or heuristic techniques based on machine learning (Umer, Junejo, Jilani, & Mathur, 2022; Zhang et al., 2022). Encryption of the data basically makes these

Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA), volume: 14, number: 1 (March), pp. 189-199 DOI: *10.58346/JOWUA.2023.I1.015*

*Corresponding author: Jongkil Kim, Assistant Professor, Department of Cyber Security, Ewha Womans University, Seoul, Republic of Korea.

middlebox systems blind and, particularly, applying the signature-based detection to NIDS/NIPS difficult.

However, middlebox systems are still important security controls for organization security as it efficiently protects users in the organization from attack actors and prevents unauthorized data leakage in the central location. Due to this, the middlebox technique called Deep Packet Inspection (DPI) is actively researched to enable a middlebox to inspect the encrypted data traffic and match known detection signatures over them. Particularly, most of the techniques target the TLS protocol which is one of the most widely used end-to-end encryption protocols. It is considered a de facto standard for network traffic encryption and is used for not only web applications but also many other applications. Various DPI techniques were introduced to break this TLS protocol in the middle of the network. Some of them do not support user privacy as they need to decrypt the encrypted data traffic totally by middlebox so that the middlebox sees all private data. However, the others are privacy-preserving as they do not compromise user privacy. They allow middleboxes to see only the malicious data but match the detection signatures over the encrypted data. This paper provides the recent trends of the research on deep packet inspection techniques and their challenges. We explore the techniques that can be utilized in the middlebox, particularly, over encrypted data traffic. In this paper, we triage the DPI techniques into two categories, non-privacy-preserving DPI techniques, which need decryption of the encrypted data traffic for inspection and privacy-preserving DPI techniques that allow the detection of malicious patterns without decryption.

2 System Architecture

We explain the system architecture of middlebox systems, which is assumed to be used for deep packet inspection. The system, in which the middlebox is located, has two users who communicate with each other, we call them a sender and a receiver. It is not really necessary to differentiate them, but we set the user who requests the connection first as the sender. The sender and the receiver use an encrypted channel for the communication that the middlebox system wants to monitor. Also, the middlebox system is located between them and probably inside the organization network. Therefore, at least one user, either a sender or a receiver, is in the same network where the middlebox is located.

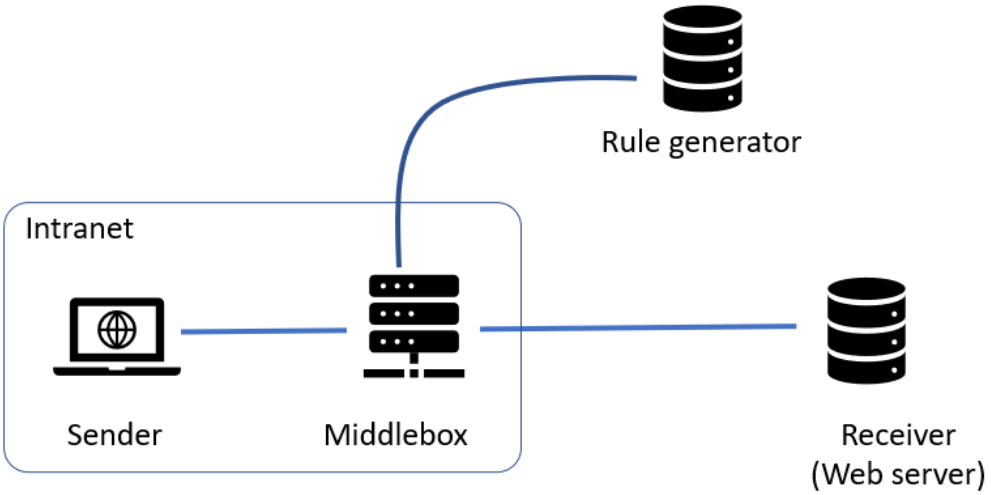

Figure 1: System Architecture of Middlebox Systems

The middlebox system has detection rules (i.e., detection signatures) which are considered malicious patterns that are collected from the set of malware programs and previous cyber security incidents. One of the well-known detection rule providers is Emerging Threats ("Proofpoint Emerging Threats Rules. https://rules.emergingthreats.net/open/snort-2.9.0/rules/," ; "Snort. https://www.snort.org/downloads,").

In this system, a rule generator is a system located outside of the organization. Therefore, it is separated from a middlebox, which is in the network of the organization. To generate the set of up-to-date detection rules, a rule generator collects the detection rules via threat intelligence and threat hunting and it regularly reviews and updates findings to its own detection rule set. It, then, feeds the detection rules to the middleboxes, which are under their supervision, to enable them to detect those attacks. Therefore, a rule generator is often the system located in the network of the middlebox manufacturer and requests a middlebox to maintain a network channel to patch up-to-date detection rules. We depict the general structure of the middlebox system the structure in Figure 1. In the figure, we assume that the middlebox is in the same network where the sender is located as it is more general. However, middleboxes also can be outside of the network. For example, Embark (Lan, Sherry, Popa, Ratnasamy, & Liu, 2016) assumes that a middlebox service is provided by a cloud server.

Threats

Before explaining middlebox techniques, we need to explain the threats that those middlebox systems are faced with and the assumptions that the middlebox systems are set on. One of the common threats that deep packet inspection systems have is that the users (the sender and the receiver) want to know the detection rules that the middlebox uses. This is because the users do not want to be censored while they communicate with each other. Therefore, if the sender and the receiver know what the detection rules are, they will try not to use them to avoid disconnection or alerts to the administrator.

Another threat is only for the privacy-preserving DPI techniques. In the privacy-preserving middlebox system, the rule generator and the middlebox are curious about all messages exchanged between the sender and the receiver. They want to see all messages exchanged between the users, if they can, not just the messages that are matched with the detection rules. It should be noted that the messages belonging to the detection rule set are not hidden in the middlebox as they know what the detection rules are, and which rules are matched with the messages.

Assumptions

There is an important assumption that the middlebox techniques shares. The middlebox system (for example, IDS system) assumes that at least one party of the sender and the receiver is trusted. This makes sense for the middlebox system because any middlebox system using the signature-based detection algorithm cannot work properly if both are malicious. Both malicious parties easily set up an encrypted or obfuscated channel to bypass any signature-based inspection to avoid their communication being monitored and caught. Therefore, in the middlebox system, at least one party in the monitored communication is assumed to be trusted.

Another assumption is about the rule generator. In DPI, the detection rules that the rule generator generates are trusted. If the sender and the middlebox do not trust the detection rules that a rule generator generates, there is little or no motivation to use or accept its rules.

3 Non-Privacy-Preserving Technique

In this section, we discuss on non-privacy-preserving techniques for the middlebox. The middlebox system requiring the decryption of the encrypted messages belongs to this triage. Because decryption is permitted by the middlebox, the middlebox can inspect messages between the sender and the receiver in plaintext format. Various techniques are introduced in this. SSL/TLS stripping, Intercepting proxy attack, which is known as a man-in-the-middle attack and subversion attack (Baek, Kim, & Susilo, 2020) belong to these categories.

SSL/TLS Stripping

One of the simple techniques for the middlebox is using SSL/TLS stripping (Shin & Lopes, 2011; Zhao, Wang, Zhao, Yang, & Ma, 2012). This technique enforces a sender not to use the encrypted channel. That means, when the sender requests an HTTPS connection, the middlebox intercepts it, pretends to be the receiver, and responds to the sender to downgrade HTTPS to HTTP or use very weak a cipher suite, which a middlebox can break. At the same time, the middlebox copies the request and establishes a connection to the receiver that the sender wants to communicate with to forward the data packets in the middle. This method is possible because 1) the sender (i.e., the client) does not authenticate the recipient (i.e., the server) in HTTP and 2) the middlebox is not authenticated by the server as client authentication is not mandatory in SSL/TLS. SSL/TLS stripping makes the connection between the sender and the middlebox insecure. That means that the other users and devices in the network may be able to see the sensitive information of the sender.

Intercepting Proxy

Intercepting proxy (Durumeric et al., 2017; Han, Kim, Ha, & Han, 2017; Huang, Rice, Ellingsen, & Jackson, 2014; Naylor et al., 2015) is the inspection method utilizing the intercepting proxy attacks, which are also known as the man-in-the-middle attack for the inspection. In this system, a middlebox is in the middle of the network between a sender and a receiver. It works similarly to the Internet proxy that forwards the traffic from one to the other. However, in this method, the encrypted data are first decrypted in the middlebox for inspection. They, then, are re-encrypted and forwarded to the other communicating party. Because the end-to-end encryption does not allow a proxy server to decrypt data, in this method, the middlebox is disguised as the other communicating parties to the sender and the receiver. This means that the sender wrongly identifies the intercepting proxy server as the receiver. Also, the receiver conceives the same proxy server as the sender. Therefore, two different end-to-end encryption channels will be set although the sender and the receiver think they are using a single direct connection between them.

A proxy attack is achieved by circumventing the server authentication from the client in the TLS protocol. By registering the root certificate issued by the middlebox as a trusted certificate to the client, when a client requests a connection, the middlebox intercepts the request and sends a response to the client with the certificate the middlebox creates. At the same time, it requests a connection to the server that the client originally wants to connect with instead of the client. The client wrongly authenticates the middlebox as the server as the certificates from the middlebox are signed by the trusted root certificate that is pre-registered. It, then, creates a secure channel with a middlebox instead of the server. Therefore, two secure connections, one between the sender and the middlebox and the other between the middlebox and the receiver, are set. All data are decrypted and then re-encrypted at the middlebox for inspection.

This technique requires the client to register the certificates issued by the middlebox as trusted certificates. This makes the client exposed to a security risk if the certificate chain that is established by the middlebox is improperly maintained. For example, if the private keys used in the certificates are leaked from the middlebox, then, the attack actor can set up any fake website requesting private information of the users like user credentials.

Subversion

Another non-privacy preserving attack is using the subversion technique (Baek et al., 2020). Subversion in cryptography protocol implies an attack technique to replace a legitimate implementation with a malicious one to sabotage the security of the end-to-end protocol. The research work (Baek et al., 2020) shows that the TLS protocol is also susceptible to this attack and it can be effectively compromised by the subversion attack. In their work, the *Client Hello* in the TLS handshake is used as information that enables the middlebox to derive the pre-master secret key of the TLS connection. For example, it uses a keyed hash function *H* to derive the pre-master secret in the RSA key exchange protocol, and then, it sets

$$\text{PMS} = H\,(\textit{InspKey}, \textit{Client Hello})$$

Where *InspKey* is an inspection key given by the middlebox. The middlebox has the inspection key, *InspKey* and can capture *Client Hello*. This means the middlebox also can compute the pre-master secret (PMS*)*. Subversion is executed by replacing a legitimate implementation with a modified one. However, it does not need to issue and manage certificates by the middlebox. Moreover, it is fully compatible with the TLS protocol. No extra controls or changes are needed in the TLS protocol for the inspection. However, it may need to manage inspection keys for each individual user as an inspection key for each user must be unique. Anyone who can access the inspection key can decrypt the data in the same procedure that the middlebox goes through.

4 Privacy-preserving Technique

The privacy-preserving technique is a technique that guarantees the users' privacy while they are still under the control. In DPI, data packets are matched in the detection rules and if the detection rules are matched with the data packets, then the middlebox knows which strings or bytes the data packet has. Therefore, in DPI, preserving privacy means the privacy of the data which are not matched with the detection rules. So, it restricts the capability of the middlebox by allowing it to see only the malicious patterns in the data traffic.

Using a Trusted Execution Environment

One of the possible ways to achieve privacy-preserving Deep packet inspection is by using Trusted Execution Environment (TEE), which is in-build encryption hardware in processors. The Intel SGX (Software Guard Extension) and AMD MET (Memory Encryption Technique) are examples of TEE. SGX-box (Han et al., 2017) and Endbox (Goltzsche et al., 2018) were introduced to use those TEEs for the middlebox solutions. In SGX-box, the client directly shares its session keys with the middlebox using the TEE of the middlebox and makes all matching be permitted inside TEE, which means via secure computing. Therefore, the middlebox cannot see the plaintext of the traffic and session keys as they are only located in plaintext format in the secure enclave. It should be noted that TEE in the middlebox maintains a secure communication channel with the client to share the session keys. In Endbox (Goltzsche et al., 2018), this inspection is made in the TEE of the client, and it is orchestrated by the

middlebox. Therefore, the client does not need to share the session keys with the others including the middlebox.

Middlebox Searchable Encryption

Middlebox searchable encryption is introduced by Sherry et al. (Sherry, Lan, Popa, & Ratnasamy, 2015). Searchable encryption (Fuhr & Paillier, 2007; Kamara, Papamanthou, & Roeder, 2012; Song, Wagner, & Perrig, 2000) is an encryption system that allows searching keywords over encrypted data without decrypting them. Therefore, it is naturally well-fitted in DPI. Middlebox searchable encryption is founded on searchable encryption, but it is refined to satisfy the assumptions of the middlebox and is also designed to be aligned with the system architecture of the middlebox system, which we introduced in Section 2.

Blind Box. The Blind Box (Sherry et al., 2015) was introduced by Sherry et al. Blind Box is proposed together with middlebox searchable encryption to provide the user's privacy. As aforementioned, the middlebox searchable encryption is precisely designed to protect privacy from the threats we explained in Section 2. The two important concepts that Blind Box satisfies are 1) the sender and the receiver do not know the detection rules and 2) the middlebox does not know the encryption key that the sender and the receiver use. These controversial concepts were achieved by using Yao's garbled circuits (Yao) for AES and oblivious transfer (Naor & Pinkas, 1999; Rabin, 2005). Those cryptographic tools enable the middlebox to receive the detection rules which are encrypted using the key that the sender and the receiver share without knowing which detection rules will be used for the inspection to them.

Blind Box shows very good detection performance as its detection speeds are only 20 nanoseconds for 1 token in the encrypted data and 1 rule in the encrypted detection rule. However, because it uses relatively heavy cryptographic protocols, the setup takes a long-time. Using 3,000 detection rules needs 97 seconds according to their implementation results. Therefore, it may be too slow to apply Blind Box for daily Internet activities. However, it is still useful for the connections that last for a long-time after they are established. Later, Embark (Lan et al., 2016) was introduced by implementing Blind box for a long-lasting network connection such as a gateway to a gateway and shows that the traffic can efficiently be monitored even via a cloud system in this scenario.

Blind IDS. Blind IDS (Canard, Diop, Kheir, Paindavoine, & Sabt, 2017) was introduced to hide the detection rules to the middlebox for the following reasons:

a. Detection rules are an important intellectual property for the middlebox manufacturer and a key differentiator with the competitors. Because the middlebox is located outside the manufacturer's network, that is its customer's network. The detection rules need to be protected, properly.
b. It is not necessary that the middlebox knows what is matched in the traffic. The middlebox is only needed to perform an action (for example, discontinuing the current connection) that is assigned in a detection rule. If the middlebox knows the detection rules, it also will know which strings or bytes are matched in the traffic. By hiding the detection rules from the middlebox, it achieves better privacy protection for the user.

Due to the above reasons, the detection rules are better to be hidden even inside the middlebox according to Blind IDS. The drawback of the Blind IDS system is efficiency. Although it offers better privacy protection, it requires a trade-off with detection performance. Because it uses pairing operations to achieve its security target, its detection becomes slower by many orders of magnitude.

PrivDPI. PrivDPI (Ning, Poh, Loh, Chia, & Chang, 2019) was introduced to improve the efficiency of Blind Box. It reduces the setup time of Blind Box by many orders of magnitudes using cyclic groups.

Instead of using Yao's garbled circuit, it uses a technique that is similar to the Diffie-Hellman key exchange protocol which is using a prime order group. As a result, it achieves a significant reduction in the setup time. At the same time, it shows the same detection efficiency. Therefore, it makes the Blind DPI more efficient and practical. After PrivDPI was introduced, the other middlebox systems (Kim et al., 2021; Ning et al., 2020) using groups and having similar properties, were introduced.

Pattern Matching on the Encrypted Data

Blind Box and PrivDPI show good performance in the detection process. However, if they are applied in practice, their schemes need a significant overhead caused by parsing the data, called tokenization, for the actual inspection. Their schemes are less expressive as a detection rule and a keyword in the data must be matched as a whole not units of bytes or strings. Therefore, for matching and detection, the data packets and the rules are properly tokenized.

There are two popular ways to tokenize the data. One is using denominators and the other is using a sliding window. If we explain two methods using the sentence" Alice booked this room.". In the method, if it uses the space and special characters as a denominator, the sentence may be tokenized as" Alice"," booked"," this" and" room". Therefore, if the middlebox wants to detect" book". It actually cannot detect the target keyword as it is tokenized into" booked" not" book". Therefore, to detect it, the detection rule set has to include not only" book" but also" books"," booked"," booking" and so on. This makes the size of the detection rules larger and, also, makes the entire detection process slower as there are more rules to be matched with the data traffic.

Another tokenization method is using a sliding window, if it uses a sliding window, and if the size of the window is six. The previous sentence is parsed as" alice"," lice b"," ice bo" and so on. This also increases the size of the encrypted traffic and at the same time, detecting a keyword less than the window size difficult. For example, if it wants to detect the name" book", which is less than six alphabet letters. It may have to consider all possible letters or special characters that can follow the word" book" for example" book a"," book10" and so on.

This obviously makes the detection difficult. Pattern matching encryption systems (Bkakria, Cuppens, & Cuppens, 2020; Desmoulins, Fouque, Onete, & Sanders, 2018) can reduce this burden. It provides a very flexible search as it encrypts the data per any unit that the system wants to set (for example, a bit, a byte or an alphabet letter). If we explain its encryption method using the previous example" Alice booked this room." it encrypts each letter with the order like" a"," l"," i"," c" and so on. For the detection, the middlebox will receive the tokens that can be used to detect a malicious pattern. For example, if a" book" is a keyword to detect because the ciphertext consists of the ciphertext elements of" b"," o"," o"," k" with the order, it can detect it. Due to this flexibility in search, those methods are very practical for some applications. However, the proposed methods also need a pairing computation for the detection. That means it has slower detection compared to the other schemes.

Zero-knowledge Middlebox

Zero-knowledge middlebox (Grubbs, Arun, Zhang, Bonneau, & Walfish, 2022) was introduced to apply the zero-knowledge proof to the middlebox. The zero-knowledge proof is a cryptographic system that enables the prover to make a verifier believe the truth of a certain statement without knowing any other secret except that it is true or not. Therefore, it is naturally suitable to show if the encrypted data traffic contains some malicious patterns or not without revealing its secret, which is encrypted messages. The Zero-knowledge middlebox shows that this zero-knowledge concept is actually applicable to the TLS

protocol, particularly, TLS 1.3 by creating proofs on the encrypted data. As it can directly create the proofs over the encrypted TLS traffic, it is compatible with TLS v1.3.

5 Challenge

Compatibility

In the middlebox solutions, compatibility with the TLS protocol is important. Particularly, some solutions (Baek et al., 2020; Grubbs et al., 2022; Han et al., 2017) show good compatibility with the modern Internet system. That means, it does not require any modification of the TLS protocol and at the same time, there is no extra implementation required for the web servers. This is important as the web service providers may not want their data traffic to be inspected and not be cooperative as they cannot correspond to various requirements from an individual organization.

The solutions using middlebox searchable encryption (Canard et al., 2017; Ning et al., 2019; Sherry et al., 2015) need an extra monitoring channel that is required for the inspection. Unfortunately, this is hard to be adopted in Internet servers due to the compatibility issue that we mentioned. However, it can be adopted in a system that the organization can control like the VPN system or a gateway to cloud systems.

Validation

Another challenge of using Blind Box (Sherry et al., 2015) and PrivDPI (Ning et al., 2019) is validation. Those middlebox solutions require at least one party to be trusted and this trusted party must validate data to guarantee that the data packets transmitted in an encryption channel and a monitoring channel are the same. This is needed as the malicious actor always wants to bypass any detection system and send two different data packets - one through the encryption channel and the other through the monitoring channel. Detecting this behavior is possible only by the trusted corresponding party, which is either the sender or the receiver because it is the only party that can see those packets in plaintext format. However, the challenge is caused by the fact that the trusted party is one that the middlebox wants to protect. Because the purpose of the middlebox is to prevent malicious data to be delivered to the end user. If the device of the end user has to decrypt the encrypted traffic, first to detect those malicious behaviors, it means that the malicious data from the attack actor already reaches the target. This may dilute the effects of the middlebox.

Tokenization

The other critical issue is tokenization. As we explained in subsection 4.3, tokenization causes significant overhead and, in some cases, detecting some keywords is impossible. The middlebox solutions using middlebox searchable encryption (Baek et al., 2020; Grubbs et al., 2022; Han et al., 2017) have this problem of tokenization. Pattern matching systems (Bkakria et al., 2020; Desmoulins et al., 2018) and Zero-knowledge middlebox (Grubbs et al., 2022) can remove the requirements for tokenization. However, their overall performances are much slower compared to the solutions using middlebox searchable encryption. It should be noted that non privacy-preserving middleboxes do not have this issue as they can see all data using the decryption and compare the plaintext rules to the data for detection.

Table 1: Property Comparisons of Middlebox Solutions
(PP: Privacy-preserving, RH: Rule-hiding, CP: Compatibility, VL: Validation, TK: Tokenization)

	PP (+)	RH (+)	CP (+)	VL (-)	TK (-)
SSL/TLS Stripping (Shin & Lopes, 2011; Zhao et al., 2012)	X	X	O	X	X
Intercepting Proxy (Durumeric et al., 2017; Han et al., 2017; Huang et al., 2014; Naylor et al., 2015)	X	X	O	X	X
Subversion (Baek et al., 2020)	X	X	O	X	X
TEE (Goltzsche et al., 2018; Han et al., 2017)	O	X	O	X	X
MBSE (Canard et al., 2017; Kim, Camtepe, Susilo, Nepal, & Baek, 2019; Ning et al., 2020; Ning et al., 2019; Sherry et al., 2015)	O	Δ	X	O	O
Pattern matching (Bkakria et al., 2020; Desmoulins et al., 2018)	O	Δ	X	O	X
Zero-Knowledge (Grubbs et al., 2022)	O	X	O	X	X

Lastly, we summarize the properties that each technique has and compare those properties in Table 1. Because some properties are advantageous and some are disadvantageous, we present them using (+) (for advantageous properties) and (-) (for disadvantageous properties). We mark the rule hiding property in MBSE using Δ because only some of the MBSE provide rule hiding property. Also, in the pattern matching systems, they may provide the rule hiding property as it does not need the information of plaintext detection rules in the matching algorithm, but this property is not formally proven. Therefore, it is marked as Δ, too.

6 Conclusion

Building middlebox systems that can monitor the encrypted traffic become an actively researched topic, recently. To detect malicious activities in the encrypted traffic, middlebox systems need to break an end-to-end encryption protocol like TLS in the middle. Various directions that can be used to build middlebox systems were proposed because breaking a secure protocol like TLS in the middle was one of the main interests of both security engineers and the attack actors for the long term.

In this paper, we summarize the important techniques that break secure protocols and allow deep packet inspection (DPI) over encrypted data for the middlebox. Particularly, we triage them into non privacy-preserving techniques and privacy-preserving techniques based on whether a middlebox can access the data in plaintext. Then, we explain the technical methodology that those systems use. Also, we briefly introduce the remained problems as challenges which are compatibility, validation, and tokenization. One may consider these are future work that is needed to be resolved.

Acknowledgment

This work was supported by the Ewha Womans University Research Grant of 2022.

References

[1] Baek, J., Kim, J., & Susilo, W. (2020). Inspecting TLS anytime anywhere: a new approach to TLS interception. In *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*, 116-126.

[2] Bkakria, A., Cuppens, N., & Cuppens, F. (2020). Privacy-preserving pattern matching on encrypted data. In *Advances in Cryptology–ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, Proceedings, Part II 26*, 191-220. Springer International Publishing.

[3] Canard, S., Diop, A., Kheir, N., Paindavoine, M., & Sabt, M. (2017). BlindIDS: Market-compliant and privacy-friendly intrusion detection system over encrypted traffic. *In Proceedings of the ACM on Asia Conference on Computer and Communications Security*, 561-574.

[4] Desmoulins, N., Fouque, P.A., Onete, C., & Sanders, O. (2018). Pattern matching on encrypted streams. In *Advances in Cryptology–ASIACRYPT 2018: 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, Proceedings, Part I 24*, 121-148. Springer International Publishing.

[5] Durumeric, Z., Ma, Z., Springall, D., Barnes, R., Sullivan, N., Bursztein, E., & Paxson, V. (2017). The Security Impact of HTTPS Interception. *In NDSS*.

[6] Fuhr, T., & Paillier, P. (2007). Decryptable searchable encryption. In *Provable Security: First International Conference, ProvSec 2007, Wollongong, Australia. Proceedings 1*, 228-236. Springer Berlin Heidelberg.

[7] Goltzsche, D., Rüsch, S., Nieke, M., Vaucher, S., Weichbrodt, N., Schiavoni, V., & Kapitza, R. (2018). Endbox: Scalable middlebox functions using client-side trusted execution. *In 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 386-397.

[8] Grubbs, P., Arun, A., Zhang, Y., Bonneau, J., & Walfish, M. (2022). {Zero-Knowledge} Middleboxes. *In 31st USENIX Security Symposium (USENIX Security 22)*, 4255-4272.

[9] Han, J., Kim, S., Ha, J., & Han, D. (2017). Sgx-box: Enabling visibility on encrypted traffic using a secure middlebox module. *In Proceedings of the First Asia-Pacific Workshop on Networking*, 99-105.

[10] HTTPS encryption on the web. https://transparencyreport.google.com/https/overview?hl= en

[11] Huang, L.S., Rice, A., Ellingsen, E., & Jackson, C. (2014). Analyzing forged SSL certificates in the wild. *In IEEE Symposium on Security and Privacy*, 83-97.

[12] Kamara, S., Papamanthou, C., & Roeder, T. (2012). Dynamic searchable symmetric encryption. In *Proceedings of the ACM conference on Computer and communications security*, 965-976.

[13] Kim, J., Camtepe, S., Baek, J., Susilo, W., Pieprzyk, J., & Nepal, S. (2021). P2DPI: practical and privacy-preserving deep packet inspection. *In Proceedings of the ACM Asia Conference on Computer and Communications Security*, 135-146.

[14] Kim, J., Camtepe, S., Susilo, W., Nepal, S., & Baek, J. (2019). Identity-based broadcast encryption with outsourced partial decryption for hybrid security models in edge computing. *In Proceedings of the ACM Asia Conference on Computer and Communications Security*, 55-66.

[15] Kumar, M., Hanumanthappa, M., & Suresh Kumar, T.V. (2012). Encrypted traffic and IPsec challenges for intrusion detection system. In *Proceedings of International Conference on Advances in Computing*, 721-727. Springer India.

[16] Lan, C., Sherry, J., Popa, R.A., Ratnasamy, S., & Liu, Z. (2016). Embark: Securely outsourcing middleboxes to the cloud. In *13th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 16)*, 255-273.

[17] Manipriya, S., Mala, C., & Mathew, S. (2020). A collaborative framework for traffic information in vehicular adhoc network applications. *Journal of Internet Services and Information Security*, *10*(3), 93-109.
[18] Naor, M., & Pinkas, B. (1999). Oblivious transfer with adaptive queries. *In Crypto*, *99*, 573-590.
[19] Naylor, D., Schomp, K., Varvello, M., Leontiadis, I., Blackburn, J., López, D.R., & Steenkiste, P. (2015). Multi-context TLS (mcTLS) enabling secure in-network functionality in TLS. *ACM SIGCOMM Computer Communication Review*, *45*(4), 199-212.
[20] Ning, J., Huang, X., Poh, G.S., Xu, S., Loh, J.C., Weng, J., & Deng, R.H. (2020). Pine: Enabling privacy-preserving deep packet inspection on TLS with rule-hiding and fast connection establishment. *In Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, Proceedings, Part I 25*, 3-22. Springer International Publishing.
[21] Ning, J., Poh, G.S., Loh, J.C.N., Chia, J., & Chang, E.C. (2019). PrivDPI: Privacy-preserving encrypted traffic inspection with reusable obfuscated rules. *In Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, 1657-1670.
[22] Papadogiannaki, E., Tsirantonakis, G., & Ioannidis, S. (2022). Network intrusion detection in encrypted traffic. *In IEEE Conference on Dependable and Secure Computing (DSC)*, 1-8.
[23] Proofpoint Emerging Threats Rules. https://rules.emergingthreats.net/open/snort-2.9.0/rules/.
[24] Rabin, M.O. (2005). How to exchange secrets with oblivious transfer. *Cryptology ePrint Archive*.
[25] Sherry, J., Lan, C., Popa, R.A., & Ratnasamy, S. (2015). Blindbox: Deep packet inspection over encrypted traffic. In *Proceedings of the 2015 ACM conference on special interest group on data communication*, 213-226.
[26] Shin, D., & Lopes, R. (2011). An empirical study of visual security cues to prevent the SSLstripping attack. *In Proceedings of the 27th Annual Computer Security Applications Conference*, 287-296.
[27] Snort. https://www.snort.org/downloads.
[28] Song, D.X., Wagner, D., & Perrig, A. (2000). Practical techniques for searches on encrypted data. *In Proceeding IEEE symposium on security and privacy. S&P 2000*, 44-55.
[29] Umer, M.A., Junejo, K.N., Jilani, M.T., & Mathur, A.P. (2022). Machine learning for intrusion detection in industrial control systems: Applications, challenges, and recommendations. *International Journal of Critical Infrastructure Protection*.
[30] Yao, A.C.C. (1986). How to generate and exchange secrets. *In IEEE 27th annual symposium on foundations of computer science (Sfcs)*, 162-167.
[31] Zhang, C., Jia, D., Wang, L., Wang, W., Liu, F., & Yang, A. (2022). Comparative research on network intrusion detection methods based on machine learning. *Computers & Security*.
[32] Zhao, S., Wang, D., Zhao, S., Yang, W., & Ma, C. (2012). Cookie-Proxy: a scheme to prevent SSLStrip attack. In *Information and Communications Security: 14th International Conference, ICICS 2012, Hong Kong, China. Proceedings 14*, 365-372. Springer Berlin Heidelberg.

Author Biography

Jongkil Kim received the Ph.D. degree in computer science from the University of Wollongong (UOW), Wollongong, NSW, Australia, in 2016., He is currently working as an Assistant Professor, Department of Cyber Security, Ewha Womans University. He has authored multiple publications in prestigious conferences and journals in an information security area. His main research interest is in applied cryptography and cybersecurity, including public-key cryptography and security protocols.