A Survey of Privacy-Preserving Techniques for Encrypted Traffic Inspection over Network Middleboxes

Geong Sen Poh
Dinil Mon Divakaran
Hoon Wei Lim
geongsen.poh@trustwave.com
dinil.divakaran@trustwave.com
hoonwei.lim@trustwave.com
Trustwave

Jianting Ning*
jtning88@gmail.com
School of Information Systems
Singapore Management University
Singapore, 178902

Achintya Desai*
achintya.desai@gmail.com
International Institute of Information Technology-Hyderabad
Hyderabad, India, 500032

ABSTRACT

Middleboxes in a computer network system inspect and analyse network traffic to detect malicious communications, monitor system performance and provide operational services. However, encrypted traffic, which has become increasingly prevalent, hinders the ability of middleboxes to perform such services. A common practice in addressing this issue is by employing a “Man-in-the-Middle” (MitM) approach, wherein an encrypted traffic flow between two end-points is interrupted, decrypted and analyzed by the middleboxes. The MitM approach is straightforward and is used by many organisations, but there are both practical and privacy concerns. Practically, due to the cost of the MitM appliances and the latency incurred due to the encrypt-decrypt processes, enterprises continue to seek solutions that are less costly and less compute-intensive. There has also been discussion on the many efforts required to configure MitM. Besides, MitM violates end-to-end privacy guarantee, raising privacy concerns and potential issues on compliance especially with the rising awareness on user privacy. Furthermore, some of the MitM implementations were found to be flawed. Consequently, new practical and privacy-preserving techniques that enable inspection over encrypted traffic were proposed.

We systematically examine these techniques to compare their advantages, limitations and challenges. We categorise them into four main categories by defining a framework that consist of system architectures, use cases, trust and threat models. These are searchable encryption, access control, machine learning and trusted hardware. We first discuss the man-in-the-middle approach as a baseline, then discuss in details each of them, and provide an in-depth comparisons of their advantages and limitations. By doing so we describe practical constraints, advantages and pitfalls towards adopting the techniques. Following this, we give insights on the gaps between research work and practical implementation in the industries, which leads us to the discussion on the challenges and research directions.

CCS CONCEPTS

• **Security and privacy** → **Network security**; **Security protocols**; *Public key (asymmetric) techniques*; **Symmetric cryptography and hash functions**; *Intrusion detection systems.*

*Work carried out when the authors were affiliated with the NUS-Singtel Cyber Security lab

KEYWORDS

Encrypted traffic analysis, deep packet inspection, data privacy, middleboxes

1 INTRODUCTION

Packet inspection and analysis have been used to detect, mitigate and block suspicious activities over home and enterprise networks. This is achieved by examining the headers and payloads of network traffic in real time. The devices deployed for this purpose are known as middleboxes[1]. A middlebox (MB) provides various services and is indispensable in today’s computer network infrastructure. One of the main services include deploying MB for system and user security, for example, as personal and organizational firewall, intrusion detection and prevention system, parental filter, data exfiltration detection system, forensic analytic tool, as well as malware detection system. In addition to security, it is also common to deploy an MB for performance and operational services. These include service for proxies/caches as in content distribution network (CDN), WAN optimization, protocol acceleration, access control, billing and usage monitoring, and network address translations. Compliance service is also deployed as an MB, to fulfill obligations such as the need to support lawful interception and control of illicit content and privacy.

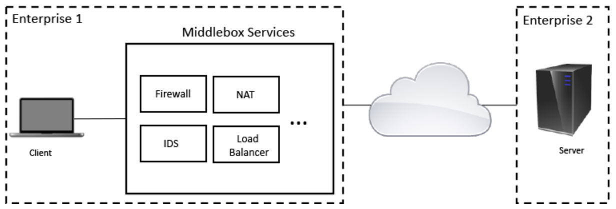

Figure 1: Traditional Model-I [Client-Oriented]: Client connects to middlebox service for in-bound and out-bound network traffic inspections

Traditionally, MBs are deployed at the premises of enterprise or customer networks. These are known as ‘on-premise’ solutions (Figures 1 and 2). With the advent of Network Function Virtualisation

[1]A middlebox, also termed as a network appliance or a network function, is defined in RFC 3234 as *any intermediary box performing functions apart from normal, standard functions of an IP router on the data path between a source host and destination host* [13]. A similar definition is also provided in Part I of ETSI proposal on middlebox security protocol and enterprise transport security (ETS) [23].

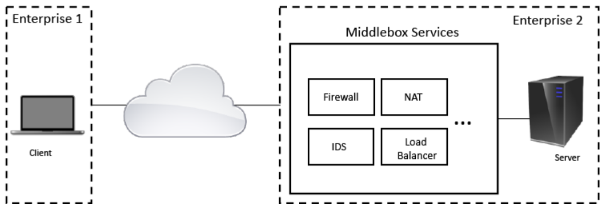

Figure 2: Traditional Model-II [Server-Oriented]: Server connects to middlebox service for in-bound and out-bound network traffic inspections

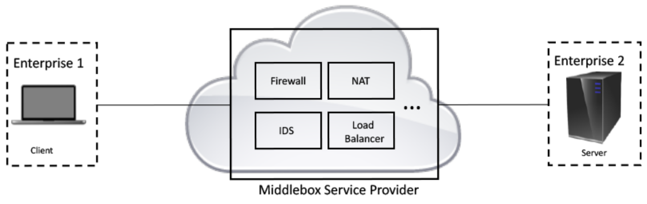

Figure 3: Outsourced Model: Client or host server subscribes to cloud-based middlebox service

(NFV) [30], the dependence on specialised and expensive hardware for deployment of MBs is also being challenged, and there is a clear shift towards deployment of software-based middlebox functions. In other words, we have started to see the outsourcing of MBs to the cloud infrastructure from the common on-premise deployment model [27, 37] (Figure 3). While cloud infrastructures provide more flexibility and dynamic scalability than a hardware-based appliance, they also come with new challenges, especially in ensuring security and privacy of these systems [8, 23].

In order to enable services mentioned above in an effective manner, middleboxes often perform network inspection and analysis using a well-established technique known as deep packet inspection (DPI)[2]. DPI implements in-depth inspection on the headers and payloads of network packets, in contrast to traditional packet filtering that inspect only packet headers. DPI can be stateful, with different useful states stored during packet processing, such as flow characteristics, application status, etc [33]. However, currently 87% – 90% of the network traffic are encrypted using TLS [15, 40]. According to Google transparancy report, as of November 2020, 81% – 98% of the traffic using Chrome platform across different operating systems are HTTPS traffic [48]. Existing DPI techniques that are only capable of inspecting plain packet properties would be of limited usage and critically impact on the various network services as was detailed by Carnavalet and Van Oorschot in [19]. This means an MB must devise mechanisms capable of analysing encrypted traffic in a manner that balances the requirements of privacy, utility and performance.

1.1 Industry Practices and New Approaches

Two common techniques widely used in the industry (in particular to deploy MBs at enterprises [10]) to inspect encrypted traffic is (1) the split-TLS technique, which is also known as a man-in-the-middle (MitM) approach, and (2) key sharing and delegation.

MitM. In MitM, instead of establishing an end-to-end TLS session between the client and the server, the client establishes a session with the middlebox. By doing so, encrypted traffic originating from the client can be decrypted, inspected and analysed by the MB. The MB re-encrypts and forwards the data to the server on behalf of the client via a second, new TLS session between the MB and the server. This provides a practical solution that can be deployed without requiring any changes to the TLS protocol, but it does require a client to install the MB's root certificate. The root certificate enables the MB to present itself as the server (i.e. the destination endpoint) to the client by copying and signing a new certificate based on the credentials of the server. In effect, the MB impersonates the server. This deployment is secure as long as the root certificate is securely stored, up-to-date TLS implementation are used, and a configurable policy engine that enables an administrator to set a whitelist[3] is provided. Unfortunately, some of the deployments have been shown to be insecure due to weaknesses in their implementation of the underlying protocols, such as allowing deprecated cipher suites, as discussed by Jarmoc [34], Carnavalet and Mannan [18], Durumeric *et al.* [22] and Waked*et al.* [58]. Also, based on the surveys conducted by Sherry *et al.* [51], a large network with heterogeneous network devices would require many experienced administrators to administer them. This poses another privacy concern when MitM approach is used, since many of these devices may have access to the decrypted data. It would be difficult to configure which are the devices that should have access and also to trace the network traffic.

Key Sharing and Delegation. In this approach, enterprises share their certificates, or private keys, and clients share session keys with the middleboxes. They are supported by industries for practical use. For example, a server sharing its private key with a CDN, or the servers share static keys with middleboxes, as detailed in the Enterprise Transport Security (ETS) standardised by the ETSI [19]. Issues with some of these approaches include not supporting Perfect Forward secrecy (PFS), and hence will not be compatible with TLS 1.3. Detailed discussion on the various approaches and issues can be found in [19].

Both the MitM and key sharing approaches violate the end-to-end encryption and data privacy guarantee of the supposedly secure two-party communications, and can be costly and difficult to implement. There is also the possibility that an MB stores the decrypted data, which opens up another set of possibilities for attack and data leak. It means the MBs and the administrator must be fully trusted in such a setting. Due to these, US-Cert has recently issued an alert stating that using HTTPS interception weakens

[2]According to the International Telecommunication Union recommendation ITU-T Y2770 [33], DPI is the *analysis, according to the layered protocol architecture OSI-BRM [specified in ITU-T X.200], of payload and/or packet properties [listed in clause 3.2.11] deeper than protocol layer 2, 3 or 4 header information, and other packet properties, in order to identify the application unambiguously.*

[3]A whitelist contains sites where the middleboxes would not inspect and merely forward the encrypted traffic. For example, encrypted traffic to online banking sites, and credit card transactions.

TLS security [17] and the National Security Agency (NSA) has also issued an advisory on potential insider threats on the use of MitM [1].

Privacy-Preserving Approaches. From the perspective of privacy, a survey on user acceptance on encrypted traffic inspection (i.e. TLS) by middleboxes was conducted by Ruoti *et al.* [50]. 1976 participants was surveyed. 75.8% express concerns about privacy and identity theft by hackers, while 70.9% are concerns about government surveillance. Furthermore, using MitM and key sharing as a tool to inspect network traffic may also violate privacy regulation such as the recently announced The EU General Data Protection Regulation (GDPR). For example, explicit consent from a consumer may be required for a third-party (e.g. the network security service provider that supplies and manages the MBs) to access the data, and 83.2% of participants in Ruoti *et al.*'s survey indicated that they should first be notified or consent.

It is due to the above concerns new approaches are proposed by the research community. Many proposals focus on the enterprise environment as discussed above (e.g. the client-oriented and server-oriented setting in Figures 1 and 2 respectively). One new approach is to perform encrypted traffic analysis (ETA) based on machine learning techniques. A comprehensive study can be found in the ENISA recent report [20]. The report surveyed six use cases that machine learning techniques are largely effective, namely application identification, network analytics, user information identification, detection of encrypted malware, fingerprinting and DNS tunnelling detection. Machine learning techniques preserve privacy as they do not inspect the (encrypted) payloads. However, it was noted that such techniques cannot offer the same level of inspection as in normal monitoring of unencrypted traffic.

On the other hand, another new approach is to devise privacy-preserving techniques to inspect encrypted payloads without the pitfalls of MitM. However, the proposed schemes require either active participation of both the client and server endpoints, or introduce accountability on both endpoints to allow visibility on all the MBs that sit between the client and the server. In practice, based on our interaction with the industries, it is not feasible to request all endpoints to install or adhere to such an architectural requirement, as was also pointed out by Carnavalet and Van Oorschot [19]. For example, detection should be performed in a way that is agnostic to the destination. A client machine, as the source endpoint, visits web application providers such as Google, Facebook, Amazon or Instagram. It would be a major undertaking for the providers to have to deploy similar solution of that of clients from different enterprises visiting their sites. Also, how would the providers trust the enterprises that request to establish connection, and what is the incentive in going through the potentially expensive customised setup cost? We believe this is one of the obstacles in adopting existing privacy-preserving proposals for actual deployment in practice. We discuss schemes using both approaches in details in Section 4.

1.2 Outsourced MB Services

The difficulty of administering the myriad devices [51] has fuelled the development of new middlebox services that departs from the traditional on-premise enterprise middlebox setting. The emerging trend is to outsource MB functionality to cloud-based services, commonly known as MB as a services. Figure 3 illustrates a general architecture of outsourced MBs. It relieves an enterprise from needing to purchase related hardware, install (both software and hardware), configure, operate and maintain middleboxes. Outsourcing of MBs therefore reduces, or virtually removes, operational costs, and thereby enables the enterprise to focus on its key business. However, outsourcing MBs to a third-party provider exacerbates privacy concern when compared to the on-premise enterprise setting, since now, data that are normally inspected internally, has to be routed to the provider for processing. Therefore, many research works have proposed to secure MB services in the cloud, including privacy-preserving inspections on encrypted traffic. In addition to this, cloud-based MB must provide low-latency operations since traffic has to be rerouted to the cloud. An example solution that enables privacy-preserving inspections with low-latency is Embark, proposed by Lan *et al.* [37]. Low-latency is achieved through an architecture known as Appliance for Outsourcing MBs (APLOMB) introduced in [52]. The general idea is to create a tokenised encrypted traffic along the TLS traffic. Searchable encryption method is used to match the tokenised encrypted traffic with the encrypted rulesets at the MB, all of which are setup under the APLOMB architecture.

In addition, many more recent outsourced-based MB proposals develop techniques that are based on secure enclave (i.e. Intel SGX). These include, for example, SafeBricks [46], ShieldBox [56] and SPlitBox [7]. The general idea of these techniques is to perform packet inspections in the secure enclave so that the cloud provider does not learn the content of the encrypted packets. We discuss in more details these proposals in Section 4.5.

1.3 Overview of Our Study

In essence, one is looking for an optimal solution between two extremes, (1) of not being able to perform in-depth inspection due to the traffic being encrypted; and (2) inspection of decrypted traffic using MitM and key sharing approach. Many solutions have been proposed in the recent years to address this issue. We categorise them into four categories based on the techniques they used. We discuss these techniques in Section 4. We also include the MitM approach as a fifth category in the beginning of the discussion for comparisons. We believe the different designs and properties of the growing list of proposals should be studied in order to clearly characterise their advantages and limitations. Our study provides insights on security assurance and performance over different use case scenarios, as well as appeals of the techniques to real-world deployment:

- We define a general architecture, identify use cases and their constraints (Section 2). We define three system models under the prescribed architecture that illustrates the flow of network traffic through the MBs. We term them client-oriented, server-oriented and client-server accountable settings. Our models provide a more fine-grained categorisation and enhancement on the one-sided and two-sided definition in the ETSI standard [23, 25].
- We define a trust model encompassing the different actors in a network system that contains MBs (Section 3). It provides a

foundation on which security model can be defined clearly. We state trust assumptions based on the architecture and settings that we defined, and define threats and security requirements based on these assumptions.

- We classify existing privacy-preserving techniques into two types, *passive* and *active*. Here, *passive* inspection encompasses techniques that analyse encrypted traffic without decryption or, modify the traffic of the underlying protocol. *Active* inspection analyses the traffic by decryption or modification of the underlying protocol. These techniques are categorised as *access control*, *searchable encryption*, *machine learning* and *trusted hardware*, as well as the MitM approach for comparison (Section 4). We compare the main features, advantages and pitfalls of each category.
- We identify the research challenges and discuss the potential research directions (Section 5). We examine the current security model and analyse the existing attacks. This leads us to believe that some of the existing proposals require further improvements for industry adoptions. For example, a protocol that utilises searchable encryption mechanisms will need to be analysed against well-established notion of information leakages in the field. We give insights, especially on the reason that existing MitM approaches, with combination of a configurable policy engine, are still preferred in the industries.

As a side note, for practical and efficient deployment with optimal inspection capability, it seems decryption of the encrypted payload is one of the better approaches. What remains to be studied is how much encrypted data should be decrypted and revealed to the MBs in order to preserve privacy. It is, we believe, based on this fact that a new MB security protocol (MSP) was proposed and is undergoing standardisation efforts [24]. It may also be the case that trusted hardware approaches, such as using Intel SGX, are being developed where encrypted traffics are decrypted in a way protected by the hardware so that MBs have no access to the decrypted data. The techniques and challenges are discussed in details in Section 4 and Section 5.

A short survey related to the issue that we studied was provided by Wang *et al.* in [59]. It studied the various mechanisms for secure outsourcing of MBs in a general manner, focusing solely on cloud-based MB. Our work complements theirs in a way that we provide broader investigation that also covers non-outsourced specific literature, hence our definition of different system models and use cases. The European Union Agency for Cybersecurity (ENISA) also published a survey on encrypted traffic analysis [20]. The survey describes in detail in particular 6 key use cases (i.e. application identification, network analytics, user information identification, detection of encrypted malware, file/device/website/location fingerprinting and DNS tunnelling detection) and techniques based on machine learning. Most recently, Carnavalet and Van Oorschot [19] presented a comprehensive survey on TLS interception mechanisms and motivations, focusing on practical considerations between the use cases and incentives of the stakeholders. They listed 19 use cases where access to unencrypted traffic is crucial, in order to understand the motivation of the various proposals in inspecting encrypted traffic. One of the key insights from their survey is the identification of gaps between the proposed mechanisms, use cases and incentives. Extensive details on the various MitM and key sharing techniques were studied and compared. Here, our focus is on privacy-preserving techniques with compilations, categorisation of the state-of-the-arts, in-depth examinations and comparisons of these various techniques, which complement these recent surveys.

2 SYSTEM MODELS AND USE CASES

We now describe the underlying models and use cases. The different models that we discuss here are based on the setting provided by Sherry *et al.* (BlindBox) [53], Canard *et al.* (BlindIDS) [18], Bhargavan *et al.* [8] and ETSI standard I [23, 25]. In order to provide practical insights, we also present use case scenarios for each models. Figure 1, 2 and 3 show a general high-level architecture of MBs operating in a computer network environment. We describe existing and potential use cases based on these architectures. We remark that it is common for MBs to act directly to the passing traffic in an *in-band* middlebox setting where one or more middleboxes are placed in-line between the client and the server. Common use cases for operational purposes include content delivery networks (CDNs), assess control, billing and usage monitoring, asset tracking, name or tag resolution and operations control. It is also possible in certain use cases that MBs store the traffic and analyse them in an *out-of-band* setting. Cyber security use cases include network firewalls, application firewalls, intrusion detection system (IDS) and intrusion prevention system (IPS). It is also used for compliance obligations, for example availability/resilience, emergency and public safety communication, data retention, identity management, cyber security, content control, personal data and privacy [23, 25].

2.1 Client-Oriented MBs

We first describe a typical client-oriented setting that is common in an enterprise network. Figure 1 illustrates a high-level model of the setting, where the goal of the middleboxes is to protect the clilent(s), and therefore are often placed closer to the client(s). Here, a client connects to a server using a secure channel. The encrypted traffic flows, both inbound and outbound, are routed through a host of MBs that inspect the traffic. MBs receive in-bound and out-bound traffic from multiple clients and server endpoints. It normally involves installing specific software, or modification on the connections at the client device in order to enable the MBs to perform the inspections.

Use Cases. Common use cases include firewalls, advertisement blocking, personal data protection and anonymisation of information [55, 62]. An example is, protection of user's privacy while enabling encrypted traffic inspection in a computer network of an organisation (e.g. a university, a financial institution or a telecommunication company) or by an outsourced middlebox service. Sherry *et al.* [53] described a scenario where students using the university network install the proposed solution to preserve privacy of their data yet allowing the system (e.g., a security solution) to inspect the encrypted traffic. Another use case, of parental filter, where a user subscribes the service from an ISP (internet service provider) is also presented. These two examples prioritises inspection of outbound traffic from the client. Bhargavan *et al.* [8] also stated that firewalls are deployed in companies and educational institutions to protect computers from malware on both inbound and outbound traffic, for example. The most relevant deployment scenario for enterprises is the client-oriented middlebox. In this scenario, an enterprise deploys an MB appliance at its premise, at the perimeter of the network. The MB inspects all incoming and outgoing traffic (except those whitelisted) from/to the clients, using the previously mentioned MitM approach. Since enterprises often enforce policies on employee machines, the installation of root certificates that allow MBs to decrypt and re-encrypt the traffic is not a practical issue.

2.2 Server-Oriented MBs

A server-oriented MB is also common in an enterprise network (Figure 2). This is to secure the servers hosted in an enterprise, for example web servers. A server in an enterprise establishes secure channels with requesting clients that are often outside the enterprise network, in particular, the Internet. The encrypted traffic to/from the server endpoint are routed through a host of MBs at the server-side (or outsourced to a third-party service provider) that inspect the encrypted traffic. The main technical difference in this setting compared to the previous one is that connections are all requested by clients, and in general the purpose is to secure one specific server. Server-oriented MBs are developed to protect servers, such as web servers, database servers, and so on. A typical example is a web application firewall that inspects incoming HTTP requests to a web server. As the web server itself has moved to the cloud since a few years now, having server-oriented MBs in the cloud is not uncommon these days [2].

Use Cases. A common scenario is where content delivery network (CDN) serves TLS traffic on behalf of customer websites. In this case, the emphasis is on inspecting the many inbound traffic flows to the websites. Another use case is provided in the ETSI standard [23, 25]. It involves a data center that would require inspection on encrypted traffic from various internal networks. This means MBs between these networks must be able to communicate and inspect the underlying encrypted traffic.

3 TRUST MODEL

At least one of the two endpoints must be honest. While deploying a traditional network security solution, such as an intrusion detection system (IDS), it is assumed that either one of the endpoints must be honest [53, 61]. Otherwise two malicious endpoints can agree on a secret key and encrypt the traffic using well-established encryption schemes. Canard *et al.* [18] described a scenario, whereby an infected bot connects to the remote command and control server using an encrypted channel. Inspection will be impossible when the traffic flows are encrypted. Similar assumption is made in the case of MBs providing parental filtering and data exfiltration detection [53]. In parental filter, it is assumed innocent child would not replace network protocol stack or install tunneling software. Data exfiltration is assumed to occur due to accidental transmission of sensitive data. In other words, the case of a user (or an adversary that gained control on the device) intentionally sabotaging a device to circumvent existing network connection is not considered in some of the proposals [12, 37, 43, 44, 53].

What if both endpoints are malicious? Given the above description, we may argue the common assumption is that either the client or the server must be honest. Nevertheless, one cannot discount an attack that intentionally modify a user's device, for example, to extract and send sensitive information to a malicious server. In fact, a scheme that assumes such a scenario, where both the user and the server are malicious is proposed by Goltzsche *et al.* [29]. Their proposed solution utilizes secure enclave to prevent a malicious client from manipulating the TLS traffic.

From these contrasting assumptions, in the following we define in clearer details, the trust assumptions under the three models and use cases that we have discussed.

3.1 Assumptions

Entities. Three main entities are involved in a typical setting. These are a *client*, a *MB provider* and a *server*. In some of the schemes, a fourth entity may be involved. For example, in the outsourced MB setting, a *cloud service provider* hosts the middleboxes. In other settings, an *appliance* acts as a trusted party in an enterprise for registration or traffic encryption, or a *rule generator* sharing/encrypting the rulesets for the middleboxes. An entity can be *honest*, *semi-honest*, or *malicious*. By *honest*, we mean that an entity follows the protocol honestly and adheres to all the security requirements and goals of the protocol. On the other hand, a *semi-honest* entity follows the protocol honestly, but also 'listens' to the communications in order to try to extract or learn the content of the messages from the communications. A semi-honest entity is also known as an *honest-but-curious* entity in the literature. A *malicious* entity is an entity that in addition to listening to the communications, has the capability to modify the communications in order to, for instance, impersonate one of the legitimate entities and learn the underlying messages being transmitted, or assume control of the legitimate entity. We note that it is possible that a client is infected by bot in an enterprise network (hence becoming a *malicious* entity) but the protocols are not affected. That is to say, the traffic flows still route through the enterprise network and the middlebox.

Trust. Table 1 lists the trust assumptions considered by existing schemes. A common setting is that the middleboxes are either honest (H) or semi-honest (S), while either the client or the server is honest[4]. This is the case for schemes using four techniques that we discussed in Section 4. These techniques are Man-in-the-Middle (MitM), searchable encryption (SE), access control (AC) and machine learning (ML) techniques, as shown in row 1–4 in the table. An exception is a scheme called EndBox [29] (row 5), where both the client and the server can be malicious, albeit the MB is assumed honest. This is made possible by deploying trusted hardware (TH) such as secure enclave (i.e. Intel SGX) at the client's device. In this way, the attacker is not able to access the scheme and modify the executions of the code that are carried out inside the enclave. Obviously if all three entities are malicious then there is nothing to be protected. Secure enclave was also used in more recently proposed schemes in the outsourced MB scenario. Examples include

[4]We do not consider the case of semi-honest client/server since the original intend of the communication is for both of them to send/receive messages, and hence they learn the underlying content anyway.

SafeBrick [46], ShieldBox [56] and SGX-Box [31]. Here, the service provider hosting the MBs can be malicious since executions of the MB functionalities, including packet inspection are performed in the secure enclave. It means a malicious service provider should not be able to learn any information from or modify the executions of the enclave.

In terms of system models, many of the schemes cater for client-oriented and server-oriented setup (C-O and S-O) as discussed in Section 2. These schemes can be straightforwardly deploy on either the client enterprise network, or at the server (e.g. web application provider). Schemes based on the searchable encryption (SE) technique such as BlindBox [53] (c.f. second row of Table 1), however, further assumes the existence of a (semi-)honest rule generator that provides rulesets to MB. There are also schemes that cater for client-server accountability (c.f. row 3), in which both endpoints have visibility of the MBs deployed and are able to authenticate these MBs. On the other hand, EndBox [29] and SplitBox [7] focus on client-oriented model, with EndBox providing stronger security assurance where client can be malicious. In summary, there are two main trust assumptions considered by the existing schemes:

- **Trust Assumption (TA) I:** Middlebox (or MB provider) is (semi-)honest and one of the endpoints (i.e. client or server) must be honest. Schemes based on this assumption in general deploy MitM, SE, AC, ML techniques.
- **Trust Assumption (TA) II:** Cloud service provider and one of the endpoints (i.e. client or server) can be malicious. Alternatively, both endpoints can be malicious, but the MB provider is honest. Schemes based on this assumption in general deploy TH techniques.

The discussed schemes cover only assumptions that relate to the perceived use cases and scenarios. Certainly, there are obvious trust combinations that are not feasible, for example:

- In the case whereby the client and the server are honest then MBs would be redundant. The client and the server communicates directly through encrypted traffic. Their main security requirements would be to prevent outsiders from listening or modifying their communication.
- It is obvious that if all three entities are malicious then there is nothing to protect from.

New trust assumptions may be required due to new requirements and scenarios that were not addressed by existing studies. Here we give an example, on the possibility of collusion between the service provider hosting the MBs and one of the endpoints, when we consider the outsourced MB scenario.

Collusion. Note that if MB is malicious (or semi-honest), and either the client or the server is malicious (or semi-honest), the network traffic is exposed to the MB and data can be modified or viewed if the *two entities collude*. This is the case because either the client or the server can share the session key with the MB. If there are many MBs, and some of them are malicious and others are honest, then we may treat this as the scenario of honest client, honest server and malicious MB, by assuming the honest MBs holding the role of clients or servers. At first glance, this may not seem to be practical since it does not make business sense for a service provider to want to learn information from the subscriber by colluding with the other endpoint. However, it can be an unscrupulous employee of the service provider that collude, or the service provider and one of the endpoint being directed to extract information under a government surveillance program.

3.2 Security Requirement

The main goal of schemes advocating privacy-preserving inspection of encrypted traffic is to protect data privacy of the encrypted network traffic, while maintaining similar utility of inspection on plain data traffic. The main security requirement is thus to ensure *data privacy*, in such a way that only the endpoints know the underlying message of the encrypted traffic while the entity that performs the inspection does not learn any information *it is not allowed to learn*. Depending on schemes, the information that is learned by the inspecting entity can vary. For instance, in a Man-in-the-Middle (MiTM) approach that decrypts the network traffic is able to learn all the information if it chooses to.

4 TECHNIQUES

We survey existing state-of-the-art techniques for privacy-preserving inspection of encrypted network traffic, which were briefly stated in the previous section.

We first define two types of inspections used by these techniques.

- *Passive Inspection.* A scheme provides passive inspection if it (1) does not modify the underlying protocol (i.e. SSL/TLS) and (2) does not decrypt the encrypted traffics in order to perform inspection.
- *Active Inspection.* A scheme provides active inspection if the underlying protocol is modified, and/or decryption is required on whole or part of the encrypted traffic. It can further be divided into two sub-categories:
  - *Partial inspection.* A scheme provides active inspection, but only in a restricted manner. For example, a few schemes based on searchable encryption technique enable only exact matching, and not regular expression type of analysis.
  - *Full inspection.* A scheme provides full active inspection. This means the scheme enables all functionality as in inspection on plain data.

4.1 Man-in-the-Middle

The *Man-in-the-Middle* (MitM) technique is commonly deployed in enterpise solutions for encrypted traffic analysis, for example as presented in [10, 18, 22, 58]. There are also widely available open source tools such as MitMProxy [16] and SSLSplit [49]. MitM uses *active inspection*. The main idea is to enable the middlebox provider to act as the server endpoint so that it can decrypt, inspect and then re-encrypt network traffic of the client. The traffic is then forwarded to the server endpoint. In general, for a client-oriented model (Figure 1), this is achieved by first installing a certificate of the MB provider in the client device (e.g. trusted CA store for browsers). When a client initiates a secure session with the server, the MB provider hosting the MBs intercepts the traffic and forges a certificate using the server credentials. In this way the service provider masquerades as the server, setups a session with the client, decrypts

Table 1: Trust Assumptions of Existing Schemes and Solutions

Scheme	Tech. (Sec. 4)	Client	MB	CSP	Server	Model	Remarks
Commercial solutions and specific techniques [19]	MitM	H/M	H	n/a	M/H	C-O S-O	Preserve privacy through a whitelist policy engine (e.g. traffic for online banking is not decrypted).
BlindBox [53], SPABox [26], BlindIDS [12], Embark [37], Yuan *et al.* [61], SplitBox [7] PrivDPI [44], Pine [43]	SE	H/M	S	n/a	M/H	C-O S-O	Introduce (semi-)honest rule generator*. No rule generator for SplitBox, but requires multiple cloud providers.
mcTLS [42], mbTLS [41], maTLS [38] MSP [24], Bhargavan *et al.* [8]	AC	H/M	–	n/a	M/H	C-S	MBs visible to both endpoints.
Anderson *et al.* [4–6] Yamada *et al.* [60]	ML	H/M	S	n/a	M/H	C-O S-O	Analyze metadata of the encrypted traffic.
EndBox [29]	TH	M	H	H	–	C-O	Deploy secure enclave as MB on client.
SafeBricks [46]+, ShieldBox [56]+ SGX-Box [31], LightBox [21]	TH	H M	M M	M M	M H	C-O S-O	MB hosted by service provider can be malicious. Secure enclave in MB performs inspection instead.

H: Honest; S: Semi-honest; M: Malicious; –: Either H, S or M. C-O: Client-Oriented; S-O: Server-Oriented; C-S: Client-Server Accountable; MB: middlebox; CSP: Cloud Service Provider.

For technique, MitM: Man-in-the-Middle, SE: Searchable Encryption, AC: Access control, ML: Machine Learning, TH: Trusted Hardware. Each technique is discussed in details in Section 4.

*: Lan *et al.* (SafeBrick) provides three different rule settings: (1) Both client and MB know the rules; (2) Only client should know the rules. In this case client encrypts the rulesets and passes them to the MB; (3) Only MB should know the rules. In this case a trusted rule generator is required to generate signatures on the rules so that the MB cannot simply generate rules to match arbitrary data from the encrypted traffic.

+: Solution for outsourced MB services. Cloud provider can be malicious but network traffic is protected under a secure enclave implemented in MB.

and inspects the data. After inspection, the service provider initiates on behalf of the client a secure session with the server. In an enterprise network, installation of the root certificate on every client device can be performed by the network administrator. The inspection can be performed by MBs hosted locally in the premise of enterprise network or through outsourced MB service.

In the server-oriented model (Figure 2), similar approach can be deployed. Nevertheless, if the server subscribes to a content delivery network (CDN), the conventional approach is to pass the server certificate, together with the secret key to the CDN provider [8]. In more recent approach, the secret key is not given, but a program interface is provided to the CDN provider to have access to the key. There are various key sharing and content delegation approaches as discussed in [19]. Durumeric *et al.* [22] listed some commercial solutions and explored the security issues of these solutions. They found that nearly all the solutions they investigated have reduced connection security and five of the solutions contain severe vulnerabilities. The technique can be coupled with a policy engine, in which a whitelist of websites can be created by an administrator so that these websites will not be inspected by the MBs [54]. This serves as an approach to mitigate privacy concerns, especially for financial transactions such as online banking and purchases. Nevertheless, the administrator has to be fully trusted to configure the policy engine correctly and honestly. Figure 4 illustrates the MitM settings for client-oriented MB model and the server-oriented MB model (for the use cases involving content delivery network). Here, we use the outsourced MB architecture since it can be used for all three system models that we have presented. Furthermore, it is the emerging direction in both industry and research.

In the followings we discuss the main advantages and limitations of the MitM approach.

- Advantages:
  – *Changes to TLS:* The technique can be deployed straightforwardly without changing the underlying protocol (e.g. TLS) since the service provider sits in the middle and manages secure sessions between the client-service provider and service provider-server.

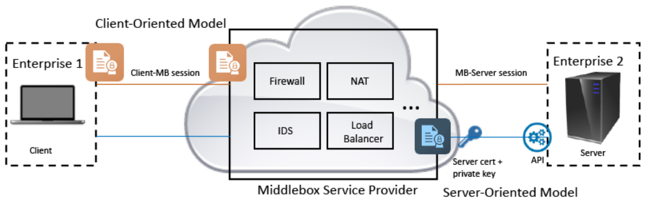

Figure 4: MitM technique: (1) Client-Oriented: Middlebox service provider installs root certificate on client devices to enable middlebox to inspect the encrypted traffic; (2) Server-Oriented in the content delivery network context: Server shares certificate and private key with the service provider, or provides a program interface for the service provider to access the private key.

  – *Functionality:* The MB decrypts the encrypted traffic. This means after the data is decrypted, it is able to run similar functions as in the case where the traffic is not encrypted. Hence it maintains full functionality as if the traffic is not encrypted.
  – *Performance:* It is relatively efficient compared to the other techniques that require additional cryptographic computations, machine learning operations or computation in the trusted hardware. As most modern CPUs support AES-NI instructions, the time to encrypt or decrypt a block of bits take 3μs per packet of 1500 bytes. According to [53], the vanilla TLS setup phase takes 73ms on a 20 Mbps throughput network. Certainly, it incurs higher overhead compared to inspection on plain data since the MB is required to decrypt and re-encrypt the traffic. For commercial solutions, the middlebox latency can be less than 40μs, and depending on the device variant, TLS inspection throughput ranges from 250Mbps-10Gbps. These are based on

publicly available information from the enterprises that we surveyed [9, 28].

- Limitations:
  - *Security:* The main issue in MitM is that the MB provider learns the content of the traffic and this may cause privacy concern, for example, for an enterprise that subscribes to an outsourced MB service. Another issue is that we must trust the MB providers to implement the underlying protocol correctly and uses the up-to-date version of the protocol. As discussed in Section 1 and Section 1.1, weaknesses were found in some of the solutions due to implementation of the TLS protocol using old versions or deprecated ciphers. In addition to this the gateway (or the service provider) becomes highly valuable target for attacker and the administrator must be fully trusted.

4.2 Searchable Encryption

A second technique that was proposed is to detect malicious traffic via token matching without decrypting the underlying encrypted traffic. We categorise the technique as *searchable encryption* (SE) because the main idea is to use searchable encryption scheme to map between encrypted keywords and the encrypted rulesets. SE uses *active inspection.* **Sherry *et al.* [53]** introduced **BlindBox**, the first privacy-preserving deep packet inspection scheme using SE technique. In BlindBox, a client initiates a TLS session with the server, as well as another connection for token matching. Both the connections route through the MB. The general idea is that the MB hosts rulesets that are encrypted using a key derived from the session key of the TLS session. The client then tokenizes and encrypts its message using the same key, and transmits the tokenized traffic through the second connection. The MB then try to match the tokenized traffic with the encrypted rulesets. If there is a match, the traffic is considered malicious and blocked. Figure 5 shows a general setting for SE-based schemes.

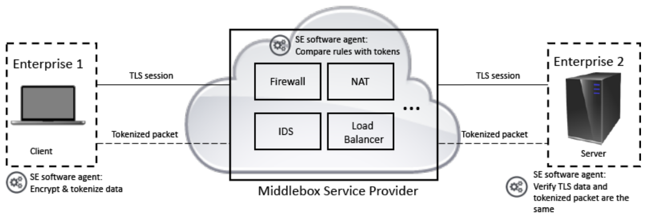

Figure 5: SE technique: All models have the same setting. The main characteristic is that there are two data flows, one the TLS session and another the tokenized data. MB inspects the tokenized data only, and forward the TLS session to the server. The server is able to verify the two data flows are identical since it has the session key used to encrypt the TLS traffic and the tokenized data.

The MB should not know the key used to encrypt the rulesets and to generate the token since if this is the case, it is able to decrypt all tokens in the tokenized traffic and learns the message. This is akin to having access to the plaintext. BlindBox mitigates this issue by performing encryption of the rulesets through garbled circuit and oblivious transfer for every session. This means BlindBox is computationally expensive, at least in the initial handshake of the TLS protocol. Furthermore, the scheme must be able to verify that the tokenized traffic is identical to the TLS session. This is because the client can send malicious message through the tokenized traffic and a benign one through the TLS session, thus defeating the inspection. Here, verification is performed by the server. The server receives data from both connections, and since the server has the session key, it can decrypt the TLS traffic as well as generate the token to verify that the content from both connections are the same. However, inspection based on token matching of the encrypted traffic is rather limited as it would not be able to cater for regular expression inspection. BlindBox proposed a probable cause privacy mechanism to mitigate this issue by allowing decryption of the underlying traffic based on the session key (without revealing the session key).

The BlindBox mechanism is extended to the setting of outsourced MB by **Lan *et al.* [37]**. The scheme is termed **Embark**. It enhanced the token matching technique to include prefix matching, as well as different searchable encryption functionalities catering specifically for different MB services such as IP firewall, NAT, HTTP Proxy, data exfiltration and intrusion detection. It also proposed a different, but practical setting. In order to mitigate the computation overhead of BlindBox, **Canard *et al.* [12]** proposed **BlindIDS**. It uses a different approach in that pairing-based public key operation is deployed for token matching and for the client to communicate with the server through the MB. This also means existing TLS protocol must be replaced with their scheme. Another scheme that uses public key operation is a scheme termed **SPABox** by **Fan *et al.* [26]**. In contrast to BlindIDS, the public key operation based on homomorphic encryption is used only to perform machine learning based inspection. The scheme also uses Diffie-Hellman based oblivious pseudo-random function for encrypted rule preparation, which is performed interactively between the MB and the server. For regular expression matching, the scheme deploys a variant of garbled circuit that is more efficient than the general construction. **Yuan *et al.* [61]** proposed a scheme that is more efficient than BlindBox using a high-performance encrypted filter. Their scheme also extends the token matching mechanism of BlindBox to work for multi-condition rulesets. Though efficient, their scheme requires the server to first register with the administration service of the enterprise hosting the client. Only then the client may initiate the TLS session with the server. The performance of BlindBox is improved significantly by **Ning *et al.* [44]** in their scheme termed as **PrivDPI**. The computation time during the initial handshake was greatly reduced when compared to BlindBox, and a reusable obfuscation mechanism generates intermediate values that can be reused across subsequent sessions for repeated tokens, which could further speedup token encryption. The efficiency of PrivDPI is further improved in a scheme called **Pine**, which is also proposed by **Ning *et al.* [43]**. In addition, Pine enables rule hiding as middlebox services migrate to third-party cloud setting, rule privacy may be a concern. **Ren *et al.*** proposed **EV-DPI [47]**. EV-DPI is a two-layered architecture design and is deployed over two non-colluding servers to outsource the middlebox. First layer filters out legitimate packets using encoded bloom filters. The second layer supports exact rule matching for packet inspection using conjunctive searchable encryption scheme proposed by lai *et al.* [36]. The scheme allows inspection results to be efficiently verified using cuckoo hashing. A related scheme that uses a different approach under the cloud setting for NFV is SplitBox proposed by Asghar *et al.* [7]. They suggest the use of two cloud systems where every rule in the rulesets is XOR with a random string and split into many blocks to the various MBs resided in one of the cloud system. Both cloud systems collaboratively compute the blocks in order to perform inspection on the traffic.

- Advantages:
  - *Changes to TLS:* The main advantage of this technique is that modification on the underlying protocol (i.e. SSL/TLS) is not required (except for BlindIDS which proposes a new protocol that may replace SSL/TLS), yet it allows privacy-preserving inspection. Some of the constructions can also be efficient in specific scenarios. As an example, assuming a trusted rule generator (which is the case for a few of the schemes stated in Table 1), encryption of the rulesets can be performed efficiently if it can be assumed that the rulesets are encrypted by the client or another trusted third party and passed to the MB. Inspections require only comparing the rulesets with the tokenized data, and does not require decryption and re-encryption as in the MitM approach.
  - *Security:* It provides better security guarantee than the MitM technique and the access control technique (Section 4.3) since the inspection is performed without decrypting the encrypted traffic. Hence the MB and the service provider would not be able to learn the content of the traffic, except for those tokens that match the rules. An exception is that there are schemes such as BlindBox, PrivDPI and EV-DPI which follows probable cause privacy where the middlebox gets to see the decrypted traffic if and only if the flow is deemed to be suspicious.
- Limitations:
  - *Functionality:* The main drawback is that utility can be limited due to simple mapping of encrypted tokens and rules. This may not be sufficient especially for inspection requiring regular expression. Some of the schemes address this problem by allowing the MB to decrypt the traffic when there is a match. Alternatively, compute-intensive primitive such as homomorphic encryption is used to enable expressive inspection. Furthermore, in may cases a separate channel is required to communicate the tokenized traffic, as shown in Figure 5. For the MB to be able to inspect inbound and outbound traffic, both endpoints, meaning the client and the server must have the scheme installed. This is in contrast to the MitM technique (as well as the machine learning and trusted hardware approach, which we will discuss in the subsequent sections).
  - *Performance:* At the client, there is the compute-intensive overhead due to the additional operations of tokenizing and encrypting the data stream compared to pure TLS connection, and the MitM technique. At the MB, however, only matching is performed. Hence, generally the setup phase for SE techniques contribute more to the overall performance delay than the inspection phase. For BlindBox, the setup time is 97s for 3000 rules and 33μs for inspecting 1 packet with 3000 rules. BlindIDS limits the setup time to 73ms but increases the inspection time to 74s for same setting. By replacing garbled circuits with reusable exponentiations in group, PrivDPI prepares 3000 encrypted rules in 0.64s which is 288 times more efficient than BlindBox. PrivDPI also reduces the bandwidth usage(from GBs to KB) by replacing the garbled tables with reusable obfuscated rules. Pine further improves the one-round connection time of PrivDPI with hash operations. For SPABox, lack of hardware support for operations results in slower(9 times than BlindBox) performance for encrypting a token at setup phase. However, SPABox saves on average 29.5% time in inspection phase compared to BlindBox due to hashing technique.

4.3 Access Control

This is a technique that advocates client and server accountability, where the client and the server are aware of all the MBs deployed in between the two of them. Also, they are given the ability to authenticate and assign access rights to these MBs. Hence the term *access control* (AC). AC uses *active inspection*. Figure 6 illustrates a typical setting for AC-based schemes.

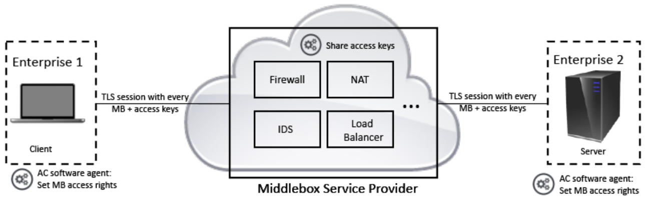
Enterprise 1
Client
AC software agent: Set MB access rights
TLS session with every MB + access keys
Share access keys
Firewall
NAT
IDS
Load Balancer
...
Middlebox Service Provider
TLS session with every MB + access keys
Enterprise 2
Server
AC software agent: Set MB access rights

Figure 6: AC technique: All models have the same setting. The main property is that MBs are visible to the endpoints and that the endpoints may decide access rights of the MBs to the encrypted traffic. The access rights are controlled using read/write keys

Naylor *et al.* [42] introduced a scheme of this type, termed **mcTLS**. It modifies existing TLS protocol to allow the client, the MBs and the server to establish secure and authenticated channel, and exchange read and write secret keys in addition to the session key. The mcTLS protocol partitioned the network traffic according to a least privilege MB access policy. Each MB is assigned a read and/or a write key. This involves providing different level of access control to read and/or write on the encrypted traffic. The encrypted traffic is partition so that some portions of the payloads can be decrypted. The benefit of such a technique is that there is a fine-grained control and flexibility in preserving privacy on portions of the data that should not be decrypted. In other words, mcTLS only partially preserve privacy of the underlying payload, since a MB has the keys to decrypt the portions of the traffic where authorisation is given.

The main issue with mcTLS is that it is a new protocol. For adoptions, existing TLS protocol that is widely used must all be replaced with mcTLS. Acknowledging this issue, **Naylor *et al.* [41]** further proposed another scheme, termed **mbTLS**. It does not change the underlying TLS protocol, except in introducing extensions that can be readily adopted using existing protocol. It is designed to be scalable for implementation in the outsourced MB setting. In more details, mbTLS considers the problem of outsourcing MB functions to multiple third parties. For example, there maybe multiple MBs contracted out by a client, in addition there can also exist a set of MBs for the server, all residing in untrusted cloud environment. They discussed several important properties desired in such a scenario, and designed mbTLS to meet some of such important properties. In mbTLS, a different TLS session exists between every two entity (client, MB, server), with each session operating using its own secret key. This design is such that, the client is only aware of the client-side MBs and none on the server side; and vice versa. The MBs in the cloud uses Intel SGX to protect the data and keys from the cloud infrastructure provider. The initial control messages between the different entities are multiplexed over a single TCP connection, thus adding no additional round trip time. In mbTLS, MBs are able to view the entire payload, as the sessions are handed over to participating entities.

Bhargavan *et al.* [8] demonstrated attacks on mcTLS, and proposed a formal model on analyzing the protocol. In brief, one of the attacks exploit the fact that mcTLS does not perform authentication and the handshake finished message for the session initiation between the client and the MB, and between the MB and the server. Due to this, Bhargavan *et al.* further proposed a protocol that addresses these attacks under the security model that they defined. It does not require any changes to the TLS protocol, in contrast to mcTLS. In response to the attack, the ETSI draft standard that is based on mcTLS proposed a fix to the mcTLS based on message authentication code in their **MB Security Protocol (MSP) specification** [24]. At the moment, it remains to be seen how the industries will accept a secure but MB-friendly new protocol for encrypted traffic, since it may mean replacing TLS with MSP. **Lee *et al.* [38]** proposed **maTLS**, a middlebox-aware protocol to address the MitM pitfalls. Middleboxes are made visible so that server can be explicitly authenticated, the encryption parameters used can be verified and whether messages are modified. Every middlebox is issued a certificate by a CA, and the certificate is logged in a middlebox transparency log server. Through the log server, a middlebox certificate is publicly verifiable and revocable. However, as stated in [19], while the middlebox certificates are vetted by the CA, the client (or end user) still need to decide to accept these certificates. An attacker may launch a phishing attack by obtaining a certificate with convincing name.

- Advantages:
  - *Security:* The main provision of the AC technique is that it provides *accountability*. It means the endpoints will be able to authenticate the deployed MBs, which is unique to this technique.
  - *Performance:* The technique does not require special cryptographic primitives as in SE and hence is more efficient. It is less efficient as the MitM approach, but provide more flexibility in terms of preserving privacy of the data. It does not require installation of certificate at the client, nor allowing any of the MBs to decrypt and view the full payload of the traffic. For example in mcTLS, the performance during handshake protocol depends on number of contexts and middleboxes. Time required for the first byte to reach the endpoint is 400ms with 10 contexts and 560ms with 14 contexts.
  - *Functionality:* It provides full functionality as in the MitM technique. This is because for a specific MB, such as an IDS, access would be given to decrypt the portions of the encrypted traffic for the IDS to perform the required inspections.
- Limitations:
  - *Changes to TLS:* The main pitfall of this approach is it would require all parties, meaning the two endpoints and all the MBs to fully co-operate in order for the scheme to work. This is because the client, and the server must know the type of MBs that they are communicating with. The client, the server and all the MBs must agree on the scheme to be used. It is also not clear how the client and the server may define context (which portion of the data can be reveal, to which MB), especially for data in different domain and then set the access policy for these MBs.
  - *Security:* Union of the access read/write keys for the scenario whereby MBs are hosted at the service provider will allow the service provider to decrypt most probably all portions of the encrypted traffic.

4.4 Machine Learning

Inspecting encrypted traffic based on machine learning (ML) technique represents an ideal solution in terms of security and application setting since it does not require any changes to the existing setup. The idea is to analyse the plain metadata and header of the protocol, extract features from the encrypted payloads, as well as analyse telemetry data from the network traffic. For examples, throgh observing behavioural properties (e.g. the round trip time, number of packets sent), observing the encrypted payloads, and observing additional information such as protocol handshakes. It has been shown to be effective in particular for use cases such as traffic clustering, application type and protocol classification, anomaly detection and file identification [20]. Among all the techniques, it is the only technique that uses *passive inspection.* Figure 7 shows a high-level setting of the ML technique.

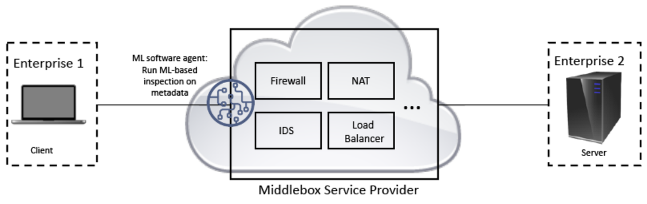
Enterprise 1
Client
ML software agent: Run ML-based inspection on metadata
Firewall
NAT
IDS
Load Balancer
...
Middlebox Service Provider
Enterprise 2
Server

Figure 7: ML technique: All models have the same setting. The main property is that no changes are required to existing industrial deployment on the endpoints. Inspections are performed without needing to modify or decrypt the encrypted traffic.

There are many proposals using ML technique for use cases such as for application type and protocol classification, anomaly detection and file identification. These were comprehensively surveyed in [20]. Another proposal for anomaly detection is the technique proposed by **Yamada *et al.* [60]**. Their scheme performs anomaly detection using only size of data and timing information of the encrypted traffic. More recently, **Anderson *et al.* [4–6]** proposed techniques for malware detection that in addition of utilising sequence of data size and timing, also uses the various TLS header information and DNS data. Anderson *et al.* demonstrates that some of the malware-based encrypted traffic possess distinct characteristics compared to enterprise network traffic. These characteristics when combined with other telemetry data, allows accurate classification of the malware traffic. They have also showed that random forest method outperforms other methods in terms of classifying malware traffic [5]. Yet by careful engineering of the features, including recommendations by domain experts, linear regression using more expressive features actually outperforms random forest method.

There are also techniques that identify malware traffic based on fingerprinting. For example, **JA3/JA3S** proposed by **Althouse *et al.* [3]** generates malware fingerprints based on the TLS metadata (e.g. handshake messages). **Anton [57]** proposed a technique that creates rules by observing the packet byte stream of the traffic to create fingerprint. Specific malware can be detected based on the unique packet byte stream transmitted from the client to the server. The technique is specifically created for use with Suricata. Both the above proposals can be considered as signature-based techniques and require prior knowledge of the malware.

- Advantages:
  - *Changes to TLS:* The main advantage of the ML technique compared to all the other techniques is that it does not require any changes to existing encrypted traffic setup. No modification is required at the client or the server. The MB installs the ML module so that it is able to analyze the metadata of the encrypted traffic.
  - *Security:* Since no changes are required to the setting and the underlying protocol, it preserves the security guarantee of the original setting. For example, using ML technique maintains end-to-end encryption of a TLS session between the client and the server. This represents another main advantage compared to all other techniques that require changes to the protocol (i.e. AC technique) or changes to the client and server settings (i.e. MitM, SE, and trusted hardware). However, we remark that machine learning-based analysis and fingerprint techniques may also be used to learn information about a user. For example, it is possible for an attacker to learn about the websites a user surfs or find out the files a user downloads and shares even though the traffic is encrypted [20]. It is an interesting area of study as to how pervasive ML techniques can also be used to learn about an entity, in contrast to the techniques being used as a privacy-preserving approach for analysing encrypted traffic without knowing the payloads.
- Limitations:
  - *Functionality:* The main concern is whether the technique is sufficiently comprehensive to cover most of the detection requirements of MBs performing security functions. As was stated in [20], there are inherent limitation to what can be analysed based on Ml techniques, and as was discussed in [19], there are use cases that will require inspection on the payloads, not just the headers and metadata. As of now, the most successful ML mechanism of Anderson *et al.* [4–6] only cater for malware detection. All in all, the technique will need to be able to cater for different types of middlebox services.
  - *Performance:* It remains to be seen how the performance of the technique as compared to other techniques. Training must be performed before detection can be carried out on real-time data. The timely dataset and accuracy of the training model are area to be further explored. As was discussed by Anderson and McGrew [5] in their work on malware classification using machine learning technique, it took approximately 200 seconds to train and less than 10 seconds to test. This is for their best performed random forest algorithm, on the enhanced feature. The timing is a rough estimation from Figure 5 in their work [5]. Also, solutions based on this approach need to continuously feeding high fidelity labelled data for training, which can be hard to obtain.

4.5 Trusted Hardware

Trusted hardware (TH) has also been deployed for privacy-preserving packet inspection. The emerging practice is to utilize the secure enclave of the Intel SGX trusted hardware. The general idea is for the client or the server to share the session key securely with the enclave residing in the MB. The decryption, inspection and re-encryption is performed in the enclave. The MBs and the service provider hosting these MBs are not able to discern or learn the data and processes. The challenge is how to implement middlebox functionalities efficiently and securely to fully utilize the capability of the trusted hardware. We may classify this technique as using *active inspection*, but the inspection is hidden from the view of the entity hosting the trusted hardware. Figure 8 shows a high-level setup of the TH technique.

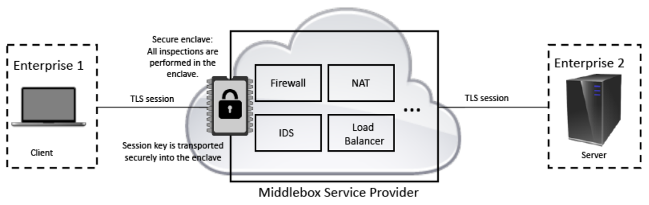
Enterprise 1
Client
TLS session
Secure enclave: All inspections are performed in the enclave.
Session key is transported securely into the enclave
Firewall
NAT
IDS
Load Balancer
...
Middlebox Service Provider
TLS session
Enterprise 2
Server

Figure 8: TH technique: Similarly all models have the same setting. No changes are required on the TLS protocol. The main property is that decryption of the traffic is performed in the secure enclave. The main requirement is on securely transporting the session key into the enclave, either by the client or the server.

Han *et al.* [31] proposed a scheme, **SGX-Box**, using this technique. In their scheme, the server shares the session key with the Intel SGX's secure enclave through an out-of-band secure channel. They presented an efficient implementation of performing the inspection inside the secure enclave.

It is also possible to push the MB functions to the network edge (or to the client's device). This is anologous to hosting the MB at the client, except that inspection are performed in the secure enclave embedded in the client device. This is beneficial in that all outbound traffic is analyzed before it is transmitted from the client device. Inbound traffic on the other hand is verified before it is released to the client device. **EndBox**, proposed by **Goltzsche *et al.* [29]**, provides such a solution.

There are also works that propose comprehensive secure Network Function Virtualization (NFV) systems using TH technqiue. These NFV systems provide inspection of encrypted traffic as one of their functionality. For example, **SafeBricks** proposed by **Poddar *et al.*** [46]. SafeBricks provides MB as a cloud service construction. It ensures the cloud provider only sees the encrypted traffic, where the application header (normally visible under TLS) is also encrypted based on IPSec. Additionally, it shields the rulesets and the network function codes from the cloud provider. LightBox further improves on existing TH-based schemes, including SafeBricks, in terms of efficiency and properties. For instance, LightBox additionally protects privacy of metadata of the traffic, including packet size, count and timing. Two related works are ShieldBox proposed by Trach *et al.* [56] and LightBox proposed by Duan *et al.* [21]. ShieldBox and LightBox provide secure middlebox functionalities by provision of NFV systems in SGX. However, both are geared towards efficient deployment of MBs on untrusted cloud providers and would require adaptation to specifically cater for encrypted traffic inspection.

- Advantages:
  - *Changes to TLS:* One advantage is that it does not require changes to the underlying protocol. However, it must be able to transport the session key to the trusted hardware securely, either by the client or the server.
  - *Functionality:* Another advantage is it provides MitM-type full functional inspection without the drawbacks of the MitM approach. It allows a MB to inspect the encrypted traffic in a shielded environment, so that no decrypted packets are leaked. It may represent the most practical solution if hardware deployment, cost and security of the trusted hardware is not an issue.
- Limitations:
  - *Security:* The current main concern is that security of the trusted hardware (i.e. Intel SGX) is still being actively study. As was presented by Lindell [39], there has been successful side-chanel attacks on Intel SGX. It remains to be seen how serious such attacks are in practical deployment.
  - *Performance:* It is less efficient for inspection, at least when compared to SE scheme as discussed in [37]. Take SafeBricks for example, it introduces 16% bandwidth overhead when compared to Embark's 21% bandwidth overhead. Nevertheless, SafeBricks also incurs 0-15% throughput performance overhead at the middlebox due to SGX compared to negligible overheads of Embark and BlindBox. However, Embark(at gateway) and BlindBox(at client) incur high latency in order to encrypt the packets at the endpoints, where SafeBricks does not pay such cost. Another issue is the availability of the hardware and cost. While most recent devices would have a secure enclave embedded (i.e. Intel SGX), issues remain for those that are not, especially legacy systems.

4.6 Comparisons

As a summary for what we have discussed thus far, Table 2 provides comparisons between the models, use cases, characteristics (e.g. security, types, models, outsourced MB to cloud), the five techniques and performances.

4.6.1 Security. In the following we define three different levels of security, shown in Table 2. The main goal of the schemes is to provide data privacy (as stated in Section 3.2). In other words, how much information is leaked to an adversary, which may include the entity hosting the MBs.

- *Full Reveal* (○). This means the network traffic or data payload is visible (or decrypted in total) to the MB. Solutions based on MitM technique are in this category.
- *Partial Reveal* (◓, ◕). This means only partial content of the network traffic is revealed (or partial decryption of the payload) to the MB. Solutions based on SE and AC techniques are in this category. However, there is a subtle difference between the partial data exposure between the two techniques. SE matches encrypted rules with the encrypted token generated from the payload, and hence only reveal matched result. Encrypted payload that does not match any of the rules remain private from the view of the MB. An exception are schemes that reveal the underlying data for more complex inspection such as regular expression. In contrast, schemes based on AC technique directly divulge the underlying payload as long as the MB has the access right to some parts of the encrypted traffic data. By the above observation, we may state that SE techniques leak less information (◕) when compare to AC techniques (◓).
- *Hidden* (●). This means the network traffic or data payload is hidden (or remain encrypted) and the MB has no visibility to the underlying content. Schemes based on ML and TH techniques are in this category. Note that here we assume the trusted hardware deployed in the schemes are considered secure, in that there is no information leakage from the hardware itself when executing the schemes. This may be a strong assumption as there are ongoing work demonstrating side-channel attacks to such hardware implementation [39]. It remains a research problem as to the effects of such attacks toward schemes using TH technique.

4.6.2 Utility. In terms of utility, we define two broad categories, which are also shown and described in Table 2.

- *Full functionality (●).* It means a scheme provides inspection utilities similar to that of inspection on plaintext traffic. MitM approaches are in this category.
- *Partial functionality (◔, ◓, ◕).* It means a scheme only provide partial utilities. For example, solutions based on SE perform direct matching and prefix matching against pre-defined rule sets. They also require specific setting such as tokenisation of the messages

and a separate encryption channel in addition to the TLS traffic. Hence functionality for SE techniques is limited when compared to other techniques (◔). ML-based solutions also have limited inspection capability as was previously discussed 4.4, but they do not require any changes to the existing setting and can be deployed directly (◓). Solutions based on AC and TH in theory can achieve full inspection utilities as in MitM approach. The AC context and policy can be set to authorise a middlebox to have full control on the traffic (i.e. decrypt then read and write on the messages), and the TH in principle can decrypt and perform all types of inspection under the secure enclave. However, in practical terms, the main goal of the AC technique is to restrict access to the encrypted payload depending on the requirement of a middlebox. By doing so the client and the server may control the portion of the data that is allowed to be read or modify by a middlebox. In contrast to the MitM approach, the setup is such that no one middlebox is able to decrypt the full message, except for the recipient. It would also require new setting and configuration of access context for the client and the server. For TH, the secure enclave provides limited storage/memory and computation capabilities. It would not be practical to perform all types of inspection under the secure enclave, at least not for the current TH technology. Hence we denote both the AC and TH techniques as achieving partial but better functionality (◕), as compared to the SE and ML-based solutions.

5 DISCUSSIONS

For all the novel schemes that have been proposed to date, a question that one may ask is why is that most of the industries still prefer and deploy MitM-based solutions. Conversely, if a MitM-based solution is designed in a careful manner, that is, with up-to-date TLS configuration and policy engine, is it not sufficiently secure? On the other hand, schemes based on the four techniques were and are continuously being proposed to address the issues with the MitM approach, i. e. violation of end-to-end encryption, privacy concern and difficulties or flaws in the configuration and settings. The main goals are mainly to maintain end-to-end security provided by the underlying protocol such as TLS, and reduce the information revealed to the MB. One of the reasons is to reduce the trust placed on the MB providers so that one does not rely on the provider to *design and implement encrypted traffic inspection in a careful manner*, which is always not possible as demonstrated by Jarmoc [34], Carnavalet and Mannan [18], and Durumeric *et al.* [22]. *All in all the aim is to replace the MitM-based solution with either of the other four techniques (or combination of them) for better privacy preservation yet maintain similar utility and practicalility in deployment.*

We discuss the challenges in the techniques discussed and suggest research directions. Our discussions are based on common properties in the cyber security landscape, that is, security, performance and utility.

5.1 Security

Information Leakage. The main challenge in terms of security is leakage of information in the techniques that we have discussed. In schemes based on SE, token matching may still leak information if the client or the service provider (in the case that the provider should not know the rulesets) has background information of the underlying rulesets. This is a reasonable assumption since there are publicly available rulesets such as the rulesets provided by Snort. This concern was discussed by Poddar *et al.* [46]. Furthermore, there are well-established attacks on searchable encryption schemes such as inference attack [32], leakage-abuse attack [14], reconstruction attacks [35] and passive attacks [45]. Similarly, further study is required on schemes that utilise trusted hardware due to the recent attacks based on side information on the secure enclave, such as the attacks discussed in [11].

Potential collusion in the outsourced MB scenario. In terms of the outsourced MB model, one of the challenges that has not been examined is the possibility of collusion between a malicious client or server and the cloud service provider. The malicious client or server may collude with the service provider by providing the session key in order for the provider to decrypt the underlying traffic, thus circumventing the executions of the schemes, be it through solutions based on SE, AC, ML or TH. A new security model may be required to model such scenario and the question would then be whether this is a practical assumption. For example, in the scenario where an enterprise outsources its content management to a content distribution network (CDN) provider, the enterprise shares the certificate private key, or delegates content to the CDN provider. Assuming that the CDN provider uses its own cloud infrastructure, there is no collusion in this instance. In contrast, if an enterprise outsources network threats detection to a managed service provider, whereby the provider uses a third-party cloud provider to host its various MB services, then the possibility of collusion might need to be taken into consideration.

Challenges for ML Technique. ML technique enables analysis without needing to change and decrypt the existing encrypted network setting (e.g. TLS). This provides the best solution since it passively inspect traffic. Nevertheless, the one challenge is whether the technique is sufficiently comprehensive to cover most of the detection requirements of MBs performing security functions. As of now, the most successful ML mechanisms of Anderson *et al.* [4–6] only cater for malware detection. Also, training must be performed before detection can be carried out on real-time data. The timely dataset and accuracy of the training model are area to be further explored.

5.2 Performance

SE-based schemes preserve privacy but incur huge overhead. Schemes based on SE technique require two communication channels (e.g. one for the TLS connection and one for the encrypted tokens), the generation of tokens and encryption of rulesets (cf. Section 4.2). This means SE-based schemes incur extra overheads, at least compare to the MitM approach. However, it represents a promising technique to privately inspect encrypted payloads without needing any specialised hardware as in the TH-based solutions. It also does not reveal partially the underlying data, as in the case of AC-based solutions, except when regular expression type of matching is required. The challenge is then to improve the efficiency on the current schemes especially in terms of rulesets and encryption and matching. A question that need to be addressed

Table 2: Privacy-Preserving Techniques for Encrypted Traffic Inspection: Types, Techniques and Applications

Scheme	chg. TLS	Types		Technique				Cl.	L.	Util.	Pri.	Configuration			
		Pa.	Act.	AC	SE	ML	TH					Initial Setup	Pre-processing	Encrypt/Decrypt	Match/Inspect
MitM approach	×		•						○	●	cert. + policy	Client install root cert Server share private key	client/server share session key Server delegate content	MBs decrypt messages	MBs inspect decrypted payload
mbTLS [42]	×		•	•			•	•	◓[*]	◕	sym.	Client & server agrees on list of MBs	Delegate MBs' read/write keys	MBs partial decrypt by read/write access	MBs inspect decrypted payload
mcTLS [41]	✓		•	•					◓[*]	◕	sym.	Same as mbTLS[@]	−−	−−	−−
MSP [24]	✓		•	•					◓[*]	◕	sym.	Same as mcTLS[+]	−−	−−	−−
Bhargavan *et al.* [8]	×		•	•					◓[*]	◕	sym.	Same as mcTLS[#]	−−	−−	−−
maTLS [38]	×		•	•					○	●	sym.	MBs get certs from CA (Certs publicly verifiable)	MBs share ciphersuites setting with client	MBs decrypt messages	MBs inspect decrypted payload
BlindBox [53]	×		•		•				◕	◔	sym. + SMC	Rule generator prepares & signs rules	Client tokenises messages Client & MB jointly encrypt rules	Client encrypts tokens	Exact match: encrypted tokens vs rules
Embark [37]	×		•		•			•	◕	◔	sym.	Enterprise gateway encrypts rules, passes encrypted rules to MBs	Gateway tokenises messages from client	Gateway encrypts tokens	Exact & prefix match: encrypted tokens vs rules
Yuan *et al.* [61]	×		•		•			•	◕	◔	sym. + SC	Server gets a key from admin at client, admin encrypts rules, passes to MB	Server/Client tokenises messages	Server/Client encrypts tokens	Exact & multi−rules match: encrypted tokens vs rules
BlindIDS [12]	✓[$]		•		•				◕	◔	Pairing	Server gen. key pair Rule gen/editor gen. encrypted rules, passes them to MBs	Client tokenises messages	Client encrypts tokens using server public key	Exact match: encrypted tokens vs rules
SPABox [26][a]	×		•		•				◕	◔	DLP + HE	Rule generator prepares rules	Client tokenises messages Client & Server negotiates DLP/HE parameters	Client encrypts tokens	Exact match, regular expressions, ML: Encrypted tokens vs rules/models
PrivDPI [44][a]	×		•		•				◕	◔	sym. + DLP	Rule generator prepares & signs rules	Client tokenises messages Client/MB/Server jointly gen. reusable encrypted rules	Client encrypts tokens	Exact match: encrypted tokens vs rules
Pine [43][a]	×		•		•				◕	◔	sym. + DLP	Enterprise gateway shares a key with rule generator, rule generator prepares & signs rules	Gateway tokenises messages from client, gateway/MB jointly gen. reusable encrypted rules	Client encrypts tokens	Exact match: encrypted tokens vs rules
EV-DPI [47]	×		•		•			•	◕	◔	sym. + BF + CH	Enterprise gateway encrypts rules, passes encrypted rules to MBs	Gateway tokenises messages from client	Gateway encrypts tokens	Filter & exact match: encrypted tokens vs rules
Yamada *et al.* [60]	×	•				•			●	◓	Statistical analysis	Features: data size, time, HTTP traffic, access frequencies	Feature vector extraction	n/a	Anomaly detection for IDS: Frequency analysis
Anderson *et al.* [4–6]	×	•				•			●	◓	ML classifiers	Features: flow metadata, TLS handshake msg. etc.	Feature vector extraction	n/a	Malware traffic classification
SGX-Box [31]	×		•				•	•	●	◕	SGX	Configure SGX, attest module, & update server app.	Server securely shares session key with enclave	Enclave decrypts messages	Enclave inspects decrypted payload
EndBox [29]	×		•				•	Edge	●	◕	SGX	Configure SGX, attest module, enclave installs CA cert, enclave gen. key pair	Client app. securely shares session key with enclave at client (e.g. Edge device)	Enclave at client decrypts messages	Enclave at client inspects payload
SafeBricks [46]	×		•				•	•	●	◕	SGX	Configure SGX, attest module, embed network functions in the enclave	Enterprise gateway decrypts TLS traffic from clients, tunnels to cloud enclave via IPSec	Enclave at cloud decrypt messages	Enclave at cloud inspects payload

"chg. TLS": Required changes to TLS, Pri.: Primitives, e.g. crypto primitives or metadata, Cl: Cloud, L: information leakage Util.: Utility, Pa.: Passive, Act.: Active, AC: Access Control, SE: Searchable Encryption, ML: Machine Learning, TH: Trusted Hardware, sym.: symmetric primitives for encryption/decryption (i.e. AES), SMC: Secure Multi-party computation, SC: Secret sharing, DLP: Discrete Log Problem, HE: Homomorphic encryption, SGX: Intel implementation of secure enclave, BF: Bloom Filter, CH: Cuckoo Hashing.

a: SpaBox, PrivDPI and Pine focus on improving the performance of BlindBox with added properties. SpaBox enables more expressive matching. PrivDPI introduces reusable obfuscated rules so that obfuscated rules can be re-use in many sessions, as compared to BlindBox that requires re-encryption of rules for every session. Pine introduces rule-hiding so that MBs do not learn the rules, and dynamic addition of rules.

$: BlindIDS proposes a new protocol based on pairing, that can be deployed as a replacement to SSL/TLS

∗: parts of the underlying encrypted traffic are decrypted based on access rights. This means MB sees part of the plaintext.

@: Similar configuration with mbTLS but the protocol constructed is a modified version of the TLS protocol. In contrast, mbTLS only requires the use of TLS extension, and hence does not affect existing TLS implementation.

+: fix security issues in mcTLS.

#: can be instantiated with unmodified TLS 1.3 draft 23. Constructed a provably secure protocol. The protocol is not meant to encourage adoption of active proxying but to demonstrate the difficulty of constructing a secure proxied end-to-end security protocol is that in the case where specialised hardware is widely-available, would SE-based approach still play a role in providing inspection of encrypted traffic in a private manner? The answer would be yes if combining the SE-based approach and the TH approach to lessen the processing loads of TH is relatively more efficient than having all encrypted traffic being processed in the trusted hardware.

ML approach presents ideal solution but inefficient? Schemes based on ML represents the ideal solution since existing setup is not required to be changed. However, performance based on ML matching may need to be further explored especially when compared with all the other approaches.

AC-based schemes as a replacement to MitM?. In order to avoid the security issues of the MitM-based solutions, schemes based on AC technique proposes protocol that requires all MBs to be accountable between the client and the server. If this is deployed it means the client and the server are able to setup a secure session with each of the MB and decide what data the MB is allowed to view. The performance would then be similar to a plain MitM-based solution since the MB performs decryption, inspection and re-encryption as before, but in the AC setting. The challenge is not so much on performance in this case, but security, configuration and utility, which we discuss in the other two sections.

Moving towards TH-based solutions. As per our observation on the most recent literatures, new schemes are leaning towards utilizing trusted hardware efficiently such as the secure enclave technology provided by Intel SGX that is available in most of the recent Intel processors. The challenge is the continuous efforts in improving the performance while at the same time minimize leakages of information. This is crucial since the secure enclave may be considered resource-constrained device in a certain sense, and not all network traffic should be routed into the memory or storage space of the enclave for processing. The research direction is thus to examine and construct efficient, yet private communication protocol between the enclave and the MB (and/or the service provider). Alternatively, a scheme may construct a secret share protocol that utilize multiple enclaves that distribute workload between these enclaves in a secure manner.

5.3 Utility

In terms of utility, schemes based on AC and TH have the capability to provide full functionality similar to inspection on plaintext data. This is because both techniques enable MBs to decrypt the underlying traffic.

Full functionalities without client-server accountability. The challenge for schemes based on AC technique is that whether it is possible to achieve such utility without having to a priori decide and authenticate the MBs involved in the communication between the client and the server. It is not clear when a new MB is introduced, or an existing one being removed, how the client and the server update their communication. Furthermore, AC introduces the concept of context, where the client and the server have the flexibility of setting a MB access policy towards their data. This means deciding the MB that can read and/or write a particular section of the encrypted data. It is also not clear how this can be performed in a systematic and accurate manner, especially there are information in different domain that may requires specific access policy. As was stated in [19], solutions based on AC technique require the support of the server and the client, which may not be feasible since for example the server has no interest to help a client that would like to prevent malware download.

Extending functionalities of SE and ML Techniques. In the case where deploying specialised hardware is not an option, especially with legacy system, one may seek to extend the limited functionalities of the SE-based and ML-based approach. The difficulty for SE-technique is the complications of extending the token matching mechanism without leaking substantial information. While Embark [37] and Yuan *et al.* [61] extended the capability of BlindBox [53], the matching still does not provide full regular expression matching. For ML-based schemes, the challenge would be to construct ML algorithms and models that enable detection of different-type of anomaly traffics without inspecting the payload.

6 CONCLUSIONS

In this work we present a comprehensive survey on the topic of privacy-preserving inspection over encrypted traffic. We define a trust model and categorise the different network settings based on existing state-of-the-art schemes. From our compilation, we further categorise the current schemes into four main techniques, that enable us to demonstrate the advantages and limitations of each of the proposals. These gave insights into the suitability of the proposals to be deployed in practice, which we also discussed. The main difficulty is to fill the gap between actual deployment, which very much still based on the man-in-the-middle approach that does not preserve privacy. To this, we listed and discussed the many challenges faced in the existing techniques and possible directions for improvements.

ACKNOWLEDGMENTS

Research supported by the National Research Foundation, Prime Minister's Office, Singapore, under its Corporate Laboratory@ University Scheme, National University of Singapore, and Singapore Telecommunications Ltd.

REFERENCES

[1] National Security Agency. 2019. Transport Layer Security Inspection. https://www.us-cert.gov/ncas/current-activity/2019/11/19/nsa-releases-cyber-advisory-managing-risk-transport-layer-security.

[2] Akamai. accessed Dec 2020. Akamai Web Service Security. https://www.akamai.com/us/en/resources/web-service-security.jsp.

[3] John Althouse, Jeff Atkinson, and Josh Atkins. accessed Dec 2020. TLS Fingerprinting with JA3 and JA3S. ttps://engineering.salesforce.com/tls-fingerprinting-with-ja3-and-ja3s-247362855967.

[4] Blake Anderson and David A. McGrew. 2016. Identifying Encrypted Malware Traffic with Contextual Flow Data. In *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security, AISec@CCS 2016, Vienna, Austria, October 28, 2016*, David Mandell Freeman, Aikaterini Mitrokotsa, and Arunesh Sinha (Eds.). ACM, 35–46. https://doi.org/10.1145/2996758.2996768

[5] Blake Anderson and David A. McGrew. 2017. Machine Learning for Encrypted Malware Traffic Classification: Accounting for Noisy Labels and Non-Stationarity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017.* ACM, 1723–1732. https://doi.org/10.1145/3097983.3098163

[6] Blake Anderson, Subharthi Paul, and David A. McGrew. 2018. Deciphering malware's use of TLS (without decryption). *J. Computer Virology and Hacking Techniques* 14, 3 (2018), 195–211. https://doi.org/10.1007/s11416-017-0306-6

[7] Hassan Jameel Asghar, Luca Melis, Cyril Soldani, Emiliano De Cristofaro, Mohamed Ali Kâafar, and Laurent Mathy. 2016. SplitBox: Toward Efficient Private Network Function Virtualization. In *Proceedings of the ACM SIGCOMM Workshop on Hot topics in Middleboxes and Network Function Virtualization, HotMiddlebox@SIGCOMM 2016, Florianopolis, Brazil, August, 2016*, Dongsu Han and Danny Raz (Eds.). ACM, 7–13. https://doi.org/10.1145/2940147.2940150

[8] Karthikeyan Bhargavan, Ioana Boureanu, Antoine Delignat-Lavaud, Pierre-Alain Fouque, and Cristina Onete. 2018. A Formal Treatment of Accountable Proxying over TLS. In *2018 IEEE Symposium on Security and Privacy, SP 2018, San Francisco, CA, USA, May 21-23, 2018*. IEEE Computer Society, 339–356.

[9] Broadcom. accessed 2020. Symantec SSL Visibility Appliance. https://docs.broadcom.com/doc/ssl-visibility-appliance-en.

[10] Broadcom. accessed Dec 2020. Manage encrypted traffic with SSL visibility appliance. https://www.broadcom.com/products/cyber-security/network/encrypted-traffic-management.

[11] Jo Van Bulck, Marina Minkin, Ofir Weisse, Daniel Genkin, Baris Kasikci, Frank Piessens, Mark Silberstein, Thomas F. Wenisch, Yuval Yarom, and Raoul Strackx. 2018. Foreshadow: Extracting the Keys to the Intel SGX Kingdom with Transient Out-of-Order Execution. In *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018.*, William Enck and Adrienne Porter Felt (Eds.). USENIX Association, 991–1008. https://www.usenix.org/conference/usenixsecurity18/presentation/bulck

[12] Sébastien Canard, Aïda Diop, Nizar Kheir, Marie Paindavoine, and Mohamed Sabt. 2017. BlindIDS: Market-Compliant and Privacy-Friendly Intrusion Detection System over Encrypted Traffic. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017, Abu Dhabi, United Arab Emirates, April 2-6, 2017*, Ramesh Karri, Ozgur Sinanoglu, Ahmad-Reza Sadeghi, and Xun Yi (Eds.). ACM, 561–574. https://doi.org/10.1145/3052973.3053013

[13] Brian Carpenter and Scott Brim. 2002. Middleboxes: Taxonomy and Issues. RFC 3234. https://tools.ietf.org/html/rfc3234.

[14] David Cash, Paul Grubbs, Jason Perry, and Thomas Ristenpart. 2015. Leakage-Abuse Attacks Against Searchable Encryption. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*, Indrajit Ray, Ninghui Li, and Christopher Kruegel (Eds.). ACM, 668–679. https://doi.org/10.1145/2810103.2813700

[15] Cisco. 2018. Encrypted Traffic Analytics. https://www.cisco.com/c/dam/en/us/solutions/collateral/enterprise-networks/enterprise-network-security/nb-09-encrytd-traf-anlytcs-wp-cte-en.pdf.

[16] Aldo Cortesi, Maximilian Hils, and Raum Fresser. 2018. MitMProxy. https://mitmproxy.org/.

[17] Cybersecurity and Infrastructure Security Agency (CISA). 2017. HTTPS Interception Weakens TLS Security. Alert (TA17-075A), CISA, U.S. Department of Homeland Security. https://www.us-cert.gov/ncas/alerts/TA17-075A.

[18] Xavier de Carné de Carnavalet and Mohammad Mannan. 2016. Killed by Proxy: Analyzing Client-end TLS Interception Software. In *23rd Annual Network and Distributed System Security Symposium, NDSS 2016, San Diego, California, USA, February 21-24, 2016*. The Internet Society. http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2017/09/killed-proxy-analyzing-client-end-tls-interception-software.pdf

[19] Xavier de Carné de Carnavalet and Paul C. van Oorschot. 2020. A survey and analysis of TLS interception mechanisms and motivations. *CoRR* abs/2010.16388 (2020). arXiv:2010.16388 https://arxiv.org/abs/2010.16388

[20] Paraskevi Dimou, Jan Fajfer, Nicolas Müller, Eva Papadogiannaki, Evangelos Rekleitis, and František Střasák. 2019. Encrypted Traffic Analysis: Use Cases and Security Challenges. European Union Agency for Cybersecurity (ENISA). https://www.enisa.europa.eu/publications/encrypted-traffic-analysis.

[21] Huayi Duan, Xingliang Yuan, and Cong Wang. 2017. LightBox: SGX-assisted Secure Network Functions at Near-native Speed. *CoRR* abs/1706.06261 (2017). arXiv:1706.06261 http://arxiv.org/abs/1706.06261

[22] Zakir Durumeric, Zane Ma, Drew Springall, Richard Barnes, Nick Sullivan, Elie Bursztein, Michael Bailey, J. Alex Halderman, and Vern Paxson. 2017. The Security Impact of HTTPS Interception. In *24th Annual Network and Distributed System Security Symposium, NDSS 2017, San Diego, California, USA, February 26 - March 1, 2017*. The Internet Society. https://www.ndss-symposium.org/ndss2017/ndss-2017-programme/security-impact-https-interception/

[23] ETSI. 2018. CYBER; Middlebox Security Protocol; Part 1: Profile Capability Requirements. Draft ETSI TS 103 523-1 V0.0.13 (2018-04).

[24] ETSI. 2018. CYBER; Middlebox Security Protocol; Part 2: Transport layer MSP, profile for fine grained access control. Draft ETSI TS 103 523-2 V0.0.8 (2018-04).

[25] ETSI. 2020. CYBER; Middlebox Security Protocol; Part 1: MSP Framework and Template Requirements. ETSI TS 103 523-1 V1.1.1 (2020-12). https://www.etsi.org/deliver/etsi_ts/103500_103599/10352301/01.01.01_60/ts_10352301v010101p.pdf.

[26] Jingyuan Fan, Chaowen Guan, Kui Ren, Yong Cui, and Chunming Qiao. 2017. SPABox: Safeguarding Privacy During Deep Packet Inspection at a MiddleBox. *IEEE/ACM Trans. Netw.* 25, 6 (2017), 3753–3766. https://doi.org/10.1109/TNET.2017.2753044

[27] Nick Feamster. 2010. Outsourcing Home Network Security. In *Proceedings of the 2010 ACM SIGCOMM Workshop on Home Networks (HomeNets '10)*. ACM, New York, NY, USA, 37–42. https://doi.org/10.1145/1851307.1851317

[28] Gigamon. accessed 2020. Gigamon Visibility and Analytics Fabric with GigaSMART. https://www.gigamon.com/content/dam/resource-library/english/feature-brief/fb-ssl-tls-decryption.pdf.

[29] David Goltzsche, Signe Rüsch, Manuel Nieke, Sébastien Vaucher, Nico Weichbrodt, Valerio Schiavoni, Pierre-Louis Aublin, Paolo Costa, Christof Fetzer, Pascal Felber, Peter Pietzuch, and Rüudiger Kapitza. 2018. ENDBOX: Scalable Middlebox Functions Using Client-Side Trusted Execution. In *48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2018, Luxembourg, June 25-28, 2018*. IEEE Computer Society, 1–12.

[30] Bo Han, Vijay Gopalakrishnan, Lusheng Ji, and Seungjoon Lee. 2015. Network function virtualization: Challenges and opportunities for innovations. *IEEE Communications Magazine* 53, 2 (2015), 90–97. https://doi.org/10.1109/MCOM.2015.7045396

[31] Juhyeng Han, Seong Min Kim, Jaehyeong Ha, and Dongsu Han. 2017. SGX-Box: Enabling Visibility on Encrypted Traffic using a Secure Middlebox Module. In *Proceedings of the First Asia-Pacific Workshop on Networking, APNet 2017, Hong Kong, China, August 3-4, 2017*, Kai Chen and Jitendra Padhye (Eds.). ACM, 99–105. https://doi.org/10.1145/3106989.3106994

[32] Mohammad Saiful Islam, Mehmet Kuzu, and Murat Kantarcioglu. 2012. Access Pattern disclosure on Searchable Encryption: Ramification, Attack and Mitigation. In *19th Annual Network and Distributed System Security Symposium, NDSS 2012, San Diego, California, USA, February 5-8, 2012*. The Internet Society. https://www.ndss-symposium.org/ndss2012/access-pattern-disclosure-searchable-encryption-ramification-attack-and-mitigation

[33] ITU-T. 2012. Requirements for deep packet inspection in next generation networks. Next Generation Networks – Security, SERIES Y: GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS AND NEXT-GENERATION NETWORKS, Rec. ITU-T Y.2770 (11/2012). https://www.itu.int/rec/T-REC-Y.2770-201211-I/en.

[34] Jeff Jarmoc. 2012. Transitive Trust: SSL/TLS Interception Proxies. https://www.secureworks.com/research/transitive-trust.

[35] Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'Neill. 2016. Generic Attacks on Secure Outsourced Databases. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi (Eds.). ACM, 1329–1340. https://doi.org/10.1145/2976749.2978386

[36] Shangqi Lai, Sikhar Patranabis, Amin Sakzad, Joseph K Liu, Debdeep Mukhopadhyay, Ron Steinfeld, Shi-Feng Sun, Dongxi Liu, and Cong Zuo. 2018. Result pattern hiding searchable encryption for conjunctive queries. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. 745–762.

[37] Chang Lan, Justine Sherry, Raluca Ada Popa, Sylvia Ratnasamy, and Zhi Liu. 2016. Embark: Securely Outsourcing Middleboxes to the Cloud. In *13th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2016, Santa Clara, CA, USA, March 16-18, 2016*, Katerina J. Argyraki and Rebecca Isaacs (Eds.). USENIX Association, 255–273. https://www.usenix.org/conference/nsdi16/technical-sessions/presentation/lan

[38] Hyunwoo Lee, Zach Smith, Junghwan Lim, Gyeongjae Choi, Selin Chun, Taejoong Chung, and Ted Taekyoung Kwon. 2019. maTLS: How to Make TLS middlebox-aware?. In *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*. The Internet Society. https://www.ndss-symposium.org/ndss-paper/matls-how-to-make-tls-middlebox-aware/

[39] Yahuda Lindell. 2018. The Security of Intel SGX for Key Protection and Data Privacy Applications. https://cdn2.hubspot.net/hubfs/1761386/security-of-intelsgx-key-protection-data-privacy-apps.pdf.

[40] Mary Meeker. 2019. Internet Trends 2019. https://www.bondcap.com/report/itr19/.

[41] David Naylor, Richard Li, Christos Gkantsidis, Thomas Karagiannis, and Peter Steenkiste. 2017. And Then There Were More: Secure Communication for More Than Two Parties. In *Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies, CoNEXT 2017, Incheon, Republic of Korea, December 12 - 15, 2017*. ACM, 88–100. https://doi.org/10.1145/3143361.3143383

[42] David Naylor, Kyle Schomp, Matteo Varvello, Ilias Leontiadis, Jeremy Blackburn, Diego R. López, Konstantina Papagiannaki, Pablo Rodríguez Rodríguez, and Peter Steenkiste. 2015. Multi-Context TLS (mcTLS): Enabling Secure In-Network Functionality in TLS. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, SIGCOMM 2015, London, United Kingdom, August 17-21, 2015*, Steve Uhlig, Olaf Maennel, Brad Karp, and Jitendra Padhye (Eds.). ACM, 199–212. https://doi.org/10.1145/2785956.2787482

[43] Jianting Ning, Xinyi Huang, Geong Sen Poh, Shengmin Xu, Jia-Ch'ng Loh, Jian Weng, and Robert H. Deng. 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, September 14-18, 2020, Proceedings, Part I (Lecture Notes in Computer Science), Liqun Chen, Ninghui Li, Kaitai Liang, and Steve A. Schneider (Eds.), Vol. 12308. Springer, 3–22. https://doi.org/10.1007/978-3-030-58951-6_1

[44] Jianting Ning, Geong Sen Poh, Jia-Ch'ng Loh, Jason Chia, and Ee-Chien Chang. 2019. PrivDPI: Privacy-Preserving Encrypted Traffic Inspection with Reusable Obfuscated Rules. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*, Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz (Eds.). ACM, 1657–1670. https://doi.org/10.1145/3319535.3354204

[45] Jianting Ning, Jia Xu, Kaitai Liang, Fan Zhang, and Ee-Chien Chang. 2019. Passive Attacks Against Searchable Encryption. *IEEE Trans. Information Forensics and Security* 14, 3 (2019), 789–802. https://doi.org/10.1109/TIFS.2018.2866321

[46] Rishabh Poddar, Chang Lan, Raluca Ada Popa, and Sylvia Ratnasamy. 2018. SafeBricks: Shielding Network Functions in the Cloud. In *15th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2018, Renton, WA, USA, April 9-11, 2018*, Sujata Banerjee and Srinivasan Seshan (Eds.). USENIX Association, 201–216. https://www.usenix.org/conference/nsdi18/presentation/poddar

[47] Hao Ren, Hongwei Li, Dongxiao Liu, Guowen Xu, Nan Cheng, and Xuemin Sherman Shen. 2020. Privacy-preserving Efficient Verifiable Deep Packet Inspection for Cloud-assisted Middlebox. *IEEE Transactions on Cloud Computing* (2020).

[48] Google Transparency Report. [n. d.]. HTTPS encryption on the web. https://transparencyreport.google.com/https/overview accessed November 2020.

[49] Daniel Roethlisberger and contributors. 2018. SSLSpit. https://www.roe.ch/SSLsplit.

[50] Scott Ruoti, Jeff Andersen, Tyler Monson, Daniel Zappala, and Kent E. Seamons. 2018. A Comparative Usability Study of Key Management in Secure Email. In *Fourteenth Symposium on Usable Privacy and Security, SOUPS 2018, Baltimore, MD, USA, August 12-14, 2018.*, Mary Ellen Zurko and Heather Richter Lipford (Eds.). USENIX Association, 375–394. https://www.usenix.org/conference/soups2018/presentation/ruoti

[51] Justine Sherry. 2016. *Middleboxes as a Cloud Service*. Ph.D. Dissertation. Electrical Engineering and Computer Sciences, University of California at Berkeley.

[52] Justine Sherry, Shaddi Hasan, Colin Scott, Arvind Krishnamurthy, Sylvia Ratnasamy, and Vyas Sekar. 2012. Making middleboxes someone else's problem: network processing as a cloud service. In *ACM SIGCOMM 2012 Conference, SIGCOMM '12, Helsinki, Finland - August 13 - 17, 2012*, Lars Eggert, Jörg Ott, Venkata N. Padmanabhan, and George Varghese (Eds.). ACM, 13–24. https://doi.org/10.1145/2342356.2342359

[53] Justine Sherry, Chang Lan, Raluca Ada Popa, and Sylvia Ratnasamy. 2015. BlindBox: Deep Packet Inspection over Encrypted Traffic. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, SIGCOMM 2015, London, United Kingdom, August 17-21, 2015*, Steve Uhlig, Olaf Maennel, Brad Karp, and Jitendra Padhye (Eds.). ACM, 213–226. https://doi.org/10.1145/2785956.2787502

[54] Symantec. 2018. Encrypted Traffic Management. https://www.symantec.com/products/encrypted-traffic-management.

[55] T-Mobile. accessed Dec 2020. T-Mobile Secure Web. https://networking.t-mobile.com/solutions/secure-web/.

[56] Bohdan Trach, Alfred Krohmer, Franz Gregor, Sergei Arnautov, Pramod Bhatotia, and Christof Fetzer. 2018. ShieldBox: Secure Middleboxes using Shielded Execution. In *Proceedings of the Symposium on SDN Research, SOSR 2018, Los Angeles, CA, USA, March 28-29, 2018*. ACM, 2:1–2:14. https://doi.org/10.1145/3185467.3185469

[57] Anton Tyurin. 2018. Detect Malicious Communications Even Under TLS. SuriCon 2018. https://suricon.net/wp-content/uploads/2019/01/SuriCon2018_Tyurin.pdf.

[58] Louis Waked, Mohammad Mannan, and Amr M. Youssef. 2018. The Sorry State of TLS Security in Enterprise Interception Appliances. *CoRR* abs/1809.08729 (2018). arXiv:1809.08729 http://arxiv.org/abs/1809.08729

[59] Cong Wang, Xingliang Yuan, Yong Cui, and Kui Ren. 2018. Toward Secure Outsourced Middlebox Services: Practices, Challenges, and Beyond. *IEEE Network* 32, 1 (2018), 166–171. https://doi.org/10.1109/MNET.2017.1700060

[60] Akira Yamada, Yutaka Miyake, Keisuke Takemori, Ahren Studer, and Adrian Perrig. 2007. Intrusion Detection for Encrypted Web Accesses. In *21st International Conference on Advanced Information Networking and Applications (AINA 2007), Workshops Proceedings, Volume 1, May 21-23, 2007, Niagara Falls, Canada*. IEEE Computer Society, 569–576. https://doi.org/10.1109/AINAW.2007.212

[61] Xingliang Yuan, Xinyu Wang, Jianxiong Lin, and Cong Wang. 2016. Privacy-preserving deep packet inspection in outsourced middleboxes. In *35th Annual IEEE International Conference on Computer Communications, INFOCOM 2016, San Francisco, CA, USA, April 10-14, 2016*. IEEE, 1–9.

[62] Zscaler, Inc. accessed Dec 2020. Zscaler Powers Comprehensive Security Solution Offered by Deutsche Telekom. https://www.zscaler.com/press/zscaler-powers-comprehensive-security-solution-offered-deutsche-telekom