Chinese Journal of Electronics
vol. 33, no. 1, pp. 1–18, January 2024
https://doi.org/10.23919/cje.2022.00.170

REVIEW

Review on Security Defense Technology Research in Edge Computing Environment

Ke SHANG, Weizhen HE, and Shuai ZHANG

National Digital Switching System Engineering and Technological R&D Center, Zhengzhou 450001, China

Corresponding author: Weizhen HE, Email: heweizhen@alu.hit.edu.cn
Manuscript Received June 14, 2022; Accepted January 17, 2023

Abstract — Edge computing, which achieves quick data processing by sinking data computing and storage to the network edge, has grown rapidly along with the Internet of things. The new network architecture of edge computing brings new security challenges. Based on this, this paper investigates the edge computing security literature published in recent years and summarizes and analyzes research work on edge computing security from different attack surfaces. We start with the definition and architecture of edge computing. From the attack surface between device and edge server, as well as on edge servers, the research describes the security threats and defense methods of edge computing. In addition, the cause of the attack and the pros and cons of defense methods is introduced. The challenges and future research directions of edge computing are given.
Keywords — Edge computing, Security, Threat, Defense, Architecture, Attack surface.
Citation — Ke SHANG, Weizhen HE, Shuai ZHANG, "Review on Security Defense Technology Research in Edge Computing Environment," *Chinese Journal of Electronics*, vol. 33, no. 1, pp. 1–18, 2024. doi: 10.23919/cje.2022.00.170.

I. Introduction

Currently, the number of mobile and Internet of things (IoT) devices deployed in the physical world, as well as the data they produce, are surging due to the rapid growth of IoT and 5G technologies. According to the prediction of the Internet data center (IDC) [1], the data generated by IoT devices will increase from 33ZB in 2018 to 175ZB in 2025. For traditional cloud computing network architecture, they collect and process data generated by IoT devices in cloud data centers. However, the explosion of massive data makes it face two challenges. For one thing, the exponential growth of IoT devices will exhaust the bandwidth of cloud computing networks, making it impossible to provide high-quality services. For another thing, the frequent transmission of device data may lead to device information leakage, which increases the possibility of the device being attacked.

Edge computing was created to cope with the challenges. It provides more stable and reliable services for edge devices by migrating a part of computing, storage and network application functions of cloud data centers to the network edge, which solves the problems of high latency, heavy computing, storage load, and heavy network bandwidth caused by cloud data centers and devices during data transmission. Meantime, data processing on the network edge side is closer to devices, which can reduce the risk of device information leakage [2]. Therefore, edge computing architecture has received extensive attention from domestic and foreign researchers after it is proposed. However, while edge computing solves many problems of cloud computing architecture, it also brings many security threats. Since edge computing platform is always deployed on edge servers with limited resources, complex environments, and heterogeneous networks, some new edge computing security defense methods are called to research to deal with cyber-attacks on edge computing.

In order to summarize these edge computing security defense technologies, some literature has reviewed the current edge computing defense mechanisms. For example, reference [3] investigates the edge computing security protocols in recent years and gives some existing problems and future development directions. Reference [4] summarizes the attack methods and defense methods faced by edge computing. Both of these literature has in-

Associate Editor: Prof. Zhi-hua ZHOU, Nanjing University.

troduced the security defense technology trends of edge computing based on investigating the current situation of edge computing security defense, but, their overview methods are not fine-grained enough. As for reference [5], it only summarized five defense methods of edge computing in terms of access control, privacy protection, attack mitigation, key management, and anomaly detection. Readers cannot choose their defense method by determining the attack method according to the specific attack surface. Therefore, this article will summarize the attack methods and defense methods of edge computing according to the attack surfaces of edge computing.

The rest of the review is divided into the following sections. We introduce the idea and fundamental architecture of edge computing in Section II. After that, Section III presents the current mainstream edge computing attack methods from the aspects of cloud side to edge side, edge side to device side, and edge side server. The current defense methods between edge servers and edge to the device are given in Section IV based on Section III. Finally, we outline the future directions for security defense technologies of edge computing in Section V.

II. Concepts and Architecture of Edge Computing

1. The concept of edge computing

The concept of edge computing originated from the platform proposed by IBM and Nokia Siemens Network in 2013, which can run applications in mobile base stations [6]. The platform can provide services to mobile users. After that, with the continuous iterative upgrading of technology, the concept of edge computing has been constantly perfected.

In the *Mobile Edge Computing Industrial Technology White Paper* [7], which was released by ETSI in 2014, it is proposed that "Edge computing is designed to complete computing at the edge of the wireless access network instead of the mobile network data center ". In September 2015, ESTI further standardized the definition of mobile edge computing in the *Mobile Edge Computing: A key Technology Towards 5G* standard [8] released. In the wireless access network close to the user's mobile terminal, mobile edge computing offers an IT services eco system and cloud processing capability with the goal of lowering latency, enhancing network operation efficiency, enhancing service distribution capabilities, and enhancing user experience.

In November 2016, the Edge Computing Industry Alliance released the *Edge Computing Industry Alliance White Paper* [9]. In this white paper, edge computing is described as being on the edge of the network, near to objects or data sources. In terms of agile links, real-time business, data optimization, application intelligence, security, and privacy protection, it is a distributed open platform that combines network, computing, storage, and core application capabilities. It also offers edge intelligent services close by and satisfies the essential requirements of industry digitization.

In December 2018, Alibaba Cloud and China Electronics Standardization Institute released the *Edge Cloud Computing Technology and Standardization White Paper* [10]. According to the study, edge cloud computing, also abbreviated as edge cloud, is a cloud platform technology created on the edge infrastructure using the foundation of cloud computing technology and the advantages of edge computing. With regard to its compute, network, storage, and other characteristics, this platform is an elastic cloud platform.

With a central cloud and IoT terminals, it creates an end-to-end technical architecture of "Threebody collaboration between cloud, edge and terminals ". It decreases reaction times, cloud pressure, and bandwidth usage by bringing network forwarding, storage, computation, and intelligent data analysis to the edge while also delivering cloud services like network-wide scheduling and processing power distribution.

In 2020, the International Organization for Standardization (ISO) mentioned in the ISO/IEC TR 23188 standard [11] that edge computing is a platform for data processing and storage close to terminals.

Summarizing the definition of edge computing in recent years, it can be seen that although different organizations have different definitions of edge computing, they have something in common. In this paper, we define edge computing as a distributed computing platform which is different from cloud computing. It provides cloud services at the edge side where close to users or data sources by migrating the networking, compute, storing and resources from the cloud data center to edge servers. Additionally, it satisfies the demands of the IT sector in terms of real-time business, flexible linkages, data optimization, application intelligence, security, and privacy demands while offering high bandwidth and low network latency.

2. The basic architecture of edge computing

Unlike the cloud computing architecture, edge computing brings in an edge devices layer between the device layer and cloud layer, thereby providing high bandwidth and low network latency services. Figure 1 shows the basic architecture of edge computing. In this paper, we divide the edge computing into three layers, namely Device layer, Edge service layer, and Cloud computing service layer. In Figure 1, in edge computing environment, devices access edge service layer through access points and base stations of edge service layer (black connection line); and in cloud computing environment, devices access cloud computing service layer through access points and base stations of cloud computing service layer (blue connection line).

The Device layer (DL) is mainly responsible for deploying edge devices used for sensing, computing and controlling, mainly including IoT equipment and mobile

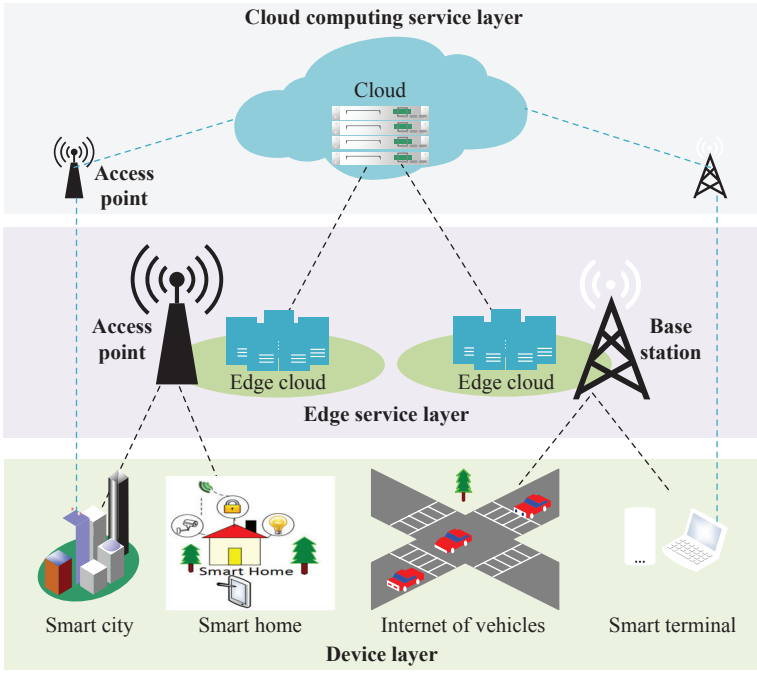


Figure 1 The architecture of edge computing and cloud computing.

equipment. IoT equipments are some lightweight devices with microcontroller. And it uses 4G/5G, WiFi, Bluetooth and other wireless protocols to connect to the edge servers. It can realize functions such as environmental perception, control as well as data processing by running on a real-time operating system. Typical IoT devices include home automation devices, health monitoring devices, smart wearable devices, etc. As opposed to IoT devices, mobile devices, such as smartphones and tablets, operate on more sophisticated operating systems, like Android and IOS, which give users customizable interfaces that make it simpler to modify programs.

The edge service layer (ESL) is mainly responsible for accessing edge devices downward, connecting to the cloud upward. In addition to storing and processing the data that was uploaded by the terminal device, it also simultaneously submits its own processed data to the cloud data center. As the core of the three-layer architecture, the ESL represents the compute nodes, which is deployed between CCSL and DL, is closer to devices. Typically, ESL represents the equipments like access points, routers, base stations, switches and so on. The ESL can more effectively address the latency, bandwidth, and security demands of edge devices since it is deployed at the network's edge.

The cloud computing service layer (CCSL) has higher performance servers and storage devices than the ESL. It mainly completes analysis tasks and integration tasks that the ESL cannot handle, and it is also responsible for storing the data reported by the ESL. Furthermore, the algorithm and strategy for service development of ESL can be controlled by CCSL adaptively.

3. Edge Computing Reference Architecture 3.0

In the *Edge computing White Paper* [12] published in 2018, the edge Computing Consortium (ECC), which is made up of well-known companies like Huawei, Shenyang Institute of Automation, the Chinese Academy of Sciences, and the China Academy of Information and Communications Technology, proposed the Edge Computing Reference Architecture 3.0. Just as Figure 2 displays, this architecture shows functions of each layer from different perspectives.

As displayed in Figure 2, the Management service layer, Data lifecycle service layer and Security service layer cut cross the entire framework, which can offer essential services. The key duties of the Management service layer are unified management for edge computing, architectural operation monitoring, and monitoring data delivery to the management platform. Providing integrated management for data pretreatment, analysis, dissemination, execution, visualization, and storage is primarily the responsibility of the Data lifecycle service layer. By utilizing the Business orchestration layer, the Security service layer may specify the logic of the data processing. Moreover, It can deploy and optimize data services flexibly to satisfy the business requirement of real-time. The Security service layer covers the whole edge computing architecture, which ensure safe and stable operation of system by using the unified security administration and perception mechanism. The Unified model-driven service framework, which is at the top of the architecture and is based on its vertical structure, is utilized to offer service creation and deployment. The Edge Computing Reference Architecture 3.0 also divides edge computing into 3 layers: Cloud layer, Edge layer and Device layer. Edge node and Edge manager make up the majority of the Edge layer. Edge nodes carry edge computing services on physical hardware, and Edge man-

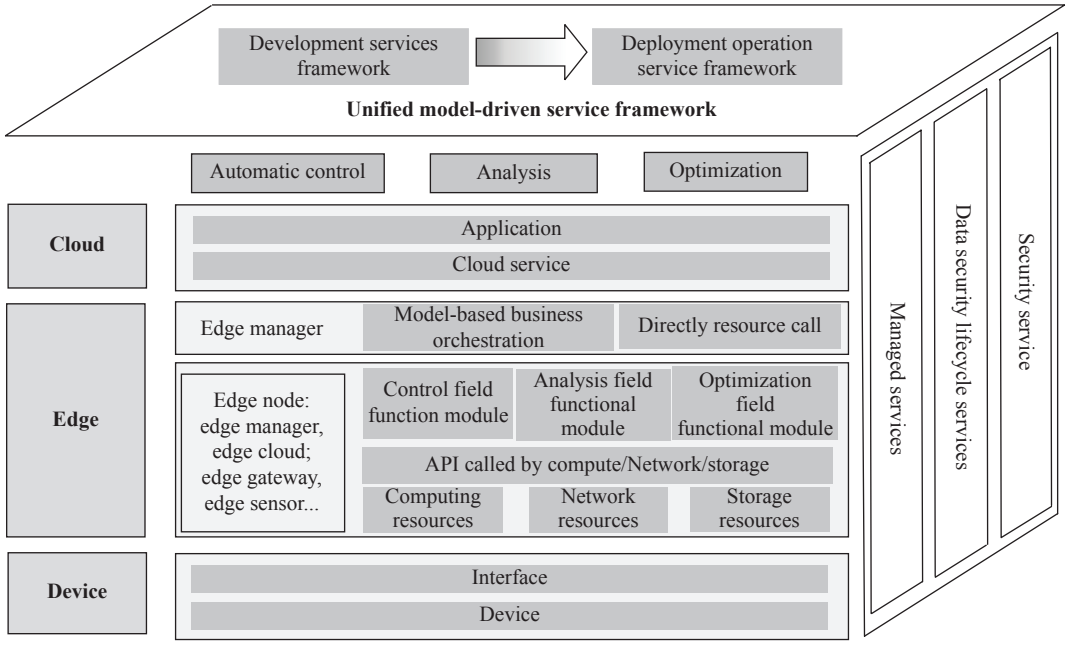


Figure 2 Edge Computing Reference Architecture 3.0.

agers use software to to manage Edge nodes uniformly.

4. EdgeX Foundry

A generic open architecture for IoT edge computing can be found in the open source EdgeX Foundry project, which is sponsored by the Linux Foundation. The framework enables the universal computing platform to be integrated at the IoT edge and speeds up the deployment of solutions through a plug-and-play element ecosystem. It is hosted on a reference software platform, which is devoid of any hardware or operating system dependencies.

EdgeX Foundry is primarily divided into Southbound and Northbound, as shown in Figure 3. The southbound mostly consists of edge servers and IoT equipments that connect with the edge servers and gather data from it. The northbound specifically consists of the cloud for data collection, storage, aggregation, analysis and conversion into information, and the network part for communicating with the cloud. EdgeX Foundry is split into 4 microservice layers and 2 enhanced basis system service layers, and it is situated between southbound and northbound traffic. From bottom to top, the four microservice layers are the Device services layer, the Core services layer, the Supporting services layer, and the Application services layer. The Device services layer proffers users with software development kits to link the southbound devices. It primarily functions to transfer data from IoT devices to the Core services layer. Additionally, it can broadcast commands to devices after receiving them from microservices. As the core of EdgeX Foundry, the Core services layer can be divided into four microservice components, mainly including Core data, Metadata, Command, Confign&Registry. The Core data service is mainly used to store and manage the data from the device. The Command service is primarily used to cache and manage the interactive requests between southbound and northbound. The Metadata service is mainly used to provide pairing for devices and services. The Config&Registry services are mainly used to provide configuration information for other microservices. Edge analytics and intelligence services is offered by the Supporting services layer. Besides, it also provides Rules engine, Scheduler, Alarms, and logging for the framework itself. The Application service layer can connect to the CCSL, transmit data to the CCSL, and ensure the individual operation of EdgeX Foundry. The Management service layer and Security service layer are the Core service layers of the EdgeX Foundry architecture, which is similar to the Edge Computing Reference Architecture 3.0. The Management service layer performs tasks including setting up, updating, initiating, stopping, and keeping track of EdgeX Foundry processes, among others. Data and device functionality are protected by the Security service layer.

III. Security Threats

According to the basic architecture of edge computing in Section II of this paper, it is clear that the majority of the edge computing attack surface is found on the edge server, between the DL and the ESL, and between the CCSL and the ESL. Since the security threats of the CCSL and the ESL are similar to those of cloud computing, this section does not further introduce the security threats between the cloud and the edge, but focuses on the security threats between ESL and DL and threats on ESL. The specific attack methods and their main causes and hazards are shown in Table 1.

1. Security threats between ESL and DL

In view of the security threat between the DL and ESL, since IoT devices access the edge servers through

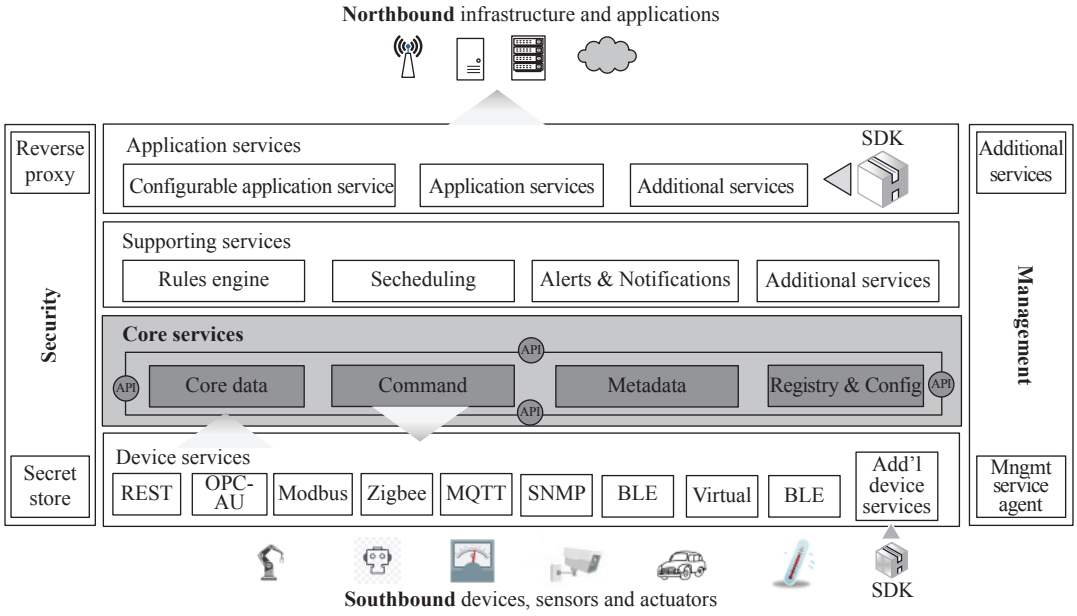


Figure 3 The architecture of EdgeX Foundry.

Table 1 Causes and main hazards of edge computing security threats

Attack surface	Attack method	Main reason	Major hazards	Related literature
The attack type between the Edge and Device	Authentication and authorization attacks	Vulnerabilities in authentication and authorization protocols	Control host permissions	[13]–[20]
	Side channel attack	Hidden correlation between sensitive data and publicly available side-channel information	Leak sensitive data	[21]–[30]
	Device-based malicious injection attack	Code-level design vulnerabilities and coarse-grained access control models for devices	Device hijacking, malicious injection	[31]–[36]
The attack type of edge server	Device-based DDoS attacks	Huge number of devices and widespread vulnerabilities	Edge server resources are exhausted and cannot provide services	[37], [38]
	Malicious injection attacks based on edge servers	Vulnerabilities of interaction protocols	Access and tamper edge server data	[39]–[42]
The attack type of between the Cloud and Edge	Access control attack	–	–	–

communication protocols, there is an attack surface between the DL and the ESL for attackers to exploit. As a result, we focus on attacks on the IoT and those brought on via device access. The attack methods mainly include authentication and authorization attacks, side-channel attacks and malicious injection attacks based on edge devices.

1) Authentication and authorization attacks

In edge computing, the edge device needs to complete two steps before obtaining the edge server information. Firstly, the edge node needs to complete the authentication before accessing the edge server to prevent malicious and illegal IoT devices from accessing the ESL and launching attack. Secondly, after the edge device is connected to the ESL, when the edge device initiates a request to the ESL, the edge device needs to complete the authorization of the edge server, and then obtain certain access rights. For the authentication and authorization process of edge devices, encryption algorithms are generally used to prevent attackers from eavesdropping and man-in-the-middle attacks. However, some existed methods still can attack the authentication and authorization process between the DL and ESL.

The simplest and most direct attack on the authentication process between the DL and ESL is the dictionary attack. A dictionary attack refers to the brute force search method. Attackers input all potential credentials/passwords into targeted system according to its dictionary, and then observe identified matches [13]. In order to apply the dictionary attack to the authentication process between DL and ESL, Nam *et al.* conducted research and found that it's quite easy for offline dictio-

nary attackers to attack the 3 password-authenticated key exchange (S-3PAKE) protocol used by Bluetooth when the two sides of communication establish the initial session key. An attacker can enumerate all possible passwords offline to determine the correct one. But this kind of single-thread dictionary attack is very time-consuming, so Nakhila *et al.* proposed a parallel multi-threaded dictionary attack method to attack the WPA2-PSK WiFi network 100 times faster than the traditional single-thread dictionary attack. However, no matter what kind of attack type the dictionary attack uses, it is so difficult to balance the attack overhead and success rate. Hence, it is necessary to study the vulnerability of the authentication protocol to increase the probability of the attacker's successful attack. In this regard, based on discovered weak binding vulnerability of WPA enterprise authentication protocol, Cassola *et al.* [14] describe a innovative evil twinning attack. Bhargavan *et al.* [15] discovered a transcription collision attack against the TLS authentication protocol. Zhu *et al.* [16] proposed a one-cycle attack against sensor-based authentication, which can bypass current gait authentication.

Additionally, the attacker can use the authorisation protocol's vulnerability to carry out the assault on the devices' authorization process. For example, in edge computing, the authorization protocol mostly adopts the OAUTH Protocol [17], [18], which enables third-party users to obtain authorization of resources without using user names and passwords, so the OAUTH Protocol has become the focus of attackers. In 2009, the OAUTH 1.0 protocol was revealed to have loopholes. Attackers could access the victim's private resources through the accessor's website [19]. Based on this, OAUTH has launched the 2.0 protocol, which is theoretically proven to be secure, but some wrong implementations can still lead to the authentication process being exploited by attackers. For example, Sun *et al.* [20] presented several key vulnerabilities of the OAUTH protocol based on the analysis of 96 vendor-dependent OAUTH single sign-on systems, which can enable attackers to take advantage of victims' resources without their consent.

2) Side channel attacks

By exploiting side channel information, which is readily accessible information that is not privacy-sensitive, side channel assaults put a user's security and privacy at risk. Attackers then investigate the covert correlations to deduce the side channels' protected data. In edge computing, attackers can detect sensitive information of edge servers and devices by exploiting the abundant open information between DL and ESL, and then launch further attacks on edge servers or edge devices based on this sensitive information. Currently, typical side-channel attacks between DL and ESL mainly use the communication signals between edge server and devices, as well as the power consumption of edge devices.

For side-channel attacks based on the communication signal between DL and ESL, the attacker usually uses the communication data signal and the waveform signal to detect the sensitive information of the system. In the way of using communication data signals to infer sensitive data, Li *et al.* [21] used a differential coding scheme in video surveillance to infer four standard human daily activities defined by HIPAA. Apthorpe *et al.* [22] infer the behavior of smart home users by exploiting the encrypted traffic of IoT. Chen *et al.* [23] performed a TCP packet injection attack by exploiting the timing channel vulnerability in wireless routers. A original flow statistics-based supervised approach to infer the network service of IoT was introduced by Lopez-Martin *et al.* [24]. The suggested method first makes use of a number of features taken from packet headers before offering enhanced detection outcomes. Acar *et al.* [25] introduced a novel multi-stage privacy attack against user privacy. By just passively analyzing the traffic between ESL and smart home devices and sensors, the approach inferred the categories of IoT devices, their statuses, and human behaviour. In terms of using waveform signals to infer sensitive data, Enev *et al.* [26] used electromagnetic interference signals released by TV power to infer video content. Yan *et al.* [27] infer entry passwords on mobile devices from WiFi-based side-channel information.

In addition, since devices have different energy consumption when performing calculations and operations, attackers consider using device power consumption to conduct side-channel attacks to obtain sensitive data of devices. Current side-channel attacks based on device power consumption mainly use smart meter data and oscilloscope data. It's accurate for smart meters to record the energy consumption of devices, so their data is widely used by attackers to infer user behavior. As early as 1992, the non-embedded device load monitoring system proposed by Hart *et al.* [28] could monitor device state according to the energy consumption of a single device, but the system has not been used by malicious attackers. Reference [29] introduced a improved non-embedded device load monitoring system. This system can obtain the data of household activities such as cuisine, television-watching, washing and game based on the power dissipation of smart meters. Moreover, attackers can use energy consumption data obtained from oscilloscopes to probe the encryption keys of embedded devices. For example, reference [30] proposes to use correlation power analysis to obtain the AES-CCM master key, which is installed into firmware of Philips Hue smart lights.

3) Device-based malicious injection attacks

Malicious software installation or injection into a computer system is referred to as a malicious injection attack. On traditional Internet devices or general-purpose computers, high-performance firewalls and threat protection systems can effectively defend against such attacks. However, in the context of edge computing, because IoT devices have restricted computing capabilities, high-security protection systems cannot be installed, moreover, edge devices with a high degree of heterogene-

ity in both hardware and firmware make IoT devices more vulnerable to malicious injection attacks. There are different kinds of zero-day vulnerabilities to help attackers inject malicious software into IoT devices. By leveraging these vulnerabilities, attackers can initiate remote code execution (RCE) attack or command injection attack. For instance, the IoT Reaper, which was found out in 2017, infected millions of IoT equipments by utilizing Internet protocol and wireless protocol. Based on at least 30 RCE vulnerabilities, which come from 9 different IoT equipments, such as network routers, IP cameras and so on, the virus perform remote malware injection into devices [31]. In addition, there are also malicious injection attacks on edge devices in academia. For example, reference [32] describes that based on exist vulnerability of the firmware update mechanism deployed on Logitech G600 mouse, it's not difficult for attackers to inject firmware through the network or USB. Reference [30] learned how to remotely and non-contact implant malicious firmware onto IoT equipments via the Zigbee Light Link Protocol.

However, it is not an easy task to inject malware with cross-access capability into mobile devices. Major mobile device operating systems such as iOS and Android use a sandbox mechanism to ensure virtual isolation of each application in memory. Applications cannot access the resources and content of other applications without kernel-level permissions, so academia has also carried out related research on this limitation. Wang *et al.* [33] identified APIs that may have permissions for the injection created by attackers that toward other third-party applications by summarizing legitimate API calls at an early stage. After that, an OS-level structure which name is Android task architecture (ATM) is used by Ren *et al.* [34], which can inject malicious UI into applications passively. Ren's work has been improved by Xiao *et al.* [35] used the active attacks of ATS, making the injection attacks based on ATS becomes more available in practice. Although these attack methods have an impact on IoT devices that use IOS and Andriod as the operating system, they cannot make significant damage to the edge computing infrastructure. Attackers frequently pick malicious third-party libraries because they are more potent and hard to spot. For example, harmful libraries are founded by Chen *et al.* [36] in both offcial Android apps and Apple App Store. According to Google Play Store and Apple App, there are 6.84% Android apps and 2.94% iOS apps are using harmful libraries. By exploiting these malicious libraries, attackers can easily inject code into IoT devices.

2. Security threats of edge servers

Different from the attack methods between DL and ESL, the security threats of the edge server mainly consider the attacker disguised as a legitimate user or using an IoT device to attack the edge server. The main attack methods include device-based DDoS attacks and edge server-based malicious injection attacks.

1) Device-based DDoS attacks

Compared with cloud computing servers, edge computing servers have fewer computing and storage resources, so edge servers are more vulnerable to DDoS attacks and affect normal services. Usually, the attacker floods the edge server with malicious packets such as ICMP, UDP, SYN, and HTTP through the compromised edge device cluster to exhaust the resources of the edge server, resulting in the edge server's inability to handle requests from normal edge devices. For example, the Mirai botnet [37] launched DDoS attacks on KrebsOnSecurity, OVH, and Dyn edge service providers by controlling 65,000 edge devices, resulting in the failure of the edge service providers to provide normal services. Similar to this, there are variants of the Mirai botnet, including Hajime, BrickerBot [38].

2) Malicious injection attacks based on edge servers

In edge computing, the WEB applications accessed by the user can be deployed on the edge server, so there are also various attack methods for the WEB applications on the edge server. Malicious injection attacks against edge servers mainly include SQL injection attacks, cross-site scripting (XSS), cross-site request forgery (CSRF), server-side request forgery (CSRF) attacks, and XML signature wrapping attacks.

SQL injection is an attack method that occurs in the back-end database of a WEB application. Attackers take advantage of the design flaw that the applications do not detect the validity of user input data, and insert illegal SQL statements into pre-defined SQL statements of WEB applications by querying or tampering database, and then, obtaining all database informations. In edge computing, there is a database for storing application information, therefore, SQL injection can be a security threat in edge computing [39]. Similar to SQL injection attacks, cross-site scripting attacks also utilize the inherent flaw of edge services that do not verify legitimate code input. By injecting malicious scripts into the program, the browser triggers the execution of malicious scripts when accessing web pages, which leads to the disclosure of application information [40].

The similarity between CSRF attack and SSRF attack is the exploit of weak authentication flaws applied on edge servers. Attackers can send requests to else edge servers by pretending as a valid edge server, which may lead to SSFR and CSRF attack suffered by those edge servers. Currently, SSRF and CSRF are usually aimed at classical WEB programs, but reference [41] also lists related attack methods for edge computing.

XML signature wrapping attack [42] occurs in the communication protocol used by edge computing infrastructure, namely Simple Object Access Protocol (SOAP). Since the protocol transmits messages in the Extensible Markup Language (XML) format, an attacker can intercept a legitimate XML message, create a new sticker, and insert the primary message's transcript (along with

authentication parameters like tokens) into the new tag (also known as a wrapper), resulting in one massive tag-value pair. Attackers then alter the main component of the original message with malicious code. The huge tag-value pair and the modified original message are then merged and sent to the edge server. Once the edge server has successfully verified these altered messages after receiving them, attackers will insert malicious code they have written into it.

3. Security threats between cloud and edge

Aiming at the security threat between CCSL and ESL, since edge computing sinks the computing and storage of data to the ESL and the CCSL manages the ESL uniformly, attackers usually exploit vulnerabilities in the CCSL to attack the ESL. For example, attackers use illegal access control to data between CCSL and ESL to intrude on the cloud platform and then illegally operate the resources of the ESL, or the attacker uses unsafe API interfaces to achieve illegal access to cloud services. These security threats all use the security vulnerabilities of the Cloud Service Layer to initiate attacks on edge computing. Therefore, such security threat scenarios will not be discussed in this paper.

4. Summary of security threat

The characteristics and shortcomings of edge computing security threats in Section III will be summarized as follows, which are mainly divided into the following aspects.

1) Authentication and authorization protocol vulnerability detection method should be explored based on the artificial intelligence technology. The authentication and authorisation processes are vulnerable to the existing dictionary attack's high overhead and ineffectiveness. Therefore, finding loopholes in the authentication and authorization protocols is the main attack method of authentication and authorization attacks. However, it is not easy to find loopholes at the protocol level, which demands a significant amount of material and human resources. In order to achieve the automatic discovery of protocol flaws, artificial intelligence technologies must be taken into consideration.

2) Side-channel attacks are stealthy and effective methods of attack. In edge computing, the interaction signals between DL and ESL and the power consumption of edge devices become the main battlefield of side-channel attacks. Attackers can use public information to infer the sensitive data of the system without accessing the system. Since this type of attack has strong stealth and strong lethality in non-contact attacks, the research on this kind of threat becomes the focus of researchers.

3) Device firmware vulnerabilities are the most serious security threat in edge computing. At present, because of the enormous quantity and heterogeneity of edge computing equipments, firmware is different from traditional computer systems with mature vulnerability detection and system protection technologies. Most firmware lacks effective security protection measures, once the firmware vulnerabilities of edge devices are exploited, they will quickly spread and cause serious losses. The device-based malicious injection attack and device-based DDoS attack proposed in this section are both attacks launched by exploiting firmware vulnerabilities of edge computing devices, so how to detect firmware vulnerabilities becomes a future research goal.

4) Flood DDoS attacks are replaced by advanced zero-day DDoS attacks. The current typical DDoS attack method of edge computing uses flooding TCP, ICMP, ARP, and HTTP to consume the resources of the edge server. However, this method is easy to be discovered by defenders. Therefore, researching new DDoS attack methods has become an urgent need for attackers. As an advanced attacker method, zero-day attacks can effectively evade defensive methods by discovering zero-day vulnerabilities running on edge servers to conduct targeted attacks.

5) Malicious injection methods based on encrypted applications should be studied. Most of the current malicious injection attacks based on edge servers are mainly aimed at non-encrypted applications, which cannot be effectively attacked on encrypted applications. Currently, most web applications use encryption to defend against malicious injection by attackers. However, most of the current malicious injection attacks based on edge servers are mainly aimed at non-encrypted applications and thus cannot effectively attack encrypted applications. Therefore, it is meaningful to study malicious injection attack methods for encrypted web applications.

IV. Security Defense Technology

This section mainly introduces the defense methods of edge computing security threats given in Section III, including the security defense methods between DL and ESL and the security defense methods on the edge server. Regarding the security defense method between CCSL and ESL, this section will not go into details about the security threat and security defense technology between CCSL and ESL since they are similar to cloud computing. The principle of the method and its advantages and disadvantages are shown in Table 2. In addition, in order for the reader to clearly understand the sequential relationship between the various types of the security defense algorithms, we give the development timeline of various security defense methods as depicted in Figure 4.

1. Security defense technology between ESL and DL

1) Defense techniques for authentication and authorization attacks

From the analysis in Section III, it can be concluded that there are two main types of attacks on authentication and authorization, namely dictionary attacks and the protocol attacks on authentication and authorization. Dictionary attacks use the weak credentials of the au-

Table 2 The principle of defense method and its advantages and disadvantages

Attack types	Defense methods	Main principles	Main advantages	Main disadvantages	Related literatures
Authentication and authorization attacks	Complex password	Use a more complex password	Increased search difficulty for attacks	Increased storage and computing pressure on edge devices	–
	Two-tier authentication method	Use two-tier authentication	Increased cost of dictionary attack	Requires human interaction	[43]–[51]
	Enhanced protocol	Increase the security of the protocol	Increased the security of the protocol itself	There are vulnerabilities unknown to the protocol itself	[52]–[60]
Side channel attack	Hide sensitive data	Enhancing the diversity of sensitive data	Enhanced data diversity	There is still a certain probability that the attacker can infer sensitive data	[61]–[65]
	Hide traffic features	Packet modification and repackaging	Obfuscate normal traffic	Increase communication delay	[25], [66]–[69]
	Disable side channel access	Source code level side channel obfuscation	To a certain extent, prevent attackers from gaining side channel access	Too difficult to achieve	[70]–[71]
Device-based malicious injection attacks	Pre-run firmware vulnerability detection	Vulnerability detection before device operation using taint analysis, symbolic analysis and simulation software	Effectively discover vulnerabilities in firmware and prevent malicious injection	Inability to defend against vulnerabilities generated by device runtime	[72]–[78]
	Runtime firmware hardening methods	Reduce firmware code attack surface	Effectively prevent attackers from exploiting vulnerabilities in firmware runtime	Implementation is complicated	[79]–[82]
Device-based DDoS attacks	Packet-level-based traffic filtering detection method	Detect malicious or malformed packets flooded by malicious attackers	DDoS attacks can be detected to a certain extent	Inability to detect advanced DDoS attackers	[14]
	Statistical detection method based on flow level	Use entropy and machine learning methods to detect data flow and find malicious attackers	Can detect advanced DDoS attacks	Overfitting problem	[83]–[88]
Malicious injection based on edge servers	Edge server security defense	Enhance the security of WEB application protocol	Can effectively defend attacks against WEB applications	There are fewer defense methods for new WEB application attack types	[89]–[98]

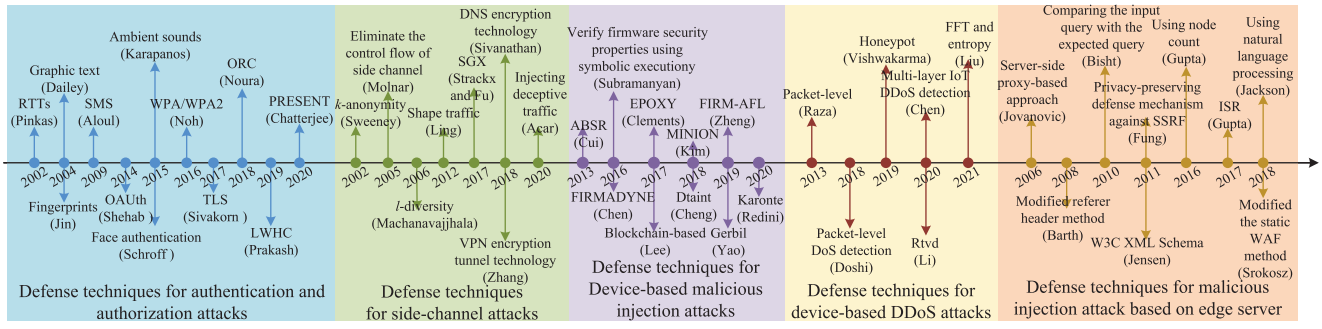


Figure 4 The development timeline of the security defense methods in edge computing.

thentication protocol to attack the authentication process. In response to this attack method, edge computing vendors and academia initially considered using complex passwords to block dictionary attacks, but the method has three limitations. Firstly, the edge computing infrastructure has limited computing resources and cannot use additional resources to calculate complex passwords. Secondly, a large number of edge users make it impossible for edge devices to store numerous complex passwords. Thirdly, there are certain security risks in storing complex passwords on edge devices. Therefore, researchers consider new defense methods to deal with dictionary attacks. One is to add a layer of authentication processes. For example, Pinkas *et al.* [43] proposed to effectively combine traditional password authentication with challenges that are easily answered by human users to in-

crease the cost of dictionary attacks. Some studies use fingerprints [44], face authentication [45], authentication codes via SMS messages [46], graphic text [47], and ambient sounds [48] as second-layer authentication methods. This method increases the cost of dictionary attacks to a certain extent, but also faces the following challenges. Firstly, these schemes require more or less manual interaction, which is unacceptable for edge computing automation equipment. Secondly, the literature [49]–[51] proves that the two-layer authentication scheme is not completely secure.

Another attack method of the authentication and authorization process is the attack on the authentication and authorization protocol. The root cause of this attack method is that the attacker exploits the vulnerability of the authentication and authorization protocol. So researchers defend the attack of the authentication and authorization protocol by enhancing the security of the protocol. For instance, in order to defend against attacks using authentication protocols, the academic community focus on methods which can enhance the security of WPA/WPA2 authentication protocols [52], [53] and TLS authentication protocols [54], [55]. As for attacks using authorization protocols vulnerability, defenders mainly enhance the security of the OAuth protocol [56], [57] to block attacks against authorization. Additionally, because edge computing platforms own limited resources, some research about lightweight cryptography are proposed to defend against authentication and authorization attack. For example, Prakash *et al.* [58] proposed a original lightweight encryption algorithm, which combines LED and RECTANGLE SPECK. It's faster, stronger and lightweight for IOT devices to deploy this cipher. On the basis of the original PRESENT cipher, Chatterjee *et al.* [59] updated the key and decreased the encoded in a novel lightweight PRESENT encryption. The lightweight cipher TEA (tiny encryption algorithm), which encrypts the value of the key register, is added as a delta value function to the key register to update it. The new layer aids in lowering the number of rounds from the current 31 to the minimum 25 needed for security. The performance of the given approach is enhanced by encrypting the key register. By examining several software factors including the N-gram, histogram, frequency distribution, and the non-homogeneity graph, the suggested approach demonstrates its superiority. Noura *et al.* [60] came up with the One Round Cipher, which is a simple cipher technique. It is constructed using an ORC, or one round roll, dynamic structure. A dynamic key is generated by this algorithm and applied to create two robust substitution tables, a dynamic permutation table, and two pseudo-random matrices. Authors in this article proved that the dynamic architecture with a single round were more random and safe and defend against statistical attacks and key-related attacks with experiments.

2) Defense techniques for side-channel attacks

Side channel attack between DL and ESL mainly uses the communication signal and the power consumption of the edge infrastructure to extrapolate the sensitive information. The main reason of this is that it is very complex and challenging to identify the hidden association between the protected information and the available exposed information. Therefore, by securing sensitive information and limiting side channel access, side-channel attacks can be defended against.

For the defense methods of hiding sensitive data, the most well-known method is the k-anonymity algorithm [61], which publishes data with lower precision through generalization and concealment technology. So the same identifier has k pieces of data corresponding to it, increasing the difficulty for attackers to obtain sensitive information through side channels. As for the k-anonymous algorithm, Ling *et al.* [62] proposed three defense strategies to defend the introduced side-channel attack based on network delay. The three ways include the statistical distribution-based approach, the k-means clustering-based strategy, and the k-anonymity-based approach. The tests show that the k-anonymity-based countermeasure can successfully strike a compromise between performance and data leakage. A practical secure deduplication mechanism based on k-Anonymity is proposed by Zhang *et al.* [63] with theoretical privacy guarantees. The protocol safeguards data security and ownership verification while defending against template side-channel assaults in the covert adversary paradigm. But the algorithm was found in 2006 that when the values of sensitive attributes in the equivalence class are the same, the k-anonymous algorithm suffers from homogeneity attacks [64]. Therefore, the l-diversity algorithm [64] appeared later, which ensures that at least l different sensitive attribute values appear in the same equivalence class, further enhancing the difficulty of inferring sensitive data. However, the l-diversity algorithm also has itself limitations. The implementation of this technique may be difficult and useless, as Li *et al.* [65] noted. Moreover, when the distribution of a certain value of the same sensitive attribute is significantly different from the distribution of other values, it is not enough to prevent attribute leakage. Based on this, to get around the l-diversity algorithm's drawbacks, Li *et al.* devised the t-closeness technique.

In addition, since attackers can use communication traffic [66], [67] to obtain sensitive information of devices and users, side-channel attacks can be defended by hiding traffic features, such as hiding header features and statistical features. As for hiding the header feature, it is to modify the packet information so that the attacker cannot obtain the useful packet header without affecting the normal data transmission. Typical technologies include VPN encryption tunnel technology [68] and DNS encryption technology [69]. Hiding statistical features is to hide the overall characteristics of traffic without affecting the normal function of the system. Typical techniques include injecting deceptive traffic between devices and edge servers to hide the actual activities of the de-

vice [25].

For the defense method that restricts access to the side channel, the current typical defense method is the side channel obfuscation at the source code level. For instance, Molnar *et al.* [70] suggested altering the C source code to remove the control flow of the side channel. Furthermore, the fast-moving development of TrustZone technology led to the generation of TrustZone-licensed hardware SGX [71], [72], which prevented attackers from accessing side channels that are not protected by TrustZone.

3) Defense technology of device-based malicious injection attack

For the device-based malicious injection attack introduced in Section III, The device's code-level design vulnerability and coarse-grained access control approach are the primary causes. Based on this, how to prevent attackers from exploiting firmware vulnerabilities has become the current main defensive target. At present, the main defense methods are firmware vulnerability detection technology before running and firmware reinforcement technology in running.

For the firmware vulnerability detection technology before running, it is primarily separated into two categories: methods for detecting firmware vulnerabilities based on static analysis and methods for detecting firmware vulnerabilities based on dynamic analysis. Symbolic execution, taint analysis, and other techniques are employed by firmware static analysis to examine the binary code structure and logic without running the firmware program in order to find vulnerabilities. Symbolic execution is a typical method for firmware analysis [73], [74], which uses symbolic values as input. The constraint solver is used to find precise values that can activate the target code after the analyzer has obtained the corresponding path restrictions when the target code has been reached. FirmUSB, which was proposed by Hernandez *et al.* [74], enables symbolic analysis of USB firmware by using Intel 8051 MCUs to detect malicious activities. The taint analysis's goal is to determine if the data that the taint source introduced into the firmware program may be transmitted directly from that source to the taint convergence point without any benign processing. If not, data transfer across the system is secure. Then no, it indicates that the system has security issues including privacy data leaks or risky data processes [75]–[77]. For example, DTaint, a static binary analysis technique Cheng *et al.* introduced, leverages taint analysis to find taint-style flaws in firmware. Firmware dynamic analysis is the real-time analysis of the state of the program when the program is running. The standard procedure at the moment is to detach the firmware program from the hardware devices, run program on simulation software, and further identify vulnerabilities using fuzzing and other techniques [78], [79].

In addition to detecting the firmware of the device before running, there are also firmware hardening methods when the device is running. For example, based on the address space layout randomization (ASLR) and instruction-set randomization (ISR), Cui *et al.* introduced automatic binary structure randomization (ABSR) [80], which inputed an arbitrary executable file or firmware and then produced the original variant. The method reduces unused code and prevents attackers from detecting attack surfaces. Furthermore, the paper proposed a software symbiosis approach, which combines the intrusion detection and binary firmware detection, to suppress malicious attack. However, these two mechanisms are proven feasible theoretically, but they can not be implemented in IoT devices. Li *et al.* applied blockchain technology to upgrade the firmware on the IOT devices [81]; nevertheless, the PoW algorithm this design adopts cannot be satisfied by the IoT devices with limited resources on hardware and time, and it also lacks the ability to detect malicious code. Additionally, some researchers use techniques like EPOXY [82] and MINION [83] to partition the firmware into distinct parts in order to implement least privilege isolation and decrease the attack surface.

2. Security defense technology of edge server

1) Security defense technology of device-based DDoS attack

According to the method of DDoS attack, it can be seen that the underlying cause of attack is the protocol's vulnerabilities in its architecture, which allows illegal users to use legal protocols to attack the edge server. Based on this, it is necessary to choose a corresponding defense method to make up for the vulnerabilities at the protocol level. At present, there are two ways to defend against DDoS attacks. One is packet-level-based traffic detection method and the other is flow-level-based statistical detection method.

The packet-level-based traffic detection methods realize the filtering of these packets by detecting malicious or malformed packets flooded by malicious attackers. The existing research includes the method of adding packet filtering in congestion control and the packet filtering method based on packet identifiers and whitelist. Although these methods are effective in detecting DDoS attacks launched by attackers, they are dwarfed by advanced DDoS attackers. Tools such as hping3 [14] can bypass defense methods that against DDoS attacks based on identifiers by changing the identifiers of packets.

Different from packet-level-based traffic detection methods, flow-level-based statistical detection methods detect DDoS attacks initiated by malicious attackers by using entropy [84], [85] and machine learning [86]–[88]. For the entropy-based traffic detection method, it calculates the entropy of the traffic of the device to obtain the entropy and compares it with the threshold set in advance to determine the DDoS attack initiated by the attacker. For instance, Liu *et al.* [84] suggested an unique DDoS detection technique based on information entropy

and the Fast Fourier transform (FFT). The technique trains a neural network to recognize DDoS attacks using features such as FFT coefficients and working information entropy. However, this method requires more or less manual participation, such as setting the threshold of entropy, and it also requires plenty of traffic to detect accurately. Therefore, traffic detection techniques based on machine learning are utilized in edge computing to achieve accurate recognition of DDoS attacks. For illustration, Chen *et al.* [87] suggested a machine learning-based multi-layer IoT DDoS attack detection system. Algorithms like LSVM, neural networks, and decision trees were employed by Aysa *et al.* [89] to identify anomalous behaviors like DDOS characteristics. By concluding the above research, we found out that even though the traffic detection method based on machine learning rarely requires manual intervention, this method overfits the traffic and some methods still have the probability of false negatives.

2) Security defense technology of malicious injection attack based on edge server

As can be seen from the introduction in Section III, malicious injection attacks based on edge servers are mainly divided into SQL injection attacks, XSS attacks, CSRF/SSRF attacks, and XML signature wrapping attacks. Therefore, this section mainly introduces the defense methods of these attacks.

For SQL injection attacks, early research was divided into two types by Halfond *et al.* One type is the defense mode for detection and the other is the defense mode for prevention [90]. The defense method for detection uses static analysis, dynamic debugging, black-box testing and taint analysis to check the code while the defense method for prevention prevents the execution of any illegal SQL statements by setting proxy filters and randomizing the instruction set. However, most of these researches are immature, some of them can only defend against SQL injection attacks on WEB servers implemented in one language. Therefore, scholars in related filed proposed an improved mechanism later. For example, Bisht *et al.* [91] propose to check for possible inconsistencies by comparing the input query with the expected query, and even simply dropping the attribute values to further analyze before executing the SQL comand. However, these defense methods require a lot of human interaction, which brings some problems regarding the feasibility of deployment. With the development of artificial intelligence technology in recent years, related researchers are looking for approaches which apply machine and deep learning, such as Jackson *et al.* [92] using natural language processing to locate SQL injection vulnerabilities in programs.

Similar to SQL injection, XSS attack defense technology has also undergone years of research and has formed certain results. Gupta *et al.* [93] divided it into 10 types. For example, using Instruction Set Randomization (ISR) to turn malicious code into harmless code and comparing the deviation between an HTTP web request and its associated HTTP response to detect XSS attacks, etc.

There are limited defense ways to target CSRF and SSRF since CSRF and SSRF have a relatively short development time. In order to detect and prevent XSRF assaults while being transparent to both the user and the web apps itself, Jovanovic *et al.* [94] presented a server-side proxy-based method for CSRF. Barth *et al.* [95] described an improved referer header approach, which can protect against CSRF attacks by sending an origin header to WEB server. For SSPF, defense methods against SSRF are more limited. Fung *et al.* [96] put forward a privacy protection mechanism, which can defend against SSRF attacks by inserting client's credentials in the request. Reference [97] modified the static WAF method to make it capable of defending against SSRF attacks.

For XML signature wrapping attack, defender can find protection method based on its restricted attack surface and simple attack method. For instance, reference [98] presented a enhancement based on W3C XML Schema. Leveraging node counts, Gupta *et al.* [99] suggested a technique to identify XML signature wrapping attacks on signed user calls.

3. Summary of security defense technology

This section summarizes the characteristics and shortcomings of edge computing defense methods, which can be mainly divided into the following aspects.

1) The defense method of dictionary attacks still has many problems. Currently, in order to defend dictionary attacks against the authentication process, the main defense schemes include setting complex authentication passwords and setting up a two-layer authentication process. However, these two schemes have certain limitations. For example, the limited resources of edge devices cannot store a large number of user passwords, and the two-layer authentication process still has certain security problems. Therefore, it is necessary to study new defense methods against dictionary attacks.

2) The defense method against side-channel attacks needs to be improved. Since side-channel attacks can use publicly accessible information to infer sensitive data, the current defense research is divided into methods for protecting sensitive data, hiding traffic characteristics, and restricting access to the opposite side channel. The first two methods have certain limitations. For example, the method of protecting sensitive data has been proved possible to be exploited by attackers, and the method of hiding traffic will have a certain impact on the normal traffic of the network. The third method fundamentally solves the possibility of attackers exploiting the side channel, but currently, there are types of research. Therefore,the future research direction should focus on restricting access to the side channel.

3) The firmware's security protection system has to be further enhanced. There are various firmware vulnera-

bilities in the massive and heterogeneous IoT devices in edge computing, which can be exploited by attackers to perform remote control of devices and lateral movement of attacks. At present, the defense against firmware vulnerabilities is mainly divided into pre-running detection and runtime reinforcement methods. Pre-running detection can find firmware vulnerabilities of devices through simulation, but advanced attackers can exploit unknown vulnerabilities to attack. Therefore, the method of runtime reinforcement is a defense method that needs to be focused on. However, the current runtime hardening method will introduce excessive power consumption and delay when deployed in actual devices, so researching new firmware protection mechanisms has become an urgent issue.

4) Artificial intelligence-based DDoS attack detection methods need to be further improved. Existing DDoS attack detection methods for edge servers mainly use machine learning methods to detect traffic, but most of these methods have the problem of overfitting. Based on this, the detection methods for DDoS attacks in the future need to be combined with the latest artificial intelligence methods to carry out in-depth research.

V. Existing Challenges and Future Research Directions

1. Challenges

- Edge computing is designed with a lack of security considerations. When edge computing was first being designed, only data storage, computation, and other tasks were taken into account for sinking to the edge of the network that can supply high bandwidth and low-latency services for devices and users. And they didn't consider the security issues brought by the introduction of the edge service layer. Although there are security defense technologies used on traditional computer equipment that can address the security issues of edge computing systems, these security defense technologies are resource-intensive and cannot be used with edge computing systems that are lightweight. Therefore, edge computing urgently needs suitable security technology for protection.
- The edge computing system has the asymmetry of network attack and defense. The current edge computing security defense technology mainly protects against existing security threats. For example, after dictionary attacks and authentication and authorization protocol vulnerabilities are found, researchers start to study the defense methods of authentication and authorization attack. In addition, defenders only protect against published SQL injection attacks, XSS attacks, CSPF/SSPF attacks and XML signature wrapping attacks. When an advanced attacker finds an unknown vulnerability and takes an attack, the original protection method will become ineffective.
- Firmware vulnerability detection scheme with limited versatility. Current pre-run firmware vulnerability detection methods [72], [74]–[78] and runtime firmware vulnerability detection [79]–[82] methods can only detect specific types of firmware. However, the underlying architecture and underlying hardware on which firmware depends are diverse, making it impossible for firmware vulnerability detection tools to be universal. When testing firmware on a new platform, a new firmware vulnerability detection tool needs to be developed.
- Unpredictable privacy leaks. Most of the current attack methods against edge computing lead to the risk of privacy leakages, such as authentication and authorization attacks, side-channel attacks, and malicious injection attacks based on edge servers, especially the side-channel attacks, where attackers can exploit publicly accessible information to infer sensitive data about the system. Although the current defense methods against side-channel attacks, including hiding sensitive data [62], [63] and hiding communication traffic [25], [66]–[69], can increase the difficulty for attackers to speculate sensitive data to a certain extent, these methods also cause new problems. For example, the method of hiding communication traffic [100] can increase the delay of normal traffic transmission. Besides, the edge computing system design cannot prevent attackers from accessing publicly accessible information, so the problem of privacy leakage cannot be effectively eradicated.
- Security protocols with limited applicability. In edge computing, devices mainly access the edge service layer through security protocols to complete the authentication and authorization process. However, the design of current security protocols lacks two considerations: First, the encryption method used in security protocols does not consider the performance of edge devices. The frequent calculation of encryption algorithms will affect the business of systems. Additionally, the edge computing system uses traditional encryption algorithms and does not consider the limited resources of edge servers. Although there are some researches about lightweight encryption algorithms [58]–[60], these algorithms lack consideration of various attack scenarios; second, there are complex and diverse application scenarios in edge computing, and the currently used security protocols do not consider different application scenarios to select different security protocols.
- Intelligent DDoS attack detection methods lack adaptability. Currently, in the environment of edge computing, flow-level-based statistical detection methods based on artificial intelligence [85]–[87] is the mainstream method that defends against DDoS attack. However, with the development of attack technology, DDoS attacks have the stealth characteristics such as low-rate denial-of-service [101]. For this new DoS attack, the existing machine learning detection methods cannot adapt to new attack scenarios and have the probability of missing alarms.

2. Future research directions

- Defend unknown vulnerabilities in edge comput-

ing by using active defense technologies. The current security defense method of edge computing mainly adopts the passive defense mode of "threat perception, cognitive decision-making, and problem removal". For example, the researchers mainly use passive defense techniques to defend against device-based DDoS attacks and malicious injection attacks based on edge servers: intrusion detection systems [102] and firewalls [103]. However, this method cannot defend against unknown threats and vulnerabilities, so active defensive technologies must be taken into account, such as moving target defense, cyber mimic defense and other technologies to defend against attacks launched by attackers using unknown vulnerabilities.

• AI-based threat detection and defense technologies. At present, Deep learning and machine learning-based techniques have become popular in network security disciplines with the advancement of artificial intelligence technologies. Just as defense technologies against device-based DDoS attacks, the main research direction is to detect the occurrence of DDoS attacks by using machine learning [104] or reinforcement learning [105]. However, these learning methods may overfit network traffic, so relevant researchers need to consider the real network scenarios and select appropriate machine learning algorithms to detect security threats in edge computing systems. In addition, use intelligent technology to form an intelligent vulnerability prediction model. The model can automatically generate vulnerability detection rules, and then efficiently predict potential firmware vulnerabilities.

• More security communication protocol. The network communication protocol in edge computing has the following two characteristics. Firstly, the complex and heterogeneous edge devices make the network communication protocol lack a unified protocol and authorization standard. Secondly, the device with limited computing resources makes the network communication protocol lightweight and lacks security considerations. Therefore, future research on edge computing communication protocols should consider high-security factors in combination with specific scenarios based on considering lightweight.

• Enhanced access control methods. The application scenarios in edge computing are complex, and there are various network attacks against authentication and authorization. The current research for access control methods cannot effectively defend against typical attacks of authentication and authorization. Therefore, it is necessary to study an enhanced access control scheme, which may enhance the security of the edge computing system and make it have high scalability under the premise of limited computing resources of the device.

• Lightweight cryptography techniques with high security. Block ciphers [106] and stream ciphers [107] are two recent cryptographic algorithms that are lightweight in terms of energy, processing capacity, and cost-effectiveness. However, because they do not exhibit resilience to different threats, neither of these is the best option for protecting resource-constrained communications in edge computing systems. Thus, key size reduction, the use of a more frequent dynamic key, and block size reduction can all be the subjects of future research.

• Firmware protection technology is based on trusted architecture. The current research about firmware vulnerability detection technologies is mainly divided into pre-running detection and runtime detection. However, both methods exist flaws. The pre-running detection technology requires a lot of human resources to analyze firmware vulnerabilities while runtime detection technology causes excessive power consumption and latency. Besides, both methods are traditional passive defense techniques. Therefore, it is necessary to proactively protect against firmware vulnerability threats by building a trusted defense architecture. For example, deploying trusted firmware on edge devices and authenticating the trustworthiness of remote device execution states on devices with limited resources is a future research direction.

• Apply lightweight blockchain technology to edge computing. Because the blockchain has the advantages of anonymity and decentralization, applying blockchain technology to edge computing has natural advantages. The advantage of decentralization avoids single points of failure for storage devices in edge computing. The advanced encryption algorithm of blockchain can ensure the security of data transmission of edge devices and prevent attackers from conducting side-channel attacks by sniffing packets. However, edge computing devices have limited computing resources and the blockchain has security problems. Therefore, studying lightweight blockchain technology is an important research direction in the future.

VI. Conclusion

The three-layer system architecture of edge computing makes the edge computing system have three attack surfaces. The first attack surface is between the Cloud computing service layer and the Edge service layer. The second attack surface is on the Edge server and the third attack surface is between the Edge service layer and the Device layer. Based on the three attack surfaces of edge computing, this paper focuses on the security threats and defense methods on edge servers and between the Edge service layer and the Device layer. On the basis of this, we suggest the issues with edge computing security protection and the future course of study.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (Grant No. 62072467) and the National Key R&D Program of China (Grant Nos. 2021YFB1006200 and 2021YFB1006201).

References

[1] J. R. David Reinsel and J. Gantz, "The digitization of the world-from edge to core," Available at: https://www.sea-

gate.com/files/www-content/ourstory/trends/files/idc-seagate-dataage-whitepaper.pdf, 2018.

[2] J. L. Zhang, Y. C. Zhao, B. Chen, *et al.*, "Survey on data security and privacy-preserving for the research of edge computing," *Journal on Communications*, vol. 39, no. 3, pp. 1–21, 2018. (in Chinese)

[3] X. W. Li, B. H. Chen, D. Q. Yang, *et al.*, "Review of security protocols in edge computing environments," *Journal of Computer Research and Development*, vol. 59, no. 4, pp. 765–780, 2022. (in Chinese)

[4] Y. H. Xiao, Y. Z. Jia, C. C. Liu, *et al.*, "Edge computing security: State of the art and challenges," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1608–1631, 2019.

[5] Z. Y. Huang, G. M. Xia, Z. H. Wang, *et al.*, "Survey on edge computing security, " in *Proceedings of 2020 International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering*, Fuzhou, China, pp. 96–105, 2020.

[6] I. N. Releases, "Ibm and nokia siemens networks announce worlds first mobile edge computing platform," Available at: http://www-03.ibm.com/press/us/en/pressrelease/40490.wss.15, 2016.

[7] ETSI, "Mobile-edge computing-introductory technical white paper," Available at: https://max.book118.com/html/2018/1006/8013022103001125.shtm, 2014.

[8] ETSI, "Mobile edge computing: A key technology towards 5g," Available at: https://docslib.org/doc/612752/mobile-edge-computinga-key-technology-towards-5g, 2015.

[9] E. C. I. Alliance, "Edge computing industry alliance white paper," Available at: https://www.digitalelite.cn/h-nd-921.html, 2016.

[10] Alibaba Cloud Computing Ltd and China Electronics Standardization Institute, "Edge cloud computing technology and standardization white paper," Available at: http://www.cesi.cn/images/editor/20181214/20181214115429307.pdf, 2018-12-12. (in Chinese)

[11] ISO, "Information technology-cloud computing-edge computing landscape," Available at: https://webstore.iec.ch/preview/info_isoiectr23188%7Bed1.0%7Den.pdf, 2020.

[12] ECC and AII, "Edge computing reference architecture 3.0," Available at: http://www.ecconsortium.org/Uploads/file/20181214/20181214104331_73917.pdf, 2018. (in Chinese)

[13] A. Greenberg, "100mb password dictionary," Available at: https://github.com/danielmiessler/SecLists/tree/master/Passwords, 2018.

[14] A. Cassola, W. K. Robertson, E. Kirda, *et al.*, "A practical, targeted, and stealthy attack against WPA enterprise authentication," in *Proceedings of the 20th Annual Network and Distributed System Security Symposium*, San Diego, CA, USA, pp. 1–15, 2013.

[15] K. Bhargavan and G. Leurent, "Transcript collision attacks: Breaking authentication in TLS, IKE, and SSH," in *Proceedings of the Network and Distributed System Security Symposium*, San Diego, CA, USA, pp. 1–17, 2016.

[16] T. T. Zhu, L. Fu, Q. Liu, *et al.*, "One cycle attack: Fool sensor-based personal gait authentication with clustering," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 553–568, 2020.

[17] E. Hammer-Lahav, "The OAuth 1.0 protocol," RFC5849, 2010.

[18] D. Hardt, "The OAuth 2.0 authorization framework," RFC 6749, 2012.

[19] A. Greenberg, "Oauth security advisory," Available at: http://oauth.net/advisories/2009–1/, 2009.

[20] S. T. Sun and K. Beznosov, "The devil is in the (implementation) details: An empirical analysis of OAuth SSO systems," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, Raleigh North, CA, USA, pp. 378–390, 2012.

[21] H. Li, Y. H. He, L. M. Sun, *et al.*, "Side-channel information leakage of encrypted video stream in video surveillance systems," in *Proceedings of the 35th Annual IEEE International Conference on Computer Communications*, San Francisco, CA, USA, pp. 1–9, 2016.

[22] N. Apthorpe, D. Reisman, and N. Feamster, "A smart home is no castle: Privacy vulnerabilities of encrypted IoT traffic," *arXiv preprint*, arXiv: 1705.06805, 2017.

[23] W. T. Chen and Z. Y. Qian, "Off-Path TCP exploit: How wireless routers can jeopardize your secrets," in *Proceedings of the 27th USENIX Conference on Security Symposium*, Baltimore, MD, USA, pp. 1581–1598, 2018.

[24] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, *et al.*, "Network traffic classifier with convolutional and recurrent neural networks for internet of things," *IEEE Access*, vol. 5, pp. 18042–18050, 2017.

[25] A. Acar, H. Fereidooni, T. Abera, *et al.*, "Peek-a-boo: I see your smart home activities, even encrypted!," in Proceed-ings of the 13th ACM Conference on Security and Pri-vacy in Wireless and Mobile Networks, Linz, Austria, pp. 207–218, 2020.

[26] M. Enev, S. Gupta, T. Kohno, *et al.*, "Televisions, video privacy, and powerline electromagnetic interference," in *Proceedings of the 18th ACM Conference on Computer and Communications Security*, Chicago, IL, USA, pp. 537–550, 2011.

[27] Y. Meng, J. L. Li, H. J. Zhu, *et al.*, "Revealing your mobile password via WiFi signals: Attacks and countermeasures," *IEEE Transactions on Mobile Computing*, vol. 19, no. 2, pp. 432–449, 2020.

[28] G. W. Hart, "Nonintrusive appliance load monitoring," *Proceedings of the IEEE*, vol. 80, no. 12, pp. 1870–1891, 1992.

[29] L. Stankovic, V. Stankovic, J. Liao, *et al.*, "Measuring the energy intensity of domestic activities from smart meter data," *Applied Energy*, vol. 183, pp. 1565–1580, 2016.

[30] E. Ronen, A. Shamir, A. O. Weingarten, *et al.*, "IoT goes nuclear: Creating a zigbee chain reaction," in *Proceedings of 2017 IEEE Symposium on Security and Privacy*, San Jose, CA, USA, pp. 195–212, 2017.

[31] A. Greenberg, "The reaper IoT botnet has already infected a million networks," Available at: https://www.wired.com/story/reaper-iotbotnet-infectedmillion-networks/, 2018-01-13.

[32] J. Maskiewicz, B. Ellis, J. Mouradian, *et al.*, "Mouse trap: Exploiting firmware updates in USB peripherals, " in *Proceedings of the 8th USENIX Conference on Offensive Technologies*, San Diego, CA, USA, pp. 1–10, 2014.

[33] R. Wang, L. Y. Xing, X. F. Wang, *et al.*, "Unauthorized origin crossing on mobile platforms: Threats and mitigation," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, Berlin, Germany, pp. 635–646, 2013.

[34] C. G. Ren, Y. L. Zhang, H. Xue, *et al.*, "Towards discovering and understanding task hijacking in android," in *Proceedings of the 24th USENIX Conference on Security Symposium*, Washington, DC, USA, pp. 945–959, 2015.

[35] Y. H. Xiao, G. D. Bai, J. Mao, *et al.*, "Privilege leakage and information stealing through the android task mechanism," in *Proceedings of 2017 IEEE Symposium on Privacy-Aware Computing*, Washington, DC, USA, pp. 152–163, 2017.

[36] K. Chen, X. Q. Wang, Y. Chen, *et al.*, "Following devil's footprints: Cross-platform analysis of potentially harmful libraries on android and iOS," in *Proceedings of 2016 IEEE Symposium on Security and Privacy*, San Jose, CA, USA, pp. 357–376, 2016.

[37] M. Antonakakis, T. April, M. Bailey, *et al.*, "Understanding the mirai botnet," in *Proceedings of the 26th USENIX Conference on Security Symposium*, Vancouver, Canada, pp.

1093–1110, 2017.

[38] C. Kolias, G. Kambourakis, A. Stavrou, *et al.*, "DDoS in the IoT: Mirai and other botnets," *Computer*, vol. 50, no. 7, pp. 80–84, 2017.

[39] S. Rizvi, A. Kurtz, J. Pfeffer, *et al.*, "Securing the internet of things (IoT): A security taxonomy for IoT," in *Proceedings of the 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/ 12th IEEE International Conference on Big Data Science and Engineering*, New York, NY, USA, pp. 163–168, 2018.

[40] Cisco, "Cisco fog director cross-site scripting vulnerability," Available at: https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20160201-fd, 2016-02-01.

[41] A. Costin, "IoT/embedded vs. security: Learn from the past, apply to the present, prepare for the future," in *Proceedings of the 22nd Conference of Open Innovations Association*, Jyvaskyla, Finland, pp. 1-13, 2018.

[42] M. S. Ansari, S. H. Alsamhi, Y. S. Qiao, *et al.*, "Security of distributed intelligence in edge computing: Threats and countermeasures," in *The Cloud-to-Thing Continuum: Opportunities and Challenges in Cloud, Fog and Edge Computing*, Eds. Springer, Cham, Germany, pp. 95–122, 2020.

[43] B. Pinkas and T. Sander, "Securing passwords against dictionary attacks," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, Washington, DC, USA, pp. 161–170, 2002.

[44] A. T. B. Jin, D. N. C. Ling, and A. Goh, "Biohashing: Two factor authentication featuring fingerprint data and tokenised random number," *Pattern Recognition*, vol. 37, no. 11, pp. 2245–2255, 2004.

[45] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proceedings of 2015 IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, USA, pp. 815–823, 2015.

[46] F. Aloul, S. Zahidi, and W. El-Hajj, "Two factor authentication using mobile phones," in *Proceedings of 2009 IEEE/ACS International Conference on Computer Systems and Applications*, Rabat, Morocco, pp. 641–644, 2009.

[47] M. Dailey and C. Namprempre, "A text graphics character CAPTCHA for password authentication," in *Proceedings of 2004 IEEE Region 10 Conference TENCON 2004*, Chiang Mai, Thailand, pp. 45–48, 2004.

[48] N. Karapanos, C. Marforio, C. Soriente, *et al.*, "SoundProof: Usable two-factor authentication based on ambient sound," in *Proceedings of the 24th USENIX Security Symposium*, Washington, DC, USA, pp. 483–498, 2015.

[49] C. Mulliner, R. Borgaonkar, P. Stewin, *et al.*, "SMS-based one-time passwords: Attacks and defense," in *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, Berlin, Germany, pp. 150–159, 2013.

[50] D. Wang, J. Ming, T. Chen, *et al.*, "Cracking IoT device user account via brute-force attack to SMS authentication code," in *Proceedings of the First Workshop on Radical and Experiential Security*, Incheon, Republic of Korea, pp. 57–60, 2018.

[51] M. Joshi, B. Mazumdar, and S. Dey, "Security vulnerabilities against fingerprint biometric system," *arXiv preprint*, arXiv: 1805.07116, 2018.

[52] Y. L. Liu, "Defense of WPA/WPA2-psk brute forcer," in *Proceedings of 2015 2nd International Conference on Information Science and Control Engineering*, Shanghai, China, pp. 185–188, 2015.

[53] J. Noh, J. Kim, G. Kwon, *et al.*, "Secure key exchange scheme for WPA/WPA2-psk using public key cryptography," in *Proceedings of 2016 IEEE International Conference on Consumer Electronics-Asia*, Seoul, Korea (South), pp. 1–4, 2016.

[54] S. Sivakorn, G. Argyros, K. X. Pei, *et al.*, "HVLearn: Automated black-box analysis of hostname verification in SSL/TLS implementations," in *Proceedings of 2017 IEEE Symposium on Security and Privacy*, San Jose, CA, USA, pp. 521–538, 2017.

[55] K. Bhargavan, B. Blanchet, and N. Kobeissi, "Verified models and reference implementations for the TLS 1.3 standard candidate," in *Proceedings of 2017 IEEE Symposium on Security and Privacy*, San Jose, CA, USA, pp. 483–502, 2017.

[56] M. Shehab and F. Mohsen, "Securing OAuth implementations in smart phones," in *Proceedings of the 4th ACM Conference on Data and Application Security and Privacy*, San Antonio, TX, USA, pp. 167–170, 2014.

[57] S. Cirani, M. Picone, P. Gonizzi, *et al.*, "IoT-OAS: An OAuth-based authorization service architecture for secure services in IoT scenarios," *IEEE Sensors Journal*, vol. 15, no. 2, pp. 1224–1234, 2015.

[58] V. Prakash, A. V. Singh, and S. K. Khatri, "A new model of light weight hybrid cryptography for internet of things," in *Proceedings of 2019 3rd International conference on Electronics, Communication and Aerospace Technology*, Coimbatore, India, pp. 282–285, 2019.

[59] R. Chatterjee and R. Chakraborty, "A modified lightweight PRESENT cipher for IoT security," in *Proceedings of 2020 International Conference on Computer Science, Engineering and Applications*, Gunupur, India, pp. 1–6, 2020.

[60] H. Noura, A. Chehab, L. Sleem, *et al.*, "One round cipher algorithm for multimedia IoT devices," *Multimedia Tools and Applications*, vol. 77, no. 14, pp. 18383–18413, 2018.

[61] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 5, pp. 557–570, 2002.

[62] Z. Ling, J. Z. Luo, Y. Zhang, *et al.*, "A novel network delay based side-channel attack: Modeling and defense," in *Proceedings of 2012 Proceedings IEEE INFOCOM*, Orlando, FL, USA, pp. 2390–2398, 2012.

[63] Y. Zhang, Y. L. Mao, M. Z. Xu, *et al.*, "Towards thwarting template side-channel attacks in secure cloud deduplications," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 3, pp. 1008–1018, 2021.

[64] A. Machanavajjhala, J. Gehrke, D. Kifer, *et al.*, "L-diversity: Privacy beyond k-anonymity," in *Proceedings of 22nd International Conference on Data Engineering*, Atlanta, GA, USA, pp. 3–19, 2006.

[65] N. H. Li, T. C. Li, and S. Venkatasubramanian, "t-closeness: Privacy beyond k-anonymity and l-diversity," in *Proceedings of 2007 IEEE 23rd International Conference on Data Engineering*, Istanbul, Turkey, pp. 106–115, 2007.

[66] A. Sivanathan, H. H. Gharakheili, F. Loi, *et al.*, "Classifying IoT devices in smart environments using network traffic characteristics," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1745–1759, 2019.

[67] D. Wood, N. Apthorpe, and N. Feamster, "Cleartext data transmissions in consumer IoT medical devices," in *Proceedings of the 2017 Workshop on Internet of Things Security and Privacy*, Dallas, TX, USA, pp. 7–12, 2017.

[68] W. Zhang, Y. Meng, Y. G. Liu, *et al.*, "HoMonit: Monitoring smart home apps from encrypted traffic," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, Toronto, Canada, pp. 1074–1088, 2018.

[69] Q. Chen, Y. Zhuang, J. Liang, *et al.* "Research on dns encryption technology", in International Conference on Computer Engineering and Networks, Singapore, pp. 1395–1405, 2022.

[70] D. Molnar, M. Piotrowski, D. Schultz, *et al.*, "The program counter security model: Automatic detection and removal of control-flow side channel attacks," in *Proceedings of the 8th International Conference on Information Security and Cryptology*, Seoul, Korea, pp. 156–168, 2005.

[71] R. Strackx and F. Piessens, "The Heisenberg defense: Proactively defending SGX enclaves against page-table-based side-channel attacks," *arXiv preprint*, arXiv: 1712.08519, 2017.

[72] Y. C. Fu, E. Bauman, R. Quinonez, *et al.*, "SGX-LAPD: Thwarting controlled side channel attacks via enclave verifiable page faults," in *International Symposium on Research in Attacks, Intrusions, and Defenses*, Atlanta, GA, USA, pp. 357–380, 2017.

[73] P. Subramanyan, S. Malik, H. Khattri, *et al.*, "Verifying information flow properties of firmware using symbolic execution," in *Proceedings of 2016 Design, Automation & Test in Europe Conference & Exhibition*, Dresden, Germany, pp. 337–342, 2016.

[74] G. Hernandez, F. Fowze, D. Tian, *et al.*, "FirmUSB: Vetting USB device firmware using domain informed symbolic execution," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, Dallas, TX, USA, pp. 2245–2262, 2017.

[75] N. Redini, A. Machiry, R. Y. Wang, *et al.*, "Karonte: Detecting insecure multi-binary interactions in embedded firmware," in *Proceedings of 2020 IEEE Symposium on Security and Privacy*, San Francisco, CA, USA, pp. 1544–1561, 2020.

[76] Y. Yao, W. Zhou, Y. Jia, *et al.*, "Identifying privilege separation vulnerabilities in IoT firmware with symbolic execution," in *Proceedings of the 24th European Symposium on Research in Computer Security*, Luxembourg, Luxembourg, pp. 638–657, 2019.

[77] K. Cheng, Q. Li, L. Wang, *et al.*, "DTaint: Detecting the taint-style vulnerability in embedded device firmware," in *Proceedings of 2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, Luxembourg, Luxembourg, pp. 430–441, 2018.

[78] D. D. Chen, M. Egele, M. Woo, *et al.*, "Towards automated dynamic analysis for linux-based Towards Automated Dynamic Analysis for Linux-based," in *Proceedings of* the NDSS, San Diego, CA, USA, pp. 21–24, 2016.

[79] Y. W. Zheng, A. Davanian, H. Yin, *et al.*, "FIRM-AFL: High-throughput greybox fuzzing of IoT firmware via augmented process emulation," in *Proceedings of the 28th USENIX Conference on Security Symposium*, Santa Clara, CA, USA, pp. 1099–1114, 2019.

[80] A. Cui, M. Costello, and S. Stolfo, "When firmware modifications attack: A case study of embedded exploitation," in *Proceedings of the 20th Annual Network & Distributed System Security Symposium*, San Diego, California, pp. 1078–1088, 2013.

[81] B. Lee and J. H. Lee, "Blockchain-based secure firmware update for embedded devices in an internet of things environment," *The Journal of Supercomputing*, vol. 73, no. 3, pp. 1152–1167, 2017.

[82] A. A. Clements, N. S. Almakhdhub, K. S. Saab, *et al.*, "Protecting bare-metal embedded systems with privilege overlays," in *Proceedings of 2017 IEEE Symposium on Security and Privacy*, San Jose, CA, USA, pp. 289–303, 2017.

[83] C. H. Kim, T. Kim, H. Choi, *et al.*, "Securing real-time microcontroller systems through customized memory view switching," in *Proceedings of the Network and Distributed System Security Symposium*, San Diego, CA, USA, pp. 1–15, 2018.

[84] Z. Liu, C. Z. Hu, and C. Shan, "Riemannian manifold on stream data: Fourier transform and entropy-based DDoS attacks detection method," *Computers & Security*, vol. 109, article no. 102392, 2021.

[85] J. B. Li, M. Liu, Z. Xue, *et al.*, "RTVD: A real-time volumetric detection scheme for DDoS in the internet of things," *IEEE Access*, vol. 8, pp. 36191–36201, 2020.

[86] R. Doshi, N. Apthorpe, and N. Feamster, "Machine learning DDoS detection for consumer internet of things devices," in *Proceedings of 2018 IEEE Security and Privacy Workshops*, San Francisco, CA, USA, pp. 29–35, 2018.

[87] Y. W. Chen, J. P. Sheu, Y. C. Kuo, *et al.*, "Design and implementation of IoT DDoS attacks detection system based on machine learning," in *Proceedings of 2020 European Conference on Networks and Communications*, Dubrovnik, Croatia, pp. 122–127, 2020.

[88] R. Vishwakarma and A. K. Jain, "A honeypot with machine learning based detection framework for defending IoT based botnet DDoS attacks," in *Proceedings of 2019 3rd International Conference on Trends in Electronics and Informatics*, Tirunelveli, India, pp. 1019–1024, 2019.

[89] M. H. Aysa, A. A. Ibrahim, and A. H. Mohammed, "IoT Ddos attack detection using machine learning," in *Proceedings of 2020 4th International Symposium on Multidisciplinary Studies and Innovative Technologies*, Istanbul, Turkey, pp. 1–7, 2020.

[90] W. G. J. Halfond, J. Viegas, and A. Orso, "A classification of SQL-injection attacks and countermeasures," in *Proceedings of the International Symposium on Secure Software Engineering*, Washington, DC, USA, pp. 13–15, 2006.

[91] P. Bisht, P. Madhusudan, and V. N. Venkatakrishnan, "CANDID: Dynamic candidate evaluations for automatic prevention of SQL injection attacks," *ACM Transactions on Information and System Security*, vol. 13, no. 2, article no. 14, 2010.

[92] K. A. Jackson and B. T. Bennett, "Locating SQL injection vulnerabilities in java byte code using natural language techniques," in *Proceedings of the SoutheastCon 2018*, St. Petersburg, FL, USA, pp. 1–5, 2018.

[93] S. Gupta and B. B. Gupta, "Cross-site scripting (XSS) attacks and defense mechanisms: Classification and state-of-the-art," *International Journal of System Assurance Engineering and Management*, vol. 8, no. S1, pp. S512–S530, 2017.

[94] N. Jovanovic, E. Kirda, and C. Kruegel, "Preventing cross site request forgery attacks," in *Proceedings of 2006 Securecomm and Workshops*, Baltimore, MD, USA, pp. 1–10, 2006.

[95] A. Barth, C. Jackson, and J. C. Mitchell, "Robust defenses for cross-site request forgery," in *Proceedings of the 15th ACM Conference on Computer and Communications Security*, Alexandria, VA, USA, pp. 75–88, 2008.

[96] B. S. Y. Fung and P. P. C. Lee, "A privacy-preserving defense mechanism against request forgery attacks," in *Proceedings of 2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications*, Changsha, China, pp. 45–52, 2011.

[97] M. Srokosz, D. Rusinek, and B. Ksiezopolski, "A new WAF-based architecture for protecting web applications against CSRF attacks in malicious environment," in *Proceedings of 2018 Federated Conference on Computer Science and Information Systems*, Poznan, Poland, pp. 391–395, 2018.

[98] M. Jensen, C. Meyer, J. Somorovsky, *et al.*, "On the effectiveness of XML schema validation for countering XML signature wrapping attacks," in *Proceedings of 2011 1st International Workshop on Securing Services on the Cloud*, Milan, Italy, pp. 7–13, 2011.

[99] A. N. Gupta and P. S. Thilagam, "Detection of XML signature wrapping attack using node counting," in *Proceedings of the 3rd International Symposium on Big Data and Cloud*

Computing Challenges (ISBCC–16'), V. Vijayakumar and V. Neelanarayanan, Eds. Springer, Cham, Germany, pp. 57–63, 2016.

[100] N. Apthorpe, D. Y. Huang, D. Reisman, *et al.*, "Keeping the smart home private with smart (er) IoT traffic shaping," *arXiv preprint*, arXiv: 1812.00955, 2018.

[101] V. De Miranda Rios, P. R. M. Inácio, D. Magoni, *et al.*, "Detection and mitigation of low-rate denial-of-service attacks: A survey," *IEEE Access*, vol. 10, pp. 76648–76668, 2022.

[102] S. Vimal, A. Suresh, P. Subbulakshmi, *et al.*, "Edge computing-based intrusion detection system for smart cities development using IoT in urban areas," in *Internet of things in smart Technologies for Sustainable Urban Development*, G. R. Kanagachidambaresan, R. Maheswar, V. Manikandan, *et al.*, Eds. Springer, Cham, Germany, pp. 219–237, 2020.

[103] F. Kamoun-Abid, M. Rekik, A. Meddeb-Makhlouf, *et al.*, "Secure architecture for Cloud/Fog computing based on firewalls and controllers," *Procedia Computer Science*, vol. 192, pp. 822–833, 2021.

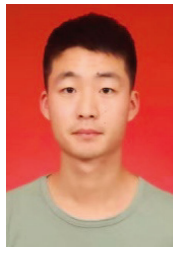
[104] S. Singh, R. Sulthana, T. Shewale, *et al.*, "Machine-learning-assisted security and privacy provisioning for edge computing: A survey," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 236–260, 2022.

[105] H. D. Zhang, J. Y. Hao, and X. H. Li, "A method for deploying distributed denial of service attack defense strategies on edge servers using reinforcement learning," *IEEE Access*, vol. 8, pp. 78482–78491, 2020.

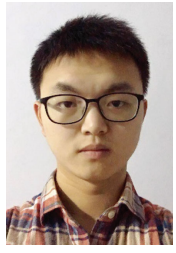
[106] N. Thangamani and M. Murugappan, "A lightweight cryptography technique with random pattern generation," *Wireless Personal Communications*, vol. 104, no. 4, pp. 1409–1432, 2019.

[107] M. J. R. Shantha and L. Arockiam, "SAT_Jo: An enhanced lightweight block cipher for the internet of things," in *Proceedings of 2018 Second International Conference on Intelligent Computing and Control Systems*, Madurai, India, pp. 1146–1150, 2018.

Ke SHANG was born in 1995. She received the M.E. degree from University of Sydney, Australia, in 2018. She is currently an Assistant Professor of the Information Engineering University, Zhengzhou, China. Her research interests focus on edge computing security. (Email: keshan1995@163.com)



Weizhen HE was born in 1996. He received the M.E. degree from the Information Engineering University, Zhengzhou, China, in 2020. He is currently a Ph.D. candidate of the Information Engineering University. His research interests include cloud security and deception. (Email: heweizhen@alu.hit.edu.cn)



Shuai ZHANG was born in 1994. He received the Ph.D. degree from the Information Engineering University, Zhengzhou, China, in 2021. He is currently an Assistant Professor of the Information Engineering University. His research interests focus on cloud native security. (Email: 2012301200229@whu.edu.cn)