A Security Framework to Protect Edge Supported Software Defined Internet of Things Infrastructure

Wajid Rafique[1,2], Maqbool Khan[1,2], Nadeem Sarwar[3], and Wanchun Dou[1,2(✉)]

[1] State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, People's Republic of China
rafiqwajid@smail.nju.edu.cn, douwc@nju.edu.cn
[2] The Department of Computer Science and Technology, Nanjing University, Nanjing, People's Republic of China
[3] Department of Computer Science, Bahria University, Lahore, Pakistan

Abstract. Managing the huge IoT infrastructure poses a vital challenge to the network community. Software Defined Networking (SDN), due to its characteristics of centralized network management has been considered as an optimal choice to manage IoT. Edge computing brings cloud recourses near the IoT to localize the cloud demands. Consequently, SDN, IoT, and edge computing can be combined into a framework to create a resourceful SDIoT-Edge architecture to efficiently orchestrate cloud services and utilize resource-limited IoT devices in a flexible way. Besides a wide adoption of IoT, the vulnerabilities present in this less secure infrastructure can be exploited by the adversaries to attack the OpenFlow channel using Distributed Denial of Service (DDoS) attacks. DDoS on OpenFlow channel have the ability to disrupt the whole network hence, providing security for the OpenFlow channel is a key challenge in SDIoT-Edge. We propose a security framework called SDIoT-Edge Security (SIESec) against the security vulnerabilities present in this architecture. SIESec prototype employs machine learning-based classification strategy, blacklist integration, and contextual network flow filtering to efficiently defend against the DDoS attacks. We perform extensive simulations using Floodlight controller and Mininet network emulator. Our results proclaim that SIESec provides extensive security against OpenFlow channel DDoS attacks and pose a very less overhead on the network.

Keywords: SDN · IoT · Edge computing · Security · DDoS

1 Introduction

Information technology (IT) has revolutionized the lifestyle of human-beings where ubiquitous computing has been widely adopted affirming Mark Weiser's prediction of extraordinary IT involvement in everyday life, which he proposed

X. Wang et al. (Eds.): CollaborateCom 2019, LNICST 292, pp. 71–88, 2019.
https://doi.org/10.1007/978-3-030-30146-0_6

28 years ago [28]. Cisco Systems claims that more than 50 billion devices will be connected to the internet until 2020 [4]. Internet of Things (IoT) has been deployed in all the fields of life, including industry, agriculture, health, transport, homes, and many others. The revenue for IoT vendors, service providers, and software solution developers is expected to reach $1 trillion until 2025 [19]. Besides, such lucrative benefits, managing such a huge repository of connected objects is a vital challenge. The resource-limitation in IoT devices makes it challenging to deploy a security solution onto the IoT infrastructure. Software Defined Networking (SDN) offers a layered architecture to enable flexible control, management, and programmability of the network. Therefore, the research community believes that SDN is an optimal choice to manage decentralized IoT infrastructure [13,15].

Since IoT devices are limited in resources, therefore cloud services facilitate compute-intensive tasks on IoT. Edge cloudlets are placed between the traditional cloud and the IoT infrastructure to offload the computations. Moreover, edge cloudlets also act as data filtering and classification resource, which only transmit the mandatory data to the traditional cloud data center and redeem vital network resources, including bandwidth, energy, and storage. Similarly, edge computing can effectively help resource-limited and latency-sensitive IoT applications by providing computation infrastructure near the edge of IoT. As SDN and edge are more powerful resources as compared to IoT, they are combined to devise a sustainable infrastructure of Software Defined Internet of Things using Edge computing (SDIoT-Edge) for efficient IoT service orchestration [16].

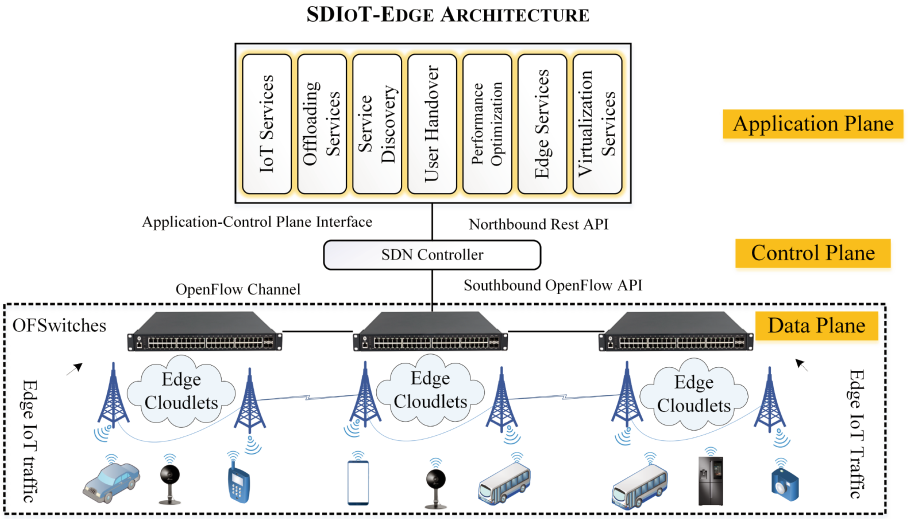

Fig. 1. An architecture of SDIoT-Edge.

Fig. 1 illustrates the integration of SDN, edge computing, and IoT to devise an SDIoT-Edge architecture where IoT devices are connected with the edge cloudlets at the data plane. The application plane contains novel edge services, including service discovery, user handover, offloading, and virtualization to facilitate edge resource provisioning. The figure also represents the OpenFlow channel, which connects the control plane and the data plane of the SDIoT-Edge framework. Although the integration of IoT-Edge infrastructure in the SDN paradigm seems a promising solution, this architecture is vulnerable toward countless novel security challenges and attacks including Link Flooding Attack (LFA) [18] and Distributed Denial of Service Attacks (DDoS) [11]. For example, in a recent Mirai botnet attack, the adversary leveraged the security vulnerabilities in IoT to prepare a huge army of compromised devices to attack internet infrastructure. The attackers used the open Teletype Network (Telnet) ports of IoT devices and tried to login using 61 different combinations of user-name and passwords that were mostly used as default credentials and never changed. After acquiring access to these devices, the attackers were able to manipulate 500,000 IoT botnets to attack internet infrastructure [11]. In another similar incident, vulnerabilities present in IoT architecture were exploited by adversaries to attack Dyn's [2] Domain Name Server (DNS)[1] causing massive information and revenue loss [8].

OpenFlow channel is a vital resource in SDIoT-Edge architecture as all the control information, e.g., flow rule installation, traffic management, and policy enforcement need to pass through this channel [12]. Consequently, the security of the OpenFlow channel is of prime importance in SDIoT-Edge ecosystem. Any attack on the OpenFlow channel can provoke management inconsistencies in the network and in severe circumstances, bring down the whole network. For example, in a DDoS attack, a malicious adversary can exploit the resource-limitation vulnerabilities in IoT to employ them as bots to attack the OpenFlow channel. In such an incident, 100,000 IoT devices were compromised, which attacked individual systems and enterprise servers around the globe, which provoked a huge revenue loss [22]. Therefore, providing security in SDIoT-Edge infrastructure is of prime importance to safeguard current networks. IoT devices are limited in memory, which makes it challenging to provide security solutions on these devices, therefore, network-based security solutions are highly needed.

Due to these vulnerabilities, there is a high need to provide security solutions for the OpenFlow channel protection in SDIoT-Edge. Therefore, we propose a network-level security solution against OpenFlow channel DDOS attacks named as SDIoT-Edge Security (SIESec). We develop this as a solution at the application plane of the SDN controller. We simulate the DDoS attack from IoT devices to demonstrate their vulnerabilities to be manipulated and provide a defense. SIESec employs an unsupervised machine learning classifier based on Self Organizing Maps (SOM) and includes blacklisting of malicious hosts, contextual traffic filtering, and customized flow rule generation for the identified malicious flows. SIESec performs SDN-oriented flow measurements therefore, no

[1] This attack targeted DNS systems of Dyn which caused major network services outage in Europe and North America.

extra hardware or measurement agents are required at the data plane of SDN. We perform extensive simulations using Mininet network emulator and Floodlight open-source controller to demonstrate the effectiveness of detection and mitigation of the SIESec. We present the contributions of this research in the following.

- We propose an architecture of SDIoT-Edge and highlight DDoS vulnerabilities on the OpenFlow channel of this architecture where any attack on this channel disrupts the whole network infrastructure.
- We devise a SIESec solutions, which utilize an unsupervised SOM-based classification and malicious traffic filtering based on blacklists and contextual information to detect and eliminate DDoS attacks in SDIoT-Edge.
- A comprehensive experimental evaluation and comparison demonstrate that SIESec provides efficient security against the OpenFlow channel DDoS attacks and induces a negligible overhead on the network.

Rest of the paper is organized as follows. Section 2 elaborates the architecture of the SDIoT-Edge and its security vulnerabilities. Section 3 discusses the SIESec solution, its components, and the working principle of all these components. Section 4 illustrates the experimental evaluation of the solution using different performance parameters. Section 5 presents the related work and comparison analysis, and finally, Sect. 6 concludes the paper and provides some future insights.

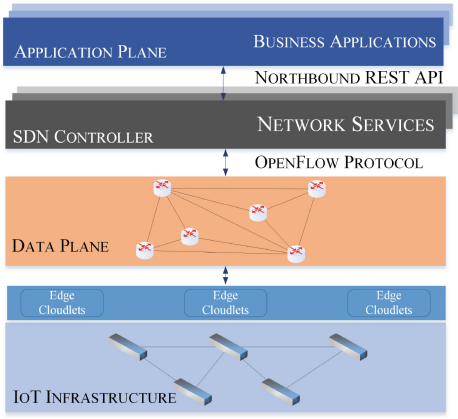

Fig. 2. A high-level architecture of SDIoT-Edge which extends the traditional SDN architecture [1].

2 SDIoT-Edge Architecture

The SDIoT-Edge architecture encompasses SDN, IoT, and edge computing to provide seamless infrastructure management and service orchestration. In this

architecture, the cloudlets provide the offloading capability to the resource-limited IoT devices. These cloudlets are placed between the IoT infrastructure and the central cloud data center to facilitate the IoT devices in performing compute-intensive tasks. A high-level architecture of SDIoT-Edge is provided in Fig. 2, which illustrates three SDN planes and an IoT infrastructure plane facilitated by edge cloudlets. The data plane includes OpenFlow-enabled switches which forward the IoT traffic by exploiting three flow rule installation strategies including reactive, proactive, and hybrid. The application plane of the controller enables programmability where the administrators can develop and deploy innovative applications to provoke a wide range of services, including customized traffic forwarding, security, and management. In this architecture, the application plane includes edge services to effectively manage the IoT-service orchestration by using cloud infrastructure at the edge.

Due to the presence of immense security vulnerabilities in the resource-limited IoT devices, they can be effortlessly manipulated by the adversaries, which can deploy them as bots and attack the sophisticated network infrastructure. A few examples of such attacks generated by the IoT in SDIoT-Edge infrastructure includes information spoofing using Man-in-the-Middle (MiTM) attacks, policy switch attacks, flow table overflow attacks, and OpenFlow channel attacks. A taxonomy of these attacks is presented in Fig. 3. The most lethal of these attacks is the OpenFlow channel flooding attack where the controller can be disconnected from the infrastructure plane by DDoS traffic. The adversaries exploit IoT vulnerabilities and devise a manipulated army of bots to generate new flow rule installation requests at the data plane switches which continuously communicate with the controller for the flow rules. A higher number of flow rule installation requests congest the OpenFlow channel and disconnect it from the data plane in severe cases. This attack has lethal consequences on the network where it can shut down the whole network in severe cases. We present a defense solution to mitigate DDoS attacks on the SDIoT-Edge infrastructure. The characteristics of the novel SDIoT-Edge architecture are illustrated in the following.

1. **Resource Limitation:** IoT encompasses resource-limited infrastructure having the lower processing speed, memory, energy, and storage capacities. Therefore, these devices cannot support complex algorithms and defense strategies such as endpoint encryption and security solutions against the attacks. Meanwhile, this resource-limitation can be exploited by the adversary to manipulate them as bots in many devastating attacks.
2. **IoT Big Data:** A large number of IoT devices produce a huge amount of data which renders the basic requirement for DDoS attacks exertion and propagation. Although the same amount of data can be generated by other powerful infrastructures, the data generating resources in IoT are countless, which become a potential enabler for a security threat toward the network infrastructure.
3. **Flow Rule Installation:** In SDIoT-Edge the controller uses OpenFlow channel to communicate with the infrastructure, which is the backbone

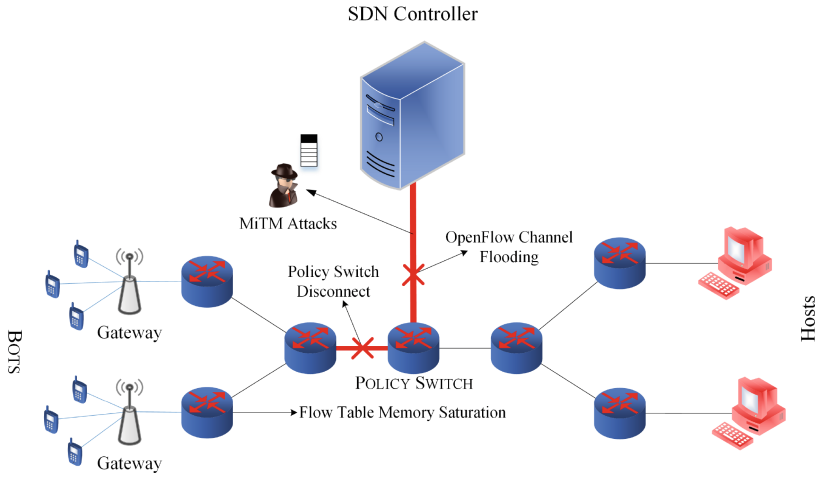

Fig. 3. Attack vulnerabilities in SDIoT-Edge.

SDIoT-Edge. When a flow arrives at the switch, it performs a flow table lookup and processes the flow using one of the three modes, including proactive, reactive, and hybrid. In the proactive mode, the network administrators proactively install the intended flow rules on the data plane switches to reduce the packet processing time. In a reactive mode, the flow rules are installed after the packets arrive at the switch using a PACKET_OUT message from the controller, whereas the hybrid mode uses both strategies to manage the flow rules. Although the choice of proactive flow rule installation method seems promising, the switches possess a meager Ternary Content Addressable Memory (TCAM) which cannot store a huge number of flow rules. Therefore, the reactive flow rule installation method is necessary to serve diverse traffic in the network. However, the reactive flow rule installation method can be exploited by the adversaries to transmit the flood of specially crafted flows to the switches which continuously transfer the requests to the controller for the flow rule installation and congest the OpenFlow channel.

4. **Offloading:** Although IoT devices are equipped with sophisticated sensors which continuously collect and transmit data, they do not possess the resources to perform compute-intensive tasks. The latency-sensitive applications in IoT suffer due to long waiting time induced by the central cloud data center in serving their requests. Alternatively, edge nodes encompass the resources to perform the offloaded tasks from the latency-sensitive IoT infrastructure. Moreover, data filtering and classification can be performed on the edge nodes to avoid unnecessary resource consumption in terms of bandwidth, storage, and energy. However, offloading and downloading of data incorporate many security, privacy, and data provenance issues.

Keeping in view the above-mentioned vulnerabilities in the SDIoT-Edge infrastructure, we propose SIESec solution to secure this infrastructure against DDoS attacks.

3 SIESec Solution

In this section we present SIESec solution, which provides defense against OpenFlow channel DDoS attacks on SDIoT-Edge architecture.

3.1 Adversary Model

The IoT-based DDoS have become one of the most devastating attacks against the current data center networks [8,11]. In the OpenFlow channel DDoS attack, the adversary exploits the vulnerabilities present in the IoT devices and flood the OpenFlow channel of the SDIoT-Edge infrastructure. We assume that the IoT manipulating adversary has the following capabilities.

- The adversary can access the IoT devices attached to an SDN, moreover, it can manipulate these devices to send attack packets to the other hosts in the network.
- The adversary can program IoT devices to send carefully crafted flood packets which cause packet miss in the switches at the infrastructure plane.
- The adversary ascertains the information of the victim's network using probing packets, including topology, network hierarchy, ingress switches, and packet miss information.
- The SDIoT-Edge network employs a reactive flow rule installation mechanism which has been widely used to provide flexible network provisioning [21,24].
- The adversary attacks the data plane switches using carefully crafted flood packets which cause packet miss and trigger new flow rule installation.

The adversary initially exploits topology discovery commands to inspect the network structure. Then it sends probing packets to the attached hosts to analyze the packet miss strategy by changing the packet header information and analyzing the Round Trip Time (RTT). When a packet miss occurs, its RTT increases as for a packet miss, the switch needs to request the controller to install a new flow rule using a PACKET_IN message. Subsequently, the controller replies with a PACKET_OUT message containing the flow rule for the packet, which increases the RTT. The adversary analyzes maximum different packet header combinations which cause packet miss and stores this information to attack the network. Furthermore, the adversary sends a flood of specially crafted attack packets to the network, which causes packet miss in the OpenFlow switches. Consequently, the controller is forced to install flow rules for a large number of new packets which causes extra overhead on the controller and impedes the flow rule installation process. With a further increase in the attack traffic, the controller becomes irresponsive, and OpenFlow channel turns into a congestion

state. The attack invokes increase in delay, RTT, extra utilization of controller CPU, and bandwidth saturation of the OpenFlow channel.

SIESec performs collaborative network measurements by exploiting the centralized control strategy of SDN and then deploys an unsupervised machine learning SOM algorithm to classify the network traffic. We explain the SOM classification strategy in the next section.

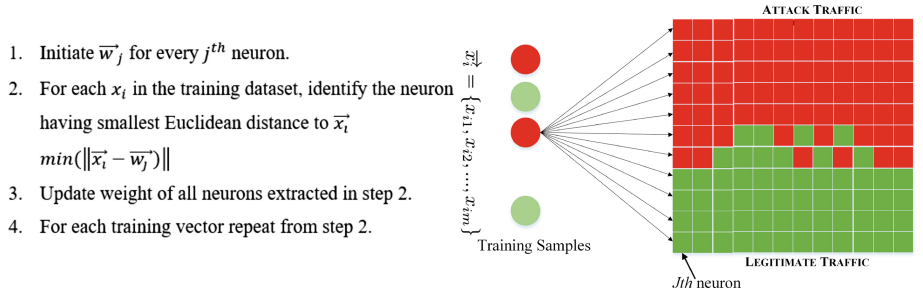

Fig. 4. Steps in SOM classification and a two dimensional graphical representation of training samples.

3.2 Self Organizing Maps Classification Algorithm

We employ an artificial neural network-based machine learning algorithm called SOM for the traffic classification. It first creates a randomized two-dimensional map of the training dataset. Then a data point is randomly selected on the map whereas a neuron called as Best Matching Unit (BMU) is chosen based on lowest Euclidean Distance and brought closer to the data point. The distance that the BMU covers is called the learning rate, which decreases after every iteration. Subsequently, the neighbors of the BMU are also moved closer to the data point to complete the first iteration. Furthermore, the learning rate and Euclidean Distance of BMU are recomputed for the next iteration. This process continues until the neurons in the grid take the shape of the data and finally reveal the intrinsic clusters in the dataset.

The SOM machine learning algorithm represents the network training samples to a set of neurons at a higher dimension and align them to a lower dimension during the classification task. The training process builds a model based on input features, and the mapping process classifies the traffic based on the lowest Euclidean Distance values. The algorithm to compute SOM is illustrated in Fig. 4 which describes four steps to classify the DDoS traffic. A two-dimensional SOM strategy is employed where the weight vector at j_{th} neuron having an m dimension is computed by the Eq. 1.

$$\overrightarrow{w_j} = [w_{j1}, w_{j2}, ..., w_{jm}] \tag{1}$$

The weight value of every neuron is assigned in a random manner where the feature values are constrained in a 0 to 1 range. The BMU neuron is selected using the Eq. 2.

$$\overrightarrow{w_i^*} = \underset{\forall \overrightarrow{wj} \in W}{min} \sqrt{\sum_{k=1}^{m} (x_{ik} - x_{ik})^2} \tag{2}$$

Where x_i is the i_{th} training sample which can be represented by the Eq. 3.

$$\overrightarrow{x_i} = [x_{j1}, x_{j2}, ..., x_{jm}] \tag{3}$$

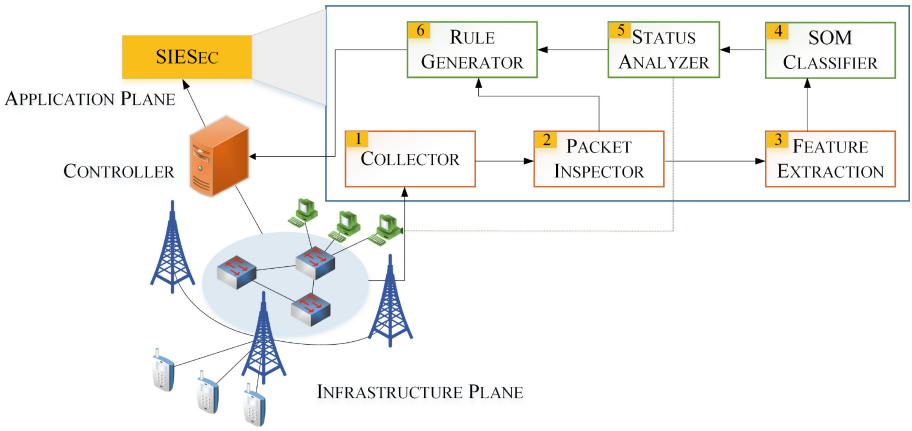

Fig. 5. Workflow of SIESec solution.

The weight of the competing neurons is finally computed in order to bring their values close to the training samples. The input to the SOM in SIESec is 6 features as discussed in the next section whereas the benign training samples were labeled manually.

3.3 Work-Flow of SIESec

SIESec is composed of six modules, as illustrated in Fig. 5. Overall, network security is ensured using the security solution. The detail of all the modules is discussed in the following.

1. **Collector:** This module continuously collects the network statistics by exploiting Representational Estate Transfer (REST) API of the open-source Floodlight controller, including the switch, packet, and flow-level statistics. It stores 6 statistics including source IP, average packet loss rate, time duration per flow, bandwidth consumption, overall link bandwidth, and packet drop rate.
2. **Packet Inspector:** The packet inspector performs two packet matching operations, including blacklist and contextual information inspection. The adversary tries to inject malicious traffic continuously using the compromised

IoT devices during the attack. Therefore, we employ a malicious packet identification database for the already identified adversaries to speed up the detection process. We keep updating the database as and when an adversary is identified. The traffic from the collector comes to the blacklist inspector, which matches the packet source with the database entries. If the incoming packet source is matched with an entry in the database, the control is forwarded to the flow rule generator, which requests the controller to generate a flow rule to drop this packet. The packet inspector also incorporates the contextual information collection, which is a vital source for the traffic filtering in the IoT network [6]. The packet inspector pre-filters the traffic based on contextual features, e.g., a compromised temperature sensor transmitting out of limit temperature values.

3. **Feature Extraction:** This module extracts the features by preprocessing the input samples. It removes extra packet header information including ack and syn-ack packets and presents a set of features to the classifier.
4. **Classifier:** The classifier employs SOM, an unsupervised classification technique to segregate the adversarial and benign traffic. We provide a manually labeled training dataset to generate the model and then classify the traffic at runtime. After the attack traffic classification, the SOM classifier forwards this information to the status analyzer.
5. **Status Analyzer:** It analyzes the traffic status classified by the SOM and forwards the malicious traffic information to the flow rule generator and directs the benign traffic toward the destination.
6. **Rule Generator:** This is the final operation in the SIESec solution, it requests the controller with the malicious flow packet identity to generate a flow rule to drop this packet. Many techniques can be applied to block or mitigate the flooding flows, including null routing, scrubbing, and dropping a flow. However, we utilize the flow-drop strategy to eliminate the malicious packets because it poses only a minor computation overhead on the network. Subsequently, the source IP of the malicious flow is added to the blacklist database, which can be utilized for future traffic filtering. An important feature of SIESec solution is that it advocates reuse, where the information of a malicious adversary can be stored and reused in the future. Therefore, it saves extra effort on the classification of already identified adversaries.

The collector module continuously collects network statistics using the REST API and provide the features to the SOM classifier. All the traffic from the SDIoT-Edge should pass through the SDN switches where the surveillance is performed using the security solution. The collector obtains network statistics from the infrastructure plane and provides it to the packet inspector, which filters the traffic packets for the identification of blacklists and contextual information. If any of the two filtering operations is true, a notification is transmitted to the flow rule generator with the packet information, which requests the controller to drop the identified flow. Subsequently, the traffic is forwarded to the feature extraction module which performs the pre-processing on the data and forwards the traffic to the SOM classifier. This module classifies the DDoS traffic

and transfers the results to the status analyzer, which moves the benign traffic toward the hosts and malicious traffic to the flow rule generator. The controller is requested to generate flow-drop rules for the identified malicious flows. It is pertinent to note here that when an adversary is detected, the source is added to the blacklists to enhance the traffic filtering in the future.

In this section, we elaborated the SIESec solution. In the next section, we discuss the experimental evaluation of SIESec.

4 Experiment Evaluation

SIESec acts as an application in the application layer of SDN. All the network measurements have been performed using the SDN controller, which poses a minimal network overhead. We use iperf tool to generate traffic from the IoT hosts, moreover, we utilize Mininet network emulator and Floodlight open-source controller for experimentation. The controller was running on a Windows 10 machine with an Intel Core i7 processor and 16 GB of RAM, whereas the Mininet emulator was configured on a Ubuntu 16.0.4 operating system running on an Oracle-Virtualbox, virtual machine manager. The input parameters for the SOM classifier includes 6 features, 2 output neurons, and a learning rate of 0.4. The training features have been manually labeled in both legitimate and DDoS traffic scenario. We emulate the network topology, as shown in Fig. 6. Moreover, we employ the following traffic features.

1. Source IP
2. Average of packet loss rate
3. Time duration per flow
4. Bandwidth consumption
5. Overall link bandwidth
6. Packet received rate.

4.1 Attack Setup

We employ the network topology represented in Fig. 6 for the experimentation, which represents four clusters of IoT devices connected with the edge gateways. The adversary utilizes a huge repository of IoT devices at three clusters to send DDoS traffic on the network. Similarly, a cluster of legitimate devices sends benign traffic toward the destination hosts in the network. Therefore, the network contains both legitimate and DDoS attack flows. The bandwidth of the OpenFlow channel was set to 1 Gbps, and the attack rate varied from o to 1000 Packets Per Second (PPS).

4.2 Results

We use four performance metrics to evaluate SIESec as represented in Fig. 7. When the attack occurs on the network, it decreases the available bandwidth of the OpenFlow channel, increases delay, RTT, and CPU utilization. The results of the experimentation are illustrated in the following.

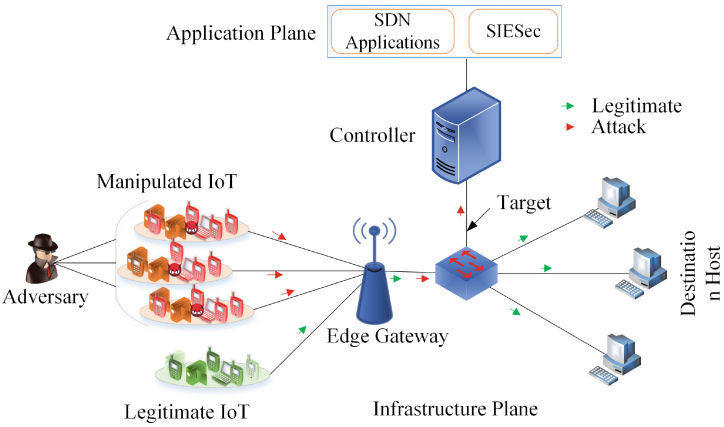

Fig. 6. The experimental topology illustrating a huge number of compromised IoT devices sending DDoS traffic to the network.

4.3 Available Bandwidth

In this experiment, we send a flood of specially crafted packets to the switches in the data plane causing packet miss with an attack packet rate from 0 to 1000 PPS. The bandwidth is measured during each round of the experiment. The experiment was conducted with and without SIESec solution. In the presence of SIESec, the available bandwidth drops initially due to the flow rule installation for the benign traffic, which further becomes stable. In the second experiment, we run the attack without SIESec solution, and the bandwidth is measured after 100 PPS attack intervals. The results demonstrate that the bandwidth of the channel dropped to 0 when the attack rate reached 900 PPS. It can also be observed in Fig. 7a that the bandwidth of OpenFlow channel saturates rapidly with the increase in the attack packets. However, it remained stable throughout the experiment when SIESec solution was deployed. The evaluation using available bandwidth demonstrates the effectiveness of the SIESec to comprehensively maintain the available bandwidth of the OpenFlow channel during the attack.

4.4 Round Trip Time

We perform RTT experiment with and without SIESec and measure the RTT as represented in Fig. 7b. We analyze the RTT using ping command at the benign host during the attack. The graph without SIESec solution illustrates that the value of RTT increased significantly when the attack rate reached to 200 PPS, which further increased continuously and reached a value peak at 1000 PPS attack rate. Experiment with the SIESec solution demonstrates that the RTT increased slightly when the attack rate was 300 PPS, the reason behind this fluctuation was the training of the classifier which induced a slight time delay and increased the RTT. Subsequently, the RTT graph became stable, and the

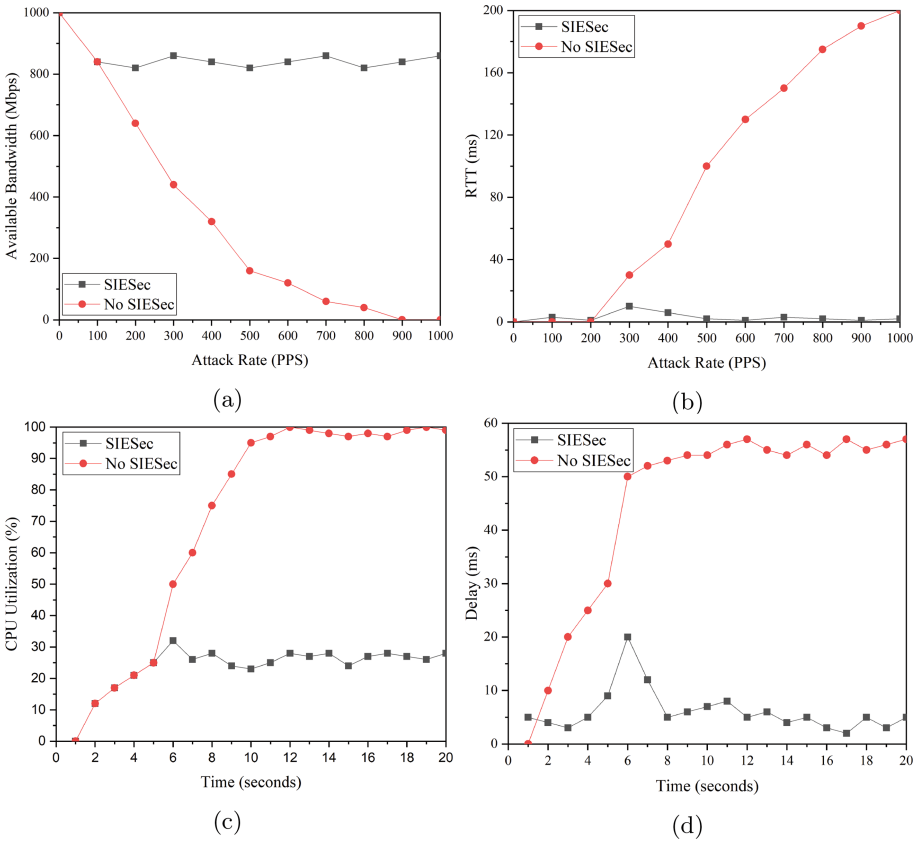

Fig. 7. Evaluation of SIESec using different evaluation measures.

RTT values remained closer to 0. This experiment demonstrates that the SIESec solution efficiently minimizes RTT during the OpenFlow channel DDoS attacks.

4.5 CPU Utilization of Controller

In this experiment, the CPU utilization of the controller is observed with and without SIESec solution, as illustrated in Fig. 7c. The CPU utilization during the attack increased from around 25% at 5 s to a peak value of up to 100%. The CPU utilization was around 25% at the start of the attack due to the normal traffic. However, with the increase in the attack traffic, the utilization continuously increased and reached to 100%. Alternatively, we perform the experiment with SIESec solution, as we can observe from Fig. 7c that the CPU utilization was stable and remained around 25% throughout the experiment. There is a slight increase in the utilization ratio near 6 s, where it reached around 33% due to the initial attack detection latency. However, further utilization remained stable until the end of the experiment, other than a negligible uplift at the start of

the experiment. This experiment demonstrates that SIESec efficiently manages controller CPU utilization during the attack.

4.6 Delay

In this experiment, we measure the traffic delay with and without SIESec during the OpenFlow channel DDoS attack. We perform the delay experiment two times with and without SIESec, the results of the experiment are illustrated in Fig. 7d. In this experiment, we measured the delay of the legitimate traffic during the attack, for this, we exploit the legitimate IoT devices to send traffic packets to the other hosts at the destination and measure the delay. We run the delay experiment multiple times and record the average value of the experiment at each step and plot the graph. The experiment without SIESec revealed significantly higher values of delay. In the second experiment, we deploy SIESec solution and run the experiment again. Figure 7d illustrates that the delay increased when the time was 6 s and reached up to 20 ms due to the delay during the training of the SOM model. Subsequently, the graph becomes stable where the delay remains around 5 ms during the rest of the experiment.

The experimental evaluation of SIESec solution portrays that SIESec is effective in systematically alleviating the OpenFlow channel DDoS attacks based on the available bandwidth, RTT, CPU utilization, and delay. SIESec actively mitigates the attack and introduces a minimal network overhead. As the SDIoT-Edge infrastructure has been increasingly deployed in the current networks, this technique provides a comprehensive solution against DDoS attacks. As the other DDoS attack mitigation techniques deploy hardware-based measurements for network statistics collection or traffic rerouting, we perform all the measurements using the SDN-based centralized control. Therefore, SIESec solution invokes a minimal overhead and efficiently provides security against OpenFlow channel DDoS attacks.

5 Related Work and Comparison Analysis

IoT is capturing tremendous attention from academia and industry during the past few years. However, the security vulnerabilities in IoT pose a vital challenge for the network community and effective realization of IoT. With the widespread adoption of IoT, the issues of security assessment and devising defense mechanisms are of prime concern for the network security researchers. SDN provides centralized network management by separating infrastructure and control planes. This separation provokes a flexible network evolution and programmability. Therefore, SDN is considered as the best choice for IoT-Edge networks [13,15]. However, the adoption of SDN for IoT management also impels numerous security challenges [3,10,22].

The integration of IoT with fog computing using SDN has been proposed by [20] whereas the security vulnerability assessment in IoT has been performed by [25]. Authors in [10] propose MiTM attacks in SDN IoT-fog infrastructure.

They provide an experimental evaluation on how the vulnerabilities in IoT can be exploited by the adversaries to attack the OpenFlow channel, including information spoofing, topology faking, and information theft attacks. A security solution employing multi-hop routing technique has been proposed in [26] where a multi-path route can be computed by identifying the neighbors, their location, and energy of the sensory devices.

In the DDoS reflection attack, the source sends a minimal query to the IoT device, which replies with a long message to the victim. In [17], authors propose that the IoT devices suffer from the vulnerability of DDoS reflection attacks. They demonstrate that the household devices can be exposed to these attacks besides being protected by the gateways. Authors in [9] propose a DDoS solution against IoT using a fast communication channel to actively detect and defend these attacks. An attack graph can be used to identify probable attack routes, where securing the route can proactively mitigate the DDoS attack. A graph-based method to detect the sequence of paths that the adversary follows during an attack in Industrial internet of things has been provided in [14]. A lightweight solution against bandwidth attacks using intrusion prevention technique in IoT has been presented in [5]. However, this mechanism is hard to implement in IoT as it needs high computation power, which is not available in the current IoT infrastructure.

Defense techniques against security in SDN can be divided into two categories, including data plane security [7] and the control plane defense [27]. FloodDefender [21] is a security solution against resource saturation attacks on both control and data planes. It employs traffic filtering, table miss analysis, and flow migration to defend against DoS attacks. However, this technique induces more delay in the network traffic due to the complex analysis and time-consuming rule migration. Similarly, FloodGuard considers DoS attack strategy where only one adversary sends flood packets. However, SIESec provides defense against DDoS attacks in SDIoT-Edge, where a huge number of IoT devices flood the OpenFlow channel. SGuard [23] provides access control using a classification strategy, however, the complex measurements in this technique increase overhead on the network. BWManager [27] provides defense against DoS attacks on the controller by using a scheduling strategy to process the flow requests. However, this technique also induces traffic overhead by directing the traffic to follow the round-robin scheduling strategy. Moreover, CyberPulse [18] provides defense against OpenFlow channel LFA using machine learning techniques. However, this solution follows a direct attack strategy on the OpenFlow channel, which differs from our proposed attack and defense mechanism.

The difference between the previous techniques and SIESec is that the previous solutions do not consider the complex SDIoT-Edge paradigm. Similarly, these techniques perform complex network measurements using specialized hardware or software agents. However, SIESec provides comprehensive security without posing extra overhead on the network. Besides, SIESec employs blacklist and contextual information filtering to mitigate DDoS traffic interactively. Although the SIESec provides promising benefits, the actual implementation of SDIoT-

Edge will precisely reveal the efficiency in a practical paradigm. Moreover, SOM classification strategy may suffer in some cases as it needs sufficient training samples to classify the attack traffic accurately. However, the cost-benefit analysis of SIESec makes it an efficient solution against DDoS attacks in SDIoT-Edge paradigm.

6 Conclusion and Future Work

The huge proliferation of decentralized IoT devices poses a vital challenge in the network management. Software defined networking due to its capability of flexible network management has been proposed to manage IoT infrastructure. Edge computing brings cloud resources near to the IoT devices to overcome resource-limitation bottleneck in IoT. Therefore, SDIoT-Edge integration provides a resourceful platform to enable efficient IoT service orchestration.

In this research, we first proposed the architecture of SDIoT-Edge infrastructure and then provided a novel security solution against DDoS attacks in SDIoT-Edge. Nevertheless, the SDN infrastructure of IoT-Edge provides promising features, the integration of diverse platforms pose several security challenges. To overcome the vulnerability of DDoS attacks on the OpenFlow channel, we presented a security solution named as SIESec. The proposed SIESec solution employs machine learning-based SOM classification algorithm, blacklist integration, and contextual information filtering of the malicious IoT traffic to provide defense against DDoS attacks. The experiments performed using Mininet network emulator, and Floodlight open-source controller demonstrates that SIESec provides an efficient solution against OpenFlow channel DDoS attacks and poses a minimal network overhead.

A scalable solution can be developed for large-sized SDIoT-Edge networks by extending SIESec using SOM filtering on smaller SDIoT-Edge network segments, and then a centralized security solution for the global network can be implemented. In future work, we plan to implement this solution using multiple algorithms and provide the evaluation using a physical testbed.

Acknowledgment. This research is supported by the National Science Foundation of China under Grant No. 61672276 and 61702277 and the Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing University.

References

1. SDN architecture. https://www.opennetworking.org/wp-content/uploads/2013/02/
2. DNS products trusted by the worlds most admired digital brands (2019). http://dyn.com/dns/
3. Administrator: MMD-0056-2016 - Linux/Mirai, how an old ELF malcode is recycled (2016). http://blog.malwaremustdie.org/2016/08/mmd-0056-2016-linuxmirai-just.html

4. Afshar, V.: Cisco: Enterprises are leading the internet of things innovation (2017). https://www.huffingtonpost.com/entry/cisco-enterprises-are-leading-the-internet-of-things_us_59a41fcee4b0a62d0987b0c6
5. Aldaej, A.: Enhancing cyber security in modern internet of things (IoT) using intrusion prevention algorithm for IoT (IPAI). IEEE Access (2019, In press)
6. Aleroud, A., Karabatis, G.: Contextual information fusion for intrusion detection: a survey and taxonomy. Knowl. Inform. Syst. **52**(3), 563–619 (2017)
7. Ambrosin, M., Conti, M., De Gaspari, F., Poovendran, R.: LineSwitch: tackling control plane saturation attacks in software-defined networking. IEEE/ACM Trans. Netw. **25**(2), 1206–1219 (2017)
8. Baker, C.: Recent IoT-based attacks: what is the impact on managed DNS operators? (2016), http://dyn.com/blog/dyn-analysis-summary-of-fridayoctober-21-attack/
9. Bhardwaj, K., Miranda, J.C., Gavrilovska, A.: Towards IoT-DDoS prevention using edge computing. In: {USENIX} Workshop on Hot Topics in Edge Computing (HotEdge 2018), Boston, MA (2018)
10. Cheng, L., Qin, Z., Novak, E., Li, Q.: Securing SDN infrastructure of IoTfog networks from MitM attacks. IEEE Internet Things J. **4**(5), 1156–1164 (2017)
11. De Donno, M., Dragoni, N., Giaretta, A., Spognardi, A.: DDoS-capable IoT malwares: comparative analysis and Mirai investigation. Secur. Commun. Netw. **2018** (2018)
12. Deng, S., Gao, X., Lu, Z., Li, Z., Gao, X.: Dos vulnerabilities and mitigation strategies in software-defined networks. J. Netw. Comput. Appl. **125**, 209–219 (2019)
13. Farris, I., Taleb, T., Khettab, Y., Song, J.: A survey on emerging SDN and NFV security mechanisms for IoT systems. IEEE Commun. Surv. Tutor. **21**(1), 812–837 (2019)
14. George, G., Thampi, S.M.: A graph-based security framework for securing industrial IoT networks from vulnerability exploitations. IEEE Access **6**, 43586–43601 (2018)
15. Jararweh, Y., Al-Ayyoub, M., Benkhelifa, E., et al.: An experimental framework for future smart cities using data fusion and software defined systems: the case of environmental monitoring for smart healthcare. Future Gener. Comput. Syst. (2018, In press)
16. Jararweh, Y., et al.: Software-defined system support for enabling ubiquitous mobile edge computing. Comput. J. **60**(10), 1443–1457 (2017)
17. Lyu, M., Sherratt, D., Sivanathan, A., Gharakheili, H.H., Radford, A., Sivaraman, V.: Quantifying the reflective DDoS attack capability of household iot devices. In: Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks, pp. 46–51. ACM, Montreal (2017)
18. Rasool, R.U., Ashraf, U., Ahmed, K., Wang, H., Rafique, W., Anwar, Z.: Cyberpulse: a machine learning based link flooding attack mitigation system for software defined networks. IEEE Access **7**, 34885–34899 (2019)
19. Sabet, K.A.: IoT revenue opportunity to exceed $1 trillion by 2025 (2018). https://www.itpro.co.uk/internet-of-things-iot/31218/iot-revenue-opportunity-to-exceed-1-trillion-by-2025
20. Salman, O., Elhajj, I., Chehab, A., Kayssi, A.: IoT survey: An SDN and fog computing perspective. Comput. Netw. **143**, 221–246 (2018)
21. Shang, G., Zhe, P., Xiao, B., Hu, A., Ren, K.: FloodDefender: protecting data and control plane resources under SDN-aimed DoS attacks. In: IEEE Conference on Computer Communications (INFOCOM), Atlanta, GA, USA, pp. 1–9 (2017)

22. Sunnyvale, C.: Proofpoint uncovers internet of things (IoT) cyberattack (2014). https://docplayer.net/16470381-Proofpoint-uncovers-internet-of-things-iot-cyberattack.html
23. Tao, W., Chen, H.: SGuard: a lightweight sdn safe-guard architecture for DoS attacks. Chin. J. **14**(6), 113–125 (2017)
24. Wang, H., Xu, L., Gu, G.: FloodGuard: a DoS attack prevention extension in software-defined networks. In: IEEE/IFIP International Conference on Dependable Systems and Networks, Washington, DC, USA (2015)
25. Wang, H., Chen, Z., Zhao, J., Di, X., Liu, D.: A vulnerability assessment method in industrial internet of things based on attack graph and maximum flow. IEEE Access **6**, 8599–8609 (2018)
26. Wang, J., Miao, Y., Zhou, P., Hossain, M.S., Rahman, S.M.M.: A software defined network routing in wireless multihop network. J. Netw. Comput. Appl. **85**, 76–83 (2017)
27. Wang, T., Guo, Z., Chen, H., Liu, W.: Bwmanager: mitigating denial of service attacks in software-defined networks through bandwidth prediction. IEEE Trans. Netw. Serv. Manage. **15**(4), 1235–1248 (2018)
28. Weiser, M.: The computer for the 21st century. IEEE Pervasive Comput. **1**(1), 19–25 (2002)