A Multi-Level DDoS Mitigation Framework for the Industrial Internet of Things

Qiao Yan, Wenyao Huang, Xupeng Luo, Qingxiang Gong, and F. Richard Yu

This article proposes a multi-level DDoS mitigation framework (MLDMF) to defend against DDoS attacks for IIoT, which includes the edge computing level, fog computing level, and cloud computing level. Software defined networking is used to manage a large number of IIoT devices and to mitigate DDoS attacks in IIoT.

Abstract

The Industrial Internet of Things is growing fast. But the rapid growth of IIoT devices raises a number of security concerns, because the IIoT device is weak in defending against malware, and the method of managing a large number of IIoT devices is awkward and inconvenient. This article proposes a multi-level DDoS mitigation framework (MLDMF) to defend against DDoS attacks for IIoT, which includes the edge computing level, fog computing level, and cloud computing level. Software defined networking is used to manage a large number of IIoT devices and to mitigate DDoS attacks in IIoT. Experimental results show the effectiveness of the proposed framework.

Introduction

The Internet of Things (IoT) has been developed tremendously due to advancements of a variety of technologies, such as sensors, wireless communications, and computing. There are a lot of applications of IoT, which range from common home and personal appliances to large-scale and safety critical systems [1, 2]. The recent Dyn event has raised our concerns about the security of IoT. On October 21, 2016, the Mirai IoT botnet was utilized by attackers to launch high-impact distributed denial of service (DDoS) attacks against the Dyn DNS service, which caused an extended Internet outage. The Mirai IoT botnet is composed of Internet-enabled digital video recorders (DVRs), surveillance cameras, and other Internet-enabled embedded devices.

An IoT device is at great risk of security vulnerabilities, because there are many distinctions between IoT and traditional Internet devices. For example, the number of IoT devices is much greater. Nodes of IoT are limited in resources and dedicated, diversified communication protocols are used in IoT, and so on. Some of these differences weaken the ability of IoT nodes to protect themselves.

IoT is connecting smart things, such as intelligent devices and sensors, to the Internet. The data collected by smart things is sent to a central cloud-based service that processes all the gathered data and shares these data with users [3, 4]. Combining the IoT concept and industrial wireless sensor networks (IWSNs) may reap the full benefits of the IoT. The Industrial IoT (IIoT) is going to reform manufacturing by enabling manageable IIoT devices to interact with each other and to join different parts of manufacturing by using networks. On one hand, IIoT profits from IoT. On the other hand, it also bears the same security risk as IoT. Although IoT and IIoT have different architectures, they face common network and host security threats due to limited resources. Traditional ways of preventing DDoS attacks are insufficient to satisfy security requirements. Security guarantee is essential to IIoT. Due to distributed sensor nodes, actuators, and machines connected to form the production system, it becomes nontrivial to ensure the authenticity and data confidentiality of the system [5]. Thus, new ways must be found to defend against DDoS attacks.

In this article, a multi-level DDoS mitigation framework (MLDMF) for IIoT is proposed, which includes the cloud computing level, fog computing level, and edge computing level. Software defined networking (SDN) is used to manage the network. They are combined to improve access security and efficient management of IIoT.

The rest of the article is organized as follows. We present related works on DDoS in IoT and IIoT environments. MLDMF is proposed. Experimental results are presented to show the effectiveness of managing devices and defending against DDoS attacks in the framework. We then conclude this work.

Related Work

DDoS flooding attacks can be classified into two categories based on the targeted protocol level [6]. They are network-level DDoS flooding attacks and application-level DDoS flooding attacks. Network-level DDoS flooding attacks are mostly launched by using TCP, UDP, ICMP, and DNS protocol packets, which aim to exhaust a target's network I/O bandwidth. Application-level DDoS flooding attacks focus on disrupting a legitimate user's services by exhausting the server resources, such as sockets, CPU, memory, disk/database bandwidth, and I/O bandwidth [6]. With the development of new Internet technologies, DDoS attack traffic is growing and creating a new record. As our previous work discussed [7], the size and frequency of DDoS attacks have grown in the cloud computing environment due to the essential characteristics of cloud computing including on-demand self-service, broad network access, resource pooling, rapid elasticity, and measured service. Recent DDoS attacks launched by using Mirai IoT botnets have created a new record of DDoS attack traffic.

Digital Object Identifier:
10.1109/MCOM.2018.1700621

Qiao Yan, Wenyao Huang, Xupeng Luo, and Qingxiang Gong are with Shenzhen University; F. Richard Yu is with Carleton University.

There are significant differences between traditional Internet and IIoT. There are many reasons to explain why the new DDoS attack record is created in IIoT environments. We just list some important points here. First, the quantity of IIoT devices is particularly large and continues to grow, which makes managing them difficult. Second, IIoT devices have unsubstantial defense systems due to limited resources. Third, lots of IIoT devices use wireless communications and can be accessed from a public network easily. Fourth, many IIoT devices can be physically accessed because they are unsupervised. IIoT devices such as distributed control system (DCSs) and industrial DVR are still computer systems using Windows or Linux. Thus, an IIoT device can easily get infected with malware and contaminate other devices conveniently. On one hand, IIoT devices are liable to be recruited by the botnet. On the other hand, IIoT devices easily become victims of the DDoS attacks because of very limited resources. DDoS attacks targeting IIoT devices are much more dangerous, because the service of IIoT devices is relative to appliances of large-scale and safety-critical systems. All of the above reasons show that DDoS attacks will be more violent and more fatal in IIoT.

With the development of IoT, the research on DDoS attack protection for IoT is growing. In 2016, Mauricio Tellez *et al.* [8] demonstrated how the BSL password could be brute forced in a matter of days by discovering significant patterns between passwords. Besides, they reverse engineered wireless sensor network (WSN) applications to obtain critical security information such as encryption keys. In [9], Pacheco *et al.* analyzed the effectiveness of DDoS attacks in a typical IoT environment through simulation. The DDoS attack is effective because bandwidth and computing resources in the IoT environment are limited. In [10], the authors presented the viewpoint that low-end and low-energy devices in IoT must devote most of their available energy to executing core applications. Some essential limits in IoT devices make conventional defense methods fail, such as limited energy and limited RAM. They also believed cryptographic security is all-or-nothing by nature, which means lightweight cryptographic algorithms are essentially no security. Dedicated cryptographic circuitry seems promising, because it can meet the requirements. In [11], the authors proposed an IoT honeypot and sandbox, which attracts and analyzes Telnet-based attacks against various IoT devices. It is helpful to know IoT state by analyzing the observation results of an IoT honeypot and captured malware samples.

IoT is creating massive data for connecting industry manufacturing to the Internet. Cloud computing and IoT complement each other because the cloud is powerful enough to process massive data generated by IoT. Cloud computing is widely used in IoT, but most cloud services have to be handled in data centers that are far away from IoT devices. Transferring between a cloud center and an IoT device cannot guarantee the low latency that some IoT applications need [12, 13]. Besides, transferring substantial data from IoT devices to cloud centers is inefficient and at risk of divulging privacy.

Fog computing is an architecture that spreads cloud computing and service to a place in the network that is most efficient to serve users. Fog computing is proposed to meet the requirements that some IoT applications need: high-speed data processing, analytics, and shorter response time. The powerful capability of fog computing nodes not only can meet the requirement of low latency, but also can be used to secure IoT efficiently [14].

The original intention of edge computing is pushing resource of computing and data to the logical extremes of the network. Edge computing is a local model of computing, which is supposed to offer much faster response by avoiding transferring data to a remote server. In general, an edge computing network consists of terminal equipment (i.e., mobile phones, intelligent devices, etc.), peripheral equipment (i.e., boundary routers, wireless access points, etc.), and so on. The difference between an edge computing network and a fog computing network is commonly considered to be that edge computing is not part of cloud service, such as infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS) [14].

SDN has attracted great interest as a new paradigm in networking, which brings numerous benefits by decoupling the control plane from the data plane including dynamic, manageable, cost-effective, and adaptable properties. SDN provides networking programming ability and centralized management so that an SDN-based IIoT gateway can manage IoT devices through SDN protocols. SDN has been used to improve the efficiency of machine-to-machine (M2M) communication in IIoT. In [15], a software-defined IIoT architecture is proposed in the context of Industry 4.0 to effectively improve the interaction between machines so that the assigned task can be finished quickly. However, IIoT faces some problems. The authors think the possible solution for data safety and system reliability is to design IIoT architecture as an SDN-based framework because SDN can manage a network using OpenFlow or another SDN southbound Interface.

SDN has attracted great interest as a new paradigm in networking, which brings numerous benefits by decoupling the control plane from the data plane including dynamic, manageable, cost-effective, and adaptable properties. SDN provides networking programming ability and centralized management so that an SDN-based IIoT gateway can manage IoT devices through SDN protocols.

A Multi-Level DDoS Mitigation Framework for IIoT

A multi-level DDoS mitigation framework (MLDMF) for IIoT is proposed, as shown in Fig. 1. The IIoT architecture can be divided as the following three major layers: the perception layer, network layer, and application layer. Defending against DDoS attacks for IIoT is difficult due to IIoT's unsubstantial defense system, tremendous number of perception nodes, and traditional management method. Therefore, MLDMF is proposed to defend against DDoS attacks on IIoT. Corresponding to the three layers of IIoT, there are also three levels in MLDMF. They are the edge computing level, fog computing level, and cloud computing level from bottom to top. We believe more nodes in different places in IIoT should collaborate and cooperate in preventing, detecting, and responding to DDoS flooding attacks.

The edge computing level mainly consists of SDN-based IIoT gateways (SDNIGWs). An SDNIGW is able to connect to IIoT devices through all kinds of IIoT access protocols, including ZigBee, Z-Wave, RUBEE, WirelessHART, IETF6lowPAN, NFC, WiFi, Ethernet, second gen-

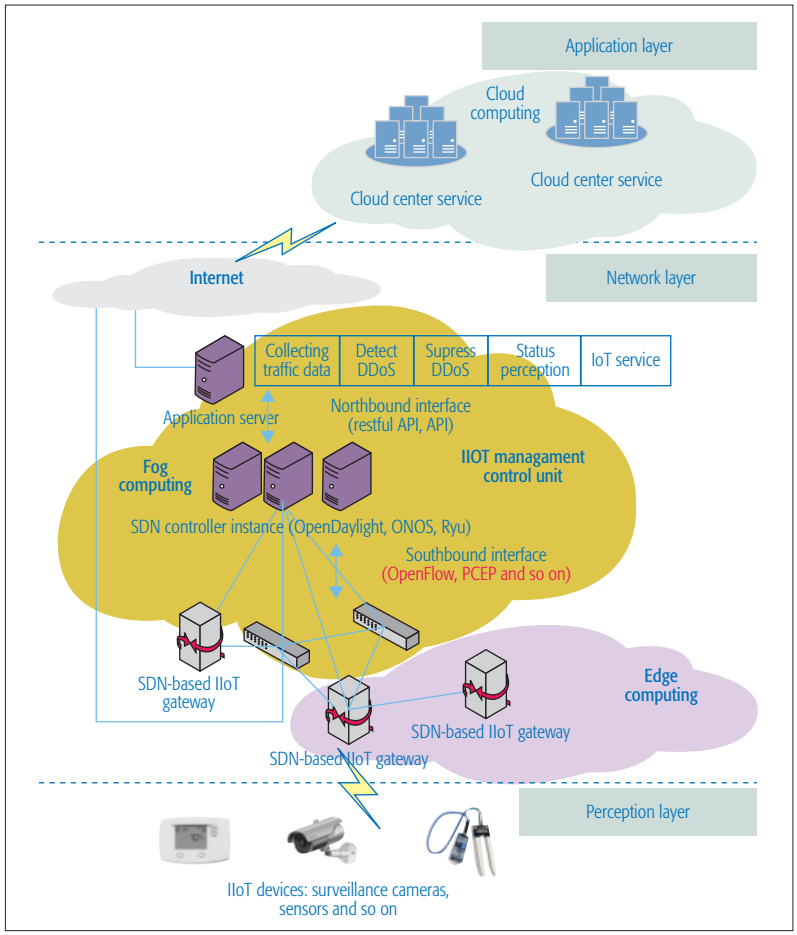

Figure 1. Overview of the proposed multi-level DDoS mitigation framework (MLDMF) for IIoT.

eration (2G)/3G/4G/5G, and so on. SDNIGW supports SDN protocols, and it can be managed by the controller at the fog computing level as shown in Fig. 1. The edge computing level can protect perception nodes its function is described later.

The edge computing level is placed in the network layer in IIoT, which serves to connect an IIoT device to the network and secure the IIoT device's security. The fog computing level is also placed in the network layer of IIoT, but the fog computing level is above the edge computing level. The fog computing level consists of SDN controller clusters and SDN application servers. SDN controller clusters connect SDN-supported switches through a southbound interface, such as OpenFlow or PCEP, which means SDN controller clusters can manage SDN-supported switches. Also, SDN controller clusters support network applications through a northbound interface. The fog computing level is used to provide low latency and high quality of service, which makes it a good place to improve the security of IIoT.

IIoT will produce large amounts of data, which need to be stored, processed, and accessed. Cloud computing is the major place to store and analyze the big data. It can be seen that detecting a DDoS flooding attack is relatively easier at the cloud computing level (victim), since all the flows can be observed at the destination. But it is difficult to respond to the attack flows, since huge attack flows mix with massive normal packets. On the contrary, it is desirable to respond to attack flows at the edge computing level (attack source) because attack flows are weak at that time.

In our framework, SDN is used to manage IIoT. As known, a centralized controller has a scalability problem including flow requests limitation and resiliency to failures. The scalability problem of SDN is compounded by dealing with two scalability problems at the same time. First, a large number of IIoT devices may reach the limitation of processing flow requests. And dealing with a mass of IIoT devices, a centralized controller may fail. It must have resiliency to failures. To tackle the issue of resiliency, backup servers are helpful to handle the problem of single point failure. For the flow requests limitation, in the framework, we try to solve the scalability problem of flow requests limitation in the design of the structure of the framework.

In the bottom layer of our framework, there are two thoughts to improve the ability of processing flow requests so as to relieve the flow requests limitation of a single SDN controller (i.e., SDNIGW in our framework). They use a powerful hardware platform to improve I/O performance and use common methods to reduce the number of flow requests. But if there are still too many IIoT devices for an SDNIGW, we need to add an extra SDNIGW to handle them, which makes the framework extensible at the edge computing level. MLDMF splits network functions between the edge and fog computing levels so that the stress of massive flow will not overwhelm the SDN controller. SDNIGW at the edge computing level is assigned to the following tasks: routing the local network traffic and routing the inter-SDNIGW network traffic. The SDN controller at the fog computing level, the IIoT management control unit (IMCU), is assigned to the following tasks: finding the path to navigate network traffic from source SDNIGW to destination SDNIGW and finding the path to navigate network traffic from source IMCU to destination IMCU.

In this design, the edge computing level is the data plane, which means that network traffic is flowing in the edge computing level. Local SDNIGW network traffic is routed by the SDNIGW itself. Inter-SDNIGW network traffic is transferred by the SDNIGW, but the routing decision is made by the IMCU. The IMCU uses a distributed SDN controller to manage the large area network formed by the SDNIGW. All these methods used by MLDMF are helpful to address the scalability issues.

Edge Computing Level

The edge computing level is shown in Fig. 2. It mainly consists of SDNIGWs. An SDNIGW provides management and access security functions for IIoT perception nodes based on traditional IIoT gateways. It can be controlled by the fog computing level using OpenFlow. SDNIGW is placed in the edge of a IIoT network that can under our control, which means it can be under surveillance to prevent unauthorized physical access. The SDNIGW is a device that can connect to power, so it can use a lot of traditional anti-malware software to protect itself and all the devices it manages without considering the power limit. The SDNIGW uses Safe Boot to validate the

firmware image before running, which helps to ensure that only authorized code will be executed before the system is loaded.

The SDNIGW uses the following mechanisms to protect the IIoT perception nodes it manages:

- Automatic connection: The SDNIGW can connect with IIoT perception nodes, which it manages based on the physical fingerprint of a device and a software hash value.
- Firmware security checks and updates automatically: The SDNIGW can check whether the IIoT perception nodes are in trusted, authorized firmware running on the device and can update device firmware automatically.
- Access control: The SDNIGW uses a strict access control mechanism.
- Concealing device IP address: The SDNIGW conceals a device's IP address to prevent unauthorized access to the device from the Internet.
- Malicious firmware/software detection: The SDNIGW uses a hash scheme and signature to detect malicious software and Trojans.
- Vulnerability scanning: The SDNIGW can detect any default or weak passwords of devices and open ports.
- Intrusion detection: The SDNIGW uses light detection methods, including white listing and statistical modeling, to detect possible intrusions.
- Attack reduction: The SDNIGW uses traffic filtering to reduce attacks.
- Honeypot monitoring: The SDNIGW can monitor the status of the honeypot nodes that are deployed in the IIoT perception layer.
- Communication data encryption: The SDNIGW uses a lightweight data encryption method to encrypt necessary data.

Fog Computing Level

The fog computing level is shown in Fig. 3 and mainly consists of the IMCU. The IMCU includes a cluster of SDN controllers and applications. We use Frenetic language, which can provide high-level abstractions and modular constructs to achieve the following security tasks. High-level abstractions and modular constructs offered by Frenetic will further open network capability and simplify the procedure of changing network settings. With these abilities, intrusion detection experts can help to protect IIoT based on their analysis much faster and easier.

- Collecting traffic data
- Detecting DDoS attacks based on network traffic data
- Restraining DDoS attacks based on detection
- Perceiving network state by using honeypots

The security functions are programmed in the SDN application plane. Then the functions are deployed to control plane through an SDN northbound application programming interface (API). Finally, they are implemented in the infrastructure plane through a southbound API.

The defending DDoS function of the fog computing level is shown in Fig. 4. There are three groups of methods used to defend against DDoS attacks. Different numeric symbols are used to distinguish these three groups: symbols ① ② ③ represent the collect-detect-mitigate (CDM) method, symbols (1) (2) (3) represent the honeypot-detect-react (HDR) method, and symbols 1 2 3 represent the cloud-detect-fog-mitigate (CDFM) method. These methods are described below.

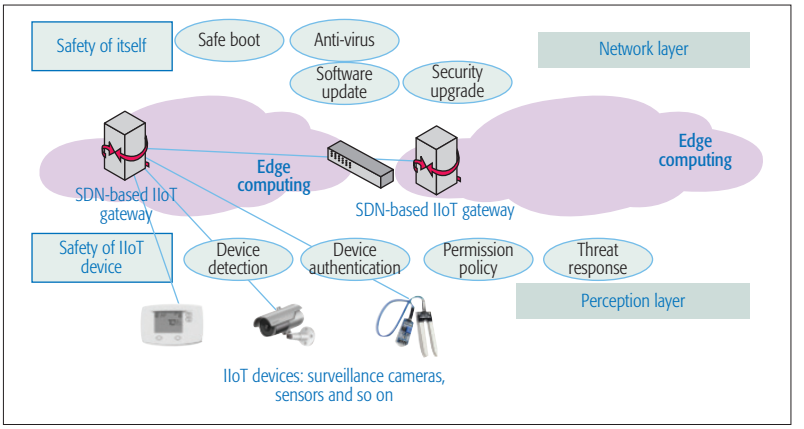

Figure 2. The edge computing level in the proposed framework.

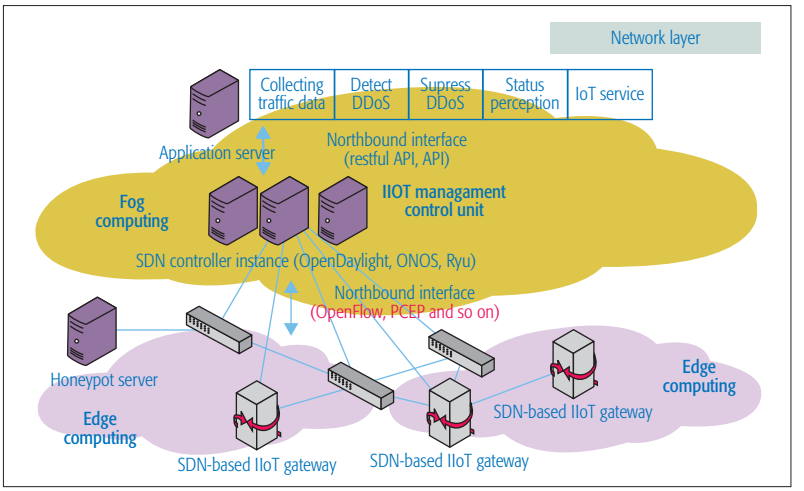

Figure 3. The fog computing level in the proposed framework.

The first method is CDM. CDM collects data in the range of IMCU management. NetFlow-enabled or sFlow-enabled switches are deployed at the edge network level. The IMCU collects network traffic data through sFlow or NetFlow. The IMCU will detect and analyze real-time traffic and then, based on analysis results and predefined policy, mitigate DDoS attacks by bandwidth throttling.

The second method is HDR. A honeypot is a computer security mechanism setting to detect and counteract attempts at unauthorized use of information systems. Many honeypot products have been developed, such as the Modern Honey Network (MHN). These traditional honeypot products are used to provide important information about network state. A honeypot can know what techniques are used by attackers, help capture malware and exploits, and help catch security breaches. A honeypot can also simulate an IoT device to capture and analyze IoT malware. An IMCU can redirect network traffic to a honeypot by using SDN. Based on the valuable information offered by a honeypot, the defense policy can become more accurate.

The third method is CDFM. It is a common

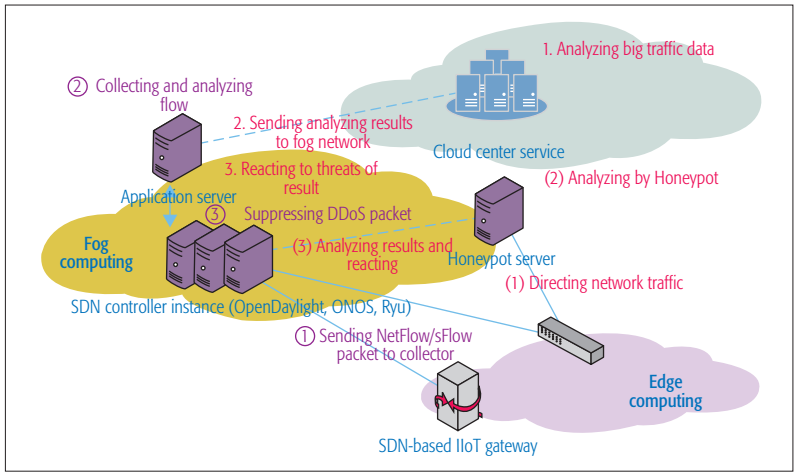

Figure 4. Defending DDoS function at the fog computing level.

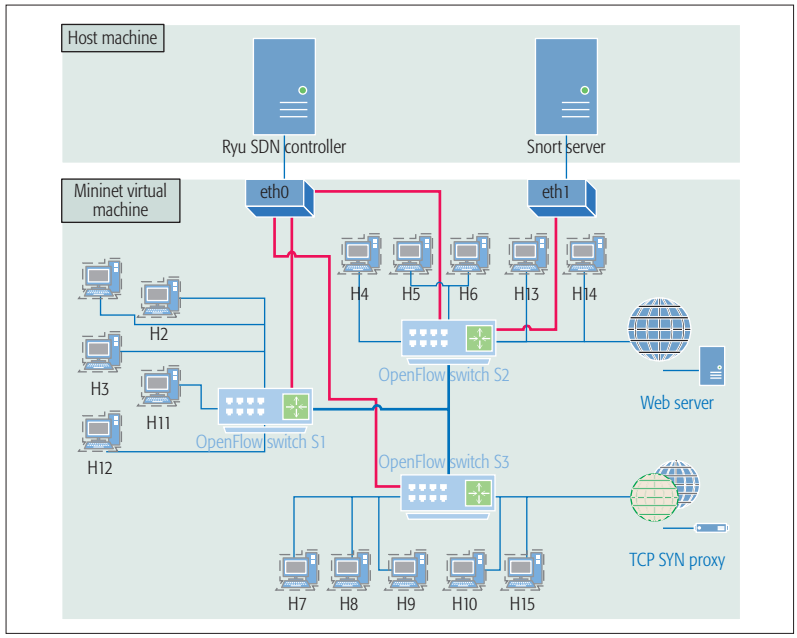

Figure 5. Topology of the experiments.

view that detecting DDoS attacks is much easier near the victim, and stopping DDoS attacks is much simpler near the attacker. Thus, the edge, fog, and cloud computing levels can cooperate to defend against DDoS attacks.

The cloud computing level has powerful data analysis capability and is an aggregation point of all DDoS attack traffic. The cloud computing level can use big data and intelligent computing to detect DDoS attacks and then send back the information to the fog computing level. The fog computing level can stop the attack flows by itself or control an SDNIGW at the edge computing level to drop the attack packets.

Cloud Computing Level

Cloud computing is suitable for IIoT applications due to its advantages of high computing power, cheap cost of services, and high scalability. On one hand, IIoT is generating an unprecedented amount of data. On the other hand, cloud computing has enough computing resources. Thus, at the cloud computing level, big data and intelligent computing can be used to detect DDoS attacks. Big data is a term for datasets that are too large for traditional database software to obtain, store, manage, and analyze. Big data is characterized by specific attributes, which are called 4V in the big data community: volume, variety, velocity, and veracity. A new way must be found to deal with these tremendous and complex datasets. Big data technology, such as Apache Hadoop and Spark, is the new data processing pattern (i.e., distributed processing structure) to handle volumes of data. Intelligent computing is an empirical computer program and is a branch of the artificial intelligence system. In a broad sense, artificial intelligence, deep neural networks, and machine learning are parts of Intelligent computing. Machine learning and deep neural networks conduce the portrayal of features of data, which is helpful to DDoS detection. The DDoS detection and mitigation system framework (DDMF) with Spark in SDN in our previous work can be applied at the cloud computing level. All kinds of intelligent computing algorithms such as neural networks and deep learning can be used at this level to detect DDoS attacks. Combining intelligent computing and big data technology, MLDMF is promising to react to DDoS in a reasonable time or even proactively.

Experimental Results and Discussions

In this section, experimental results are presented to demonstrate that using SDN to manage IIoT can protect IIoT from botnets and DDoS attacks. The following experiments use ping of death and TCP SYN flood to attack a server. The proposed MLDMF can relieve the effect of DDoS attacks by using SDN to manage IIoT.

The topology of the experiment is shown in Fig. 5. The host computer's CPU is Intel i5 4570 3.2 GHz and 8 GB DDR3 1600MHZ memory. The mininet is a virtual machine with one core CPU and 1 GB memory. In the topology, hosts 1 to 10 are used to pretend to be malicious users. They will send fake packets to the server, which want to disrupt the server's service. The malicious packets will use fake IP to construct packets and are injected to the network from the 10th second. Hosts 11 to 15 are used as normal users. The normal hosts will execute the command of ping or curl to connect to the server. When the previous command finishes, the normal hosts then execute another ping or curl. There will be 60 events of ping or curl to access the server and test their time-delay. The web server is a simple http server that is used to return simple web contents. The TCP SYN flood proxy is used to receive a SYN packet and then send a SYN ACK packet to the client. When the client sends another ACK to the server, the TCP proxy will ask the SDN controller to set a route path to the web server for the web client.

All the network traffic will be mirrored to snort detect the server. Snort uses detect rules to detect ping of death DDoS and TCP syn flood DDoS. Snort notifies the SDN controller to mitigate DDoS attacks. The strategy of defending against ping of death is to block the remaining packets of the ping of death attack, and the strategy of defending TCP SYN flood

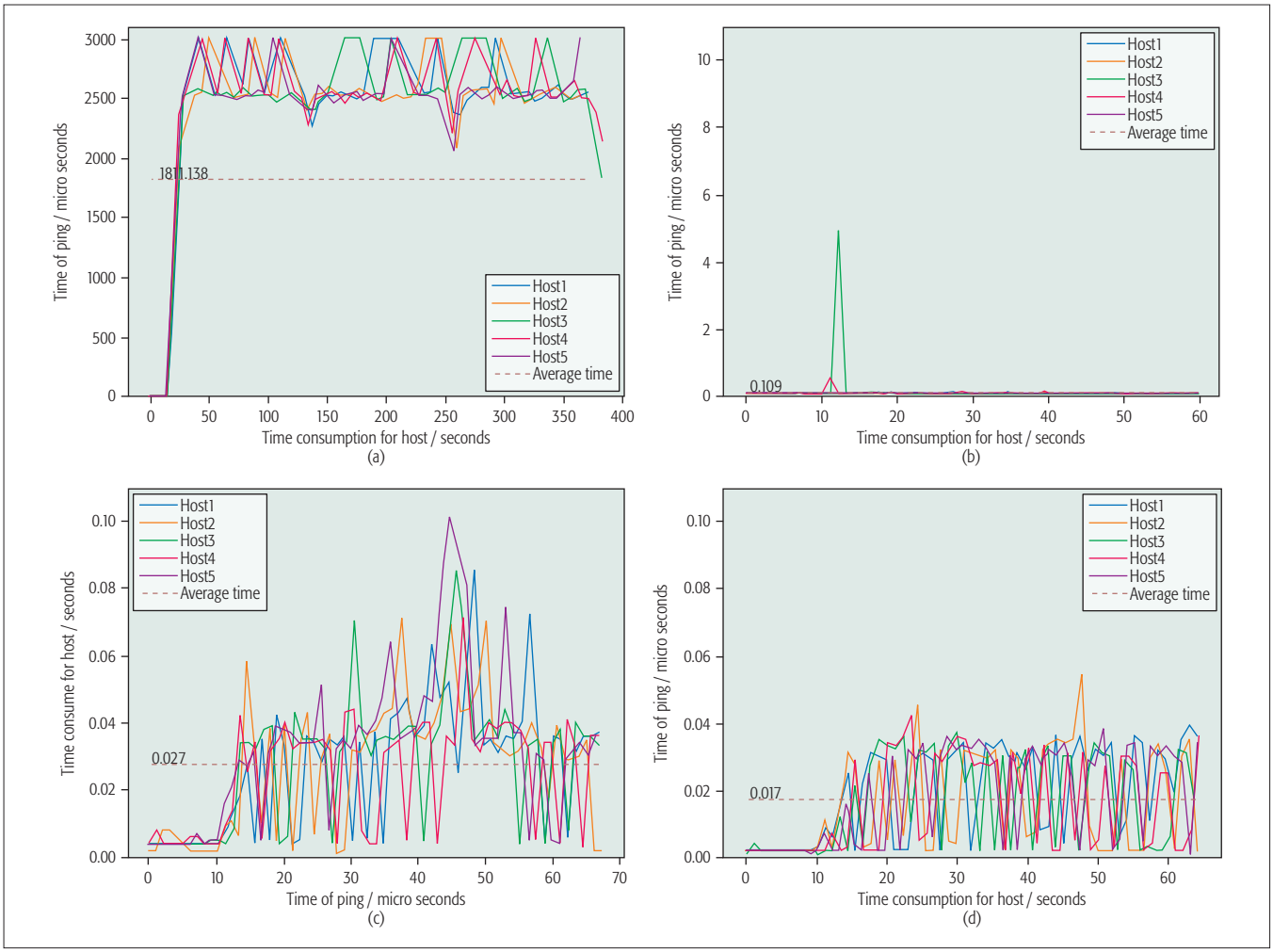

Figure 6. Experimental results.

is to use the TCP SYN proxy to select a normal user and then set a path to a web server for a web client.

The experiment results are shown in Fig. 6. Figure 6a shows the time-delay of normal users in the context of no defense methods being used to mitigate the ping of death attack, while Fig. 6b shows the time-delay of normal users in the context of using SDN to mitigate the ping of death attack. The solid lines labeled host1, host2, host3, host4, and host5 mean the time of delay of ping. The dashed line means the average time of all five hosts.

In the context of no defense, the time-delays of normal users ascend from the 10th second, which is the second the malicious network traffic is injected into the the network. The 3000 ms time-delay is used to represent the situation of ping failure. In the context of using SDN to defend against ping of death, the time-delay of normal users remains under 1 ms, except the time-delay of host 1 in the 11th second.

Figures 6c and 6d show the time-delay of normal users in the TCP SYN flood attack. Figure 6c shows the time-delay of a normal user in the context of no defense methods being used to mitigate the TCP SYN flood attack. Figure 6d shows the time-delay of a normal user in the context of using SDN to mitigate the TCP SYN flood attack. The average of time-delay in Fig. 6c is 0.27, and the corresponding average of time-delay in Fig. 6d is 0.17. The performance improvement is about 37.03 percent. Besides, in the context of using SDN, there is no occurrence of high time-delay of 0.10.

Conclusion and Future Work

In this article, we propose a multi-level DDoS mitigation framework for IIoT that includes an edge computing level, a fog computing level, and a cloud computing level. The edge computing level uses SDN-based IIoT gateways to manage and protect IIoT perception nodes. The fog computing level mainly consists of an IIoT management control unit. The IMCU uses a cluster of SDN controllers and applications to detect and counteract DDoS attacks. The cloud computing level employs big data and intelligent computing to analyze network traffic, which forms an intelligent attack detection and mitigation framework to defend against DDoS attacks. Simulation results are presented to show that a combination of edge computing's quick response ability, fog computing's state awareness ability, cloud computing's powerful computing capability, and SDN's networking programmability is promising to solve the DDoS attack problem in IIoT. Future work is in progress to consider blockchain technologies in the proposed framework.

References

[1] J. Lloret *et al.*, "An Integrated IoT Architecture for Smart Metering," *IEEE Commun. Mag.*, vol. 54, no. 12, Dec. 2016, pp. 50–57.

[2] M. L. Martn, A. Snchez-Esguevillas, and B. Carro, "Review of Methods to Predict Connectivity of Iot Wireless Devices," *Ad Hoc and Sensor Wireless Networks*, vol. 38, no. 1–4, 2017, pp. 125–41.

[3] O. Kaiwartya *et al.*, "Virtualization in Wireless Sensor Networks: Fault Tolerant Embedding for Internet of Things," *IEEE Internet of Things J.*, 2017, pp. 1–1.

[4] Q. Li *et al.*, "Game Theoretic Framework for Energy Cooperation in Wireless Sensor Networks with Energy Harvesting and Wireless Power Transfer," *Ad Hoc and Sensor Wireless Networks*, vol. 36, no. 1–4, 2017, pp. 233–56.

[5] J. Li *et al.*, "Industrial Internet: A Survey on the Enabling Technologies, Applications, and Challenges," *IEEE Commun. Surveys & Tutorials*, vol. 19, no. 3, 2017, pp. 1504–26.

[6] S. T. Zargar, J. Joshi, and D. Tipper, "A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks," *IEEE Commun. Surveys & Tutorials*, vol. 15, no. 4, 2013, pp. 2046–69.

[7] Q. Yan *et al.*, "Software-Defined Networking (SDN) and Distributed Denial of Service (DDoS) Attacks in Cloud Computing Environments: A Survey, Some Research Issues, and Challenges," *IEEE Commun. Surveys & Tutorials*, vol. 18, no. 1, 2016, pp. 602–22.

[8] M. Tellez, S. El-Tawab, and M. H. Heydari, "IoT Security Attacks Using Reverse Engineering Methods on WSN Applications," *Proc. IEEE World Forum on Internet of Things*, 2016, pp. 182–87.

[9] L. A. B. Pacheco *et al.*, "Evaluation of Distributed Denial of Service Threat in the Internet of Things," *Proc. IEEE 15th Int'l Symp. Network Computing and Applications*, Oct. 2016, pp. 89–92.

[10] W. Trappe, R. Howard, and R. S. Moore, "Low-Energy Security: Limits and Opportunities in the Internet of Things," *IEEE Security & Privacy*, vol. 13, no. 1, 2015, pp. 14–21.

[11] Yin Minn *et al.*, "IoTPOT: A Novel Honeypot for Revealing Current IoT Threats," *J. Info. Processing*, vol. 24, no. 3, 2016, pp. 522–33.

[12] X. Masip-Bruin *et al.*, "Foggy Clouds and Cloudy Fogs: A Real Need for Coordinated Management of Fog-to-Cloud Computing Systems," *IEEE Wireless Commun.*, vol. 23, no. 5, Oct. 2016, pp. 120–28.

[13] C. Alippi *et al.*, "A Cloud to the Ground: The New Frontier of Intelligent and Autonomous Networks of Things," *IEEE Commun. Mag.*, vol. 54, no. 12, Dec. 2016, pp. 14–20.

[14] H. Zhang *et al.*, "Computing Resource Allocation in Three-Tier IoT Fog Networks: A Joint Optimization Approach Combining Stackelberg Game and Matching," *IEEE Internet of Things J.*, vol. 4, no. 5, Oct. 2017, pp. 1204–15.

[15] J. Wan *et al.*, "Software-Defined industrial Internet of Things in the Context of Industry 4.0," *IEEE Sensors J.*, vol. 16, no. 20, 2016, pp. 7373–80.

Biographies

Qiao Yan (yanq@szu.edu.cn) is a professor at the College of Computer Science and Software Engineering, Shenzhen University, China. She received her Ph.D. degree in information and communication engineering from Xidian University, Xi,'an, China, in 2003. From 2004 to 2005 she worked at Tsinghua University, Beijing, China, as a postdoctoral researcher. From 2013 to 2014 she worked at Carleton University, Ottawa, Canada, as a visiting scholar. Her research interests are network security, cloud computing, and software-defined networking.

Wenyao Huang is a Master's student in the College of Computer Science and Software Engineering, Shenzhen University. His research interests include DDoS and software defined networking (SDN).

Xupeng Luo is a Master's student in the College of Computer Science and Software Engineering,Shenzhen University. His research interests include DDoS and software defined networking.

Qingxiang Gong is a Master's student in the College of Computer Science and Software Engineering, Shenzhen University. His research interests include DDoS and software defined networking.

F. Richard Yu [S'00, M'04, SM'08] (richard.yu@carleton.ca) is a professor at Carleton University. His research interests include connected vehicles, security, and green ICT. He serves on the Editorial Boards of several journals, including Co-Editor-in-Chief of *Ad Hoc & Sensor Wireless Networks*, a Lead Series Editor of *IEEE Transactions on Vehicular Technology*, and *IEEE Transactions on Green Communications and Networking* and IEEE Communications Surveys & Tutorials. He is a Distinguished Lecturer and a member of the Board of Governors of the IEEE Vehicular Technology Society.