International Journal of Computer Engineering and Information Technology

VOL. 9, NO. 6, June 2017, 108–116
Available online at: www.ijceit.org
E-ISSN 2412-8856 (Online)

Architectures and Security of Software Defined Networks for Internet of Things: State-of-the-Art and Challenges

Mohammed S. Al-kahtani[1] and Lutful Karim[2]

[1] Dept. of Computer Engineering, Prince Sattam bin Abdulaziz University, Saudi Arabia

[2] School of ICT, Seneca College of Applied Arts and Technology, Toronto, Canada

[1]alkahtani@psau.edu.sa

ABSTRACT

Internet of Things (IoT) connects thousands of everyday use objects and devices under the same network. Hence, Software Defined Networking (SDN) is crucial in evolving IoT. SDN provides a programmable and adaptive networking through the use of dedicated central controller that can handle a multitude of different connected devices. Integrating SDN into IoT can enable intelligent routing, simplified data acquisition, transition, and analysis, centralized management of network resources and applications, and dynamic on-demand reconfiguration of the network. At the same time, they pose challenges in terms of performance, interoperability, scalability, reliability and security. In this paper, we present a comprehensive survey on SDN for IoT, i.e., the concepts of SDN-IoT and their impact on each other, several SDN-IoT architectures, security and privacy implications, and other challenges.

Keywords: *SDN, IoT, Device to Device Communication, IoT Security and Privacy, Threat model.*

1. INTRODUCTION

The ever changing connectivity in the world today has resulted in an explosive growth of the number of devices that are connected through a vast network - more specifically the Internet. This network is defined as Internet of Things (IoT), which allows billions of objects or devices ranging from chairs, bridges, cameras, vehicles, to animals or even people (embedded with RFID) to be connected on a network level. Thus, IoT is a network of objects, animals or people with a unique identifier of each object so that objects can transfer data over a wireless or wired network without human-to-human or human-to-computer interaction. IoT is experiencing a tremendous growth in these days and is expected to reach 50 billion objects by 2020 [1]. International Data Corporation (IDC) also predicts that IoT will account for 10% of all data on the planet by 2020.

The enormous amount of devices that are expected to connect and generate traffic over the Internet and the mobile networks bring many new challenges and a need for changes in the existing network operation in order to properly support the new features and the additional load [22]. A few of the challenges for the successful operation of IoT include network management, interoperability, performance, and flexibility. To address these challenges the network must support heterogeneity, i.e., they should support various network connectivity capabilities such as Wi-Fi, RFID, Sensor, WiMAX, LTE, BLE, NFC and other wireless communications over different infrastructure and standards such as 4G and future 5G. However, the network heterogeneity along service requirements such as high level of scalability and high volume of traffic and mobility of the IoT devices pose a great challenge. Thus, the researchers and industry leaders require re-thinking of the existing network solutions and development of appropriate infrastructure that can cope with the requirements that IoT demands in an efficient way. More specifically, existing LTE communication networks do not seem to be appropriately able to cope with the load and the emerging needs of IoT devices as well as the cloud computing [2]. However, cloud computing is an integral part of IoT and hence, mobile communication industries are greatly interested in being able to support these technologies.

The solutions to address above mentioned challenge are either (i) mold new IoT objects to be compatible with the existing protocols and communication mediums, or (ii) design completely new devices. One of the leading solutions for the IoT challenges is the use of Software Defined Networking (SDN) – a new concept that decouples network services into control plane (makes decision on traffic flow) and data plane (forward traffic to

desired destination) and then connects the planes via an open programmable interface. Thus, SDN allows utilization of both approaches as mentioned above. The SDN supports centralization, abstraction and flexibility that are essential to support the wide IoT presence in the network. Hence, SDN is referred as the "most promising solution for future Internet" [3]. There are also models of SDN that achieve convergence of separate architectures used by IoT devices (WiFi-4G-LTE) [3].

Traditional network approaches are based on manual configuration of the network devices which are believed to be error prone and do not fully utilize the physical network infrastructure. The SDN approach enables network evolution by simplifying network management and enabling innovation. In SDN, networking devices act as mere packet forwarding devices and are programmed through open interface such as OpenFlow. Hence, it separates the control plane from data plane and provides programmability for network application development.

However, SDN and IoT are also vulnerable to a considerable amount of security threats due to their characteristics such as heterogeneity and sheer size. Network devices can be overloaded and disabled by high volume traffic. These devices can be compromised and turned into bots [8]. The SDN controller acts as an intermediary between application program interfaces (APIs) and network devices. Hence, the placement and number of SDN controllers as well as how efficiently it manages shared resources across applications is crucial to the scalability and reliability of SDN. Interoperability is essential when dealing with heterogeneous devices, different data formats, and various protocols for device-to-device communications in different communication domains [11]. Another major threat is the problem of authentication and authorization among a myriad of heterogeneous devices and applications that operate in different domains.

This paper presents a comprehensive survey on SDN in IoT. More specifically, the paper presents (i) general SDN architecture (ii) SDN architectures that are specifically adjusted to accommodate IoT (These SDN architecture differs from the SDN used in data centers). Furthermore, SDN for IoT implementation will be examined from the perspective of security to analyze the risks associated with different vulnerabilities and threats in existing frameworks. The risks will be classified and compared along with the defensive mechanisms. A threat model for SDN for IoT is also presented and discussed. The rest of the paper is organized as follows.

Section 2 defines some important terminologies related to IoT and SDN that are used in this paper. Section 3 presents several SDN architectures in literature that are suitable for general and IoT applications. Then, we also present security and privacy aspects, challenges of SDN-IoT frameworks in Section 4. Finally, Section 5 concludes the papers with discussion on challenges (security and others) and prospective solutions.

2. BACKGROUND

Internet of Things (IoT) devices can be applied to almost any kind of network. In the future, IoT will be present in every home, neighborhoods, on the road, in the office and on people. Unfortunately, existing network devices and typical network topologies in each scenario will unlikely support the new wave of IoT devices. How will these devices be connected using both wired and wireless medium? Many IoT devices in our home may be wired. Programmable doors, lighting fixtures, heating sensors for water and HVAC systems, or ID verification through biometric hand print readers will be connected through a centralized processing machine in home and further be relayed to control servers through a number of other networking devices. This wired connection poses problems such as routing protocols: will the devices have easy plug and play characteristics? Or will they need manual configuration and require expertise? For wireless sensor nodes, routing protocols are the bigger concern. Will they provide adequate performance while maintaining energy and processing efficiency? The underlying catch all question is, will these new implementations be secure or will they open up new vulnerabilities? One solution to all these challenges is integrating software defined network (SDN) into IoT frameworks that also result in other research challenges: what would be the optimal IoT-SDN architecture, how to design the threat model for IoT-SDN by successfully identifying threats and vulnerabilities in IoT-SDN and also what would be defensive mechanisms for the security/privacy concerns against IoT-SDN. Discussion and analysis on these challenges are presented in the following sections.

3. ARCHITECTURE

This section presents architecture of software defined networks (SDN) for Internet of Things (IoT).

3.1 General SDN Architecture

Software defined networks (SDNs) enable implementation of high-level network policies and a global view of the network as a whole which in turn improves understanding and management capabilities of the network. This is not possible to achieve in the traditional network architectures.

The evolution of the SDN can be divided into three stages: (i) active networks, (ii) separation of data and

control planes, and (iii) the OpenFlow API and NOS [4]. The exponential growth of network traffic requires improving network management process and capabilities such as management of paths circulating the network, traffic prediction and identification of network failure and fast failure recovery. The scope of such network management functions (e.g., calculating paths, predicting traffic, identifying and recovery of network failure) are limited using traditional networks as software and hardware of traditional networks are closely connected. While packet forwarding process is focused on hardware, the control of the network management can be best done by software applications that are installed and run on a server with higher resources (processing speeds and memory) compared to a network node. Forwarding and Control Element Separation (ForCES) proposed by the IETF had the control element separated from the forwarding element and thus, redefined the internal architecture of the network device. However, some other architecture proposed in [3] did not gain large popularity.

A well accepted protocol for SDN is OpenFlow while Open Networking Foundation is responsible for its publication. OpenFlow uses hardware features that are available in the current offered network devices. This makes it acceptable across the industry. OpenFlow enables external control of the functions of common network devices. These functions include reading the header, sending packets to a port and dropping a packet. Another important characteristic is that an upgrade of firmware on the network device may suffice to enable OpenFlow support on the device without the need to make any hardware changes. There are (i) hybrid switches (or OpenFlow-enabled) that support both the traditional network approach and OpenFlow protocol and (ii) OpenFlow-only switches that support OpenFlow only.

In OpenFlow enabled architecture, the packet forwarding device contains flow tables and an abstraction layer that enables secure communication with the controller by using the OpenFlow protocol. The flow tables contain flow entries that set the way that the arrived packet will be processed and forwarded. Flow entries include match field (information from packet header, ingress port and metadata), counter field (to collect statistics on the flow) and set of instructions that are to be applied when a packet is matched. There is also a mandatory table-miss flow entry that rules what to do with packets that did not match any of the existing match fields. The number of fields that can be processed by a switch depends on the OpenFlow version: v1.0 supports 12 fields while v1.3 supports 40 fields including support of IPv6 which is crucial for implementation of IoT devices.

The controller received information from multiple switches and configures the switch flow tables (remotely). This enables the user to program the network from a centralized place. The controller hosts a Network Operating System (NOS). It is software that abstracts the installation of the state in switches of the logic that controls the network behavior [4]. Based on the same concept as host operating systems, NOS allows creation of applications using high level abstraction of resources and hardware. The abstraction of network resources are classified as southbound and northbound, while former refers to functionality of the switch and its connection to the controller. The later refers to creation of high level network policies by applications and their transmission to NOS. Examples of NOS include NOX (C++ based), POX (Python written), BEACON (Java-based) and others. Traditional and SDN network architecture is illustrated in Figure 1 and Figure 2.

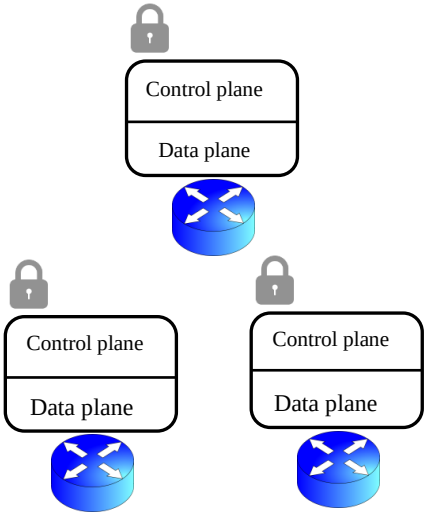

Fig. 1. Traditional networks

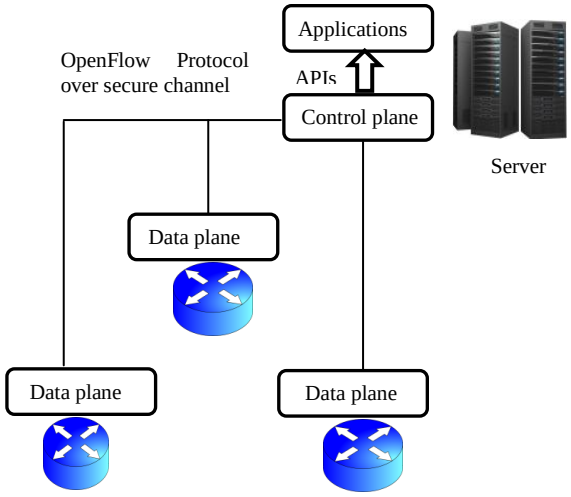

Fig. 2. Software Defined Network (SDN)

According to [5] it is believed that cloud platforms will become a de-facto standard for IoT applications. As IoT are expected to be a significant source of big data generation, it is natural to think that cloud computing should be the choice for IoT. The central processing units of IoT would be comprised of server farms that can analyze real time data. These servers are to be hosted in cloud computing and the NFV component will greatly improve the efficiency. IoT are characterized with multiple state changes (going up and moving to hibernate or sleep mode later based on when the information should be collected)

3.2 IoT-SDN Controller Architecture

As mentioned earlier, the SDN architecture seems to be a promising candidate for providing the resource management needs of IoT environment. It is achieved through separation of control plane from data plane which simplifies network administration and a balance between centralized control (SDN controller) and decentralized operations with flow-based routing and rescheduling. The current implementation of SDN as described in section 3.1 fails to address the heterogeneous and vibrant needs of IoT. SDN is currently implemented in distributed cloud networking (DCN) and focused on network statistics collection in data centers. In contrary, IoT multinetworks environment is usually distributed over wide area network and the performance metrics also include the collection overhead reduction and effectiveness of the total data needs. There are also additional timing related needs for IoT which are delay, jitter, packet loss and throughput. The importance of these metrics varies based on the applications of the IoT device, e.g., real-time information from an end device such as camera about road status requires low latency and reliable delivery of information while querying multiple data sources periodically for data about traffic statistics on vehicles that were charged at given recharging site require handling of significant number of updates generated in asymmetric way. These requirements demonstrate that flow scheduling that is offered in SDN for DCN does not address all of the IoT needs.

Another issue with current SDN implementation is that current protocol realization are more concentrated on the south-bound communication, between controllers and network devices while the north-bound interactions between the applications and the controller are not yet standardized [6]. It is worth mentioning that there are proposed techniques to apply SDN to wireless networks. These include OpenRadio, CellSDN for cellular applications and OpenWireless for streaming video data seamlessly between Wi-Fi and WiMAX networks [6]. The work done in [6] also proposes a new IoT multi-networks controller architecture to address the above mentioned issues. The proposed controller architecture is comprised of data collection, API, task-resource matching, solution specifications, flow scheduling and communications layer. Data collection component collects information from IoT networks and stores it in databases. This information is used by the other components while needed. The API enables analysts, admins and external processes to control processes. It is important to mention that proper security of API from unauthorized access is essential and should be addresses appropriately during implementation. In order to eliminate a single point of failure and improve scalability, the controller can be instantiated and placed in different locations.

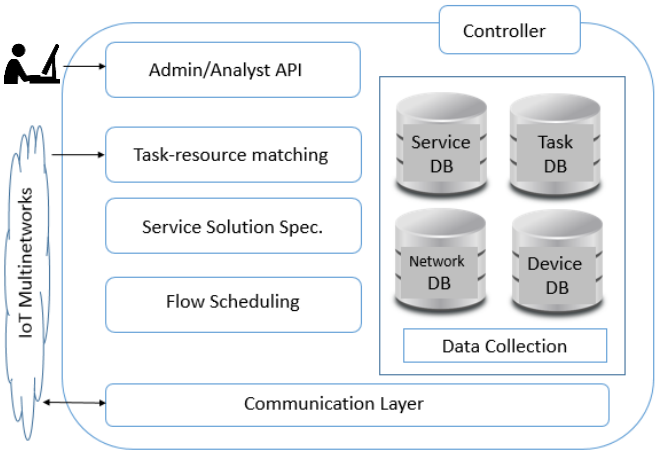

Fig. 3. IoT-SDN Controller Architecture

The abstraction level used in the described architecture is essential in order to be able to flexibly use heterogeneous multi-network resources. Figure 3 demonstrates the proposed controller architecture. The highest level of abstraction is in tasks that define what is required to perform the task. The work done in [6] provides an example of the task "locate cab 001 in I-5 freeway section 107". Given this example, task resource matching component will determine the available resources in given location for accomplishing this task. Then resources will be filtered based on whether they are capable of accomplishing the task or not. This will be accomplished by accessing the information stored in Service and Device DB in the controller. The resource solutions can then be further refined. If the solution for above example is road camera and a server for image processing, then refinement may state that the video stream from the given road camera should be sent to server and processed according to specified techniques. The service solution can be filtered by automatic controller policies or a network operator.

After choosing a solution, service solution component matches the device and the service in the solution to specific requirements and constraints of the devices and

services in use. In given example this could be video resolution or receiver's buffer. Flow scheduling module schedules flows that would match the requirements that were found earlier. Due to different QoS requirements and the large variance of networks involved, this task may be quite complex. The next step is performed by the communication layer by using the appropriate communications in IoT networks to be sent to network devices and routed along the right path. In given example it could be routing the video data from camera 001 through Ethernet.

3.3 SDN-DDS Architecture for IoT

Data Distribution Service (DDS) is a protocol for IoT that was standardized by Object Management Group (OMG). DDS allows connected machines and mobile device to interact each other. It can be deployed in both the cloud and low power devices to support efficient bandwidth usage. As described in [7] DDS provides a flexible structure with the following properties: (i) supports location by anonymous publish/subscribe (ii) provides redundancy by allowing any number of readers and writers (iii) allows asynchronous data distribution (iv) permits message-based data-centric connection management (v) supports independent platform model.

The DDS domain represents a virtual global data-space where information provided are accessible by applications that registered to this domain. In addition, DDS recognizes two areas that are important in IoT systems – discovery and meta-data. The SDN-DDS architecture in [6] is presented in Figure 4. It is based on the assumption that IoT system architecture is comprised of sensors and actuators that are connected to local processing and Internet**.** The Internet is provided through a service provider core router through terrestrial or mobile access. The router communicates with IoT gateway which supports various communication technologies (e.g., Wi-Fi, Ethernet, VPN, and ZigBee). In this architecture, IoT and the gateways use DDS middleware to publish and subscribe data. There is a DDS northbound interface in the SDN controller that enables IoT systems to support SDN. The interface exposes the functionalities of IoT network applications to controller so it can provide generalized network support for the IoT devices.

The suggested controller contains DDS middleware (or messaging layer) as a connector between IoT framework and network. The publish/subscribe capability of DDS allows anonymous, asynchronous and many-to-many communication schemes hat is significantly important for the IoT. The DDS middleware communicates with the SDN control plane to provide the following three services.

1. Packet handler of SDN control plane uses DDS notification, i.e., DR listener to read PACKET_IN events that were forwarded by SDN data plane.
2. Packet forwarder forwards packets that were received through PACKET_IN events of SDN data plane or created by IoT applications.
3. Flow programming service of DDS defines flow programming rules on the OpenFlow switches (described in section A under general SDN architecture).

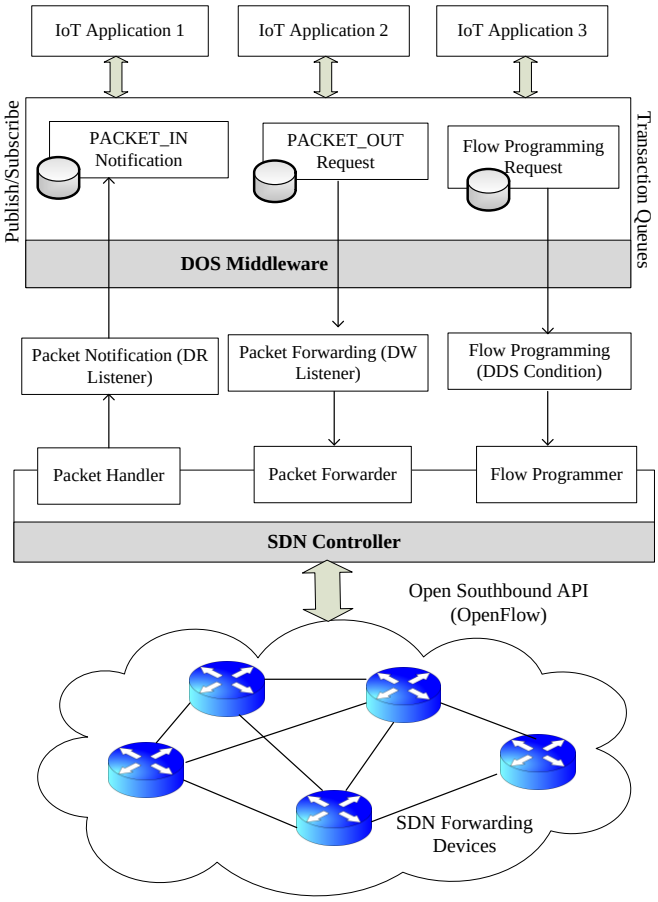

Fig. 4. SDN-DDS Architecture

3.4 SDN and NFV

Network Function Virtualization (NFV) is a complementary technology to SDN and can potentially significantly impact the future network performance by virtualizing many network functions. In other words, NFV enables dedicated network hardware devices such as routers, firewall and load balancers to be hosted on virtual machines. NFV can help to better utilize the network for the ever changing network demands of the IoT devices.

4. SECURITY AND PRIVACY CONCERNS

In this section, we present and analyze the threat model of SDN-IoT by identifying threats and vulnerabilities. We also present security challenges as well as defensive mechanisms of SDN-IoT.

4.1 Threat Modeling

Threat modeling is essentially the process of capturing and analyzing all information related to the secure operation of a particular implementation of a technology in real world scenarios. The process includes the assessment of threats (i.e., how likely is something to represent an attractive target to an attacker, what is the value of my data, what is the cost of downtime), vulnerabilities (where are defenses weakest), and the production of a model to aid in the deployment of security improvements.

Threat modeling is mainly classified into four types. The first is software-centric or system-centric. In this model, system design is evaluated first and each aspect of the system is examined and the threats against it determined. Elevation of Privilege (the card game) is an example of the system-centric model. The next is an asset-centric model. In this model, we begin by examining high value targets (what does my organization control that would be of value to an attacker). In this case, the security response is determined by the value of these targets. An attacker-centric approach is also viable. The goal of this model is to determine who likely attackers are, and what might motivate them. This approach is related to the asset-centric approach in that it is dependent of determining the assets an attacker may wish to access. Finally, a hybrid of any of the three previous models is also possible.

Another approach - Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege (STRIDE), which is used in many emerging technologies. For each STRIDE threat, we describe the sources of threat and vulnerability and how the threat action takes place [14]. We employ STRIDE threat modeling in SDNs. Denial of service comes mostly from a controller or switch, Information disclosure from an employee of an organization or devices of the network that may disclose information to an attacker and lastly tampering comes mostly from some parts of the network that changes or alters data to gain access into the network.

Thus, we identify threats and vulnerabilities in SDN to design threat modeling for SDN, which are as follow [19 - 20].

- SDN opens potential security holes with regards to connections between controllers and network elements, through which the SDN stack itself might be the subject of a distributed denial of service attack [19].
- The configuration errors in SDN can have more severe consequences than those in traditional networks.
- The centralized architecture of SDN could allow attackers to take the control of the entire network.
- Open APIs with appropriate security features have not been implemented and standardized for SDN. Hence, the API incompatibilities may cause security flaws [19].
- As SDN provides control via well-documented, easy-to-navigate APIs [20] even in proprietary vendor implementations, .SDN worms do not require managing hundreds of different attack vectors.
- If an attacker can compromise the SDN control layer, they will be able to hide their activities from monitoring and management subscribers.

Table 1 lists the possible threat sources in SDN-IOT with detail description [14]. Moreover, we classify the sources of vulnerabilities in SDN, which is presented in Table 2 [14]. The STRIDE approach is used to construct an OpenFlow threat model that identifies the potential vulnerabilities. Then, attack trees are used to explore how an attacker could exploit vulnerabilities. The attacks on OpenFlow and their countermeasures are presented as follow [21].

- Denial of Service (DoS) is a severe attack against flow table. In DoS, attackers generate a large number of packets and send to the controller. Hence, the controller implements/installs a new flow rule for each packet that overloads the flow table. The DoS attacks can be eliminated by limiting data transmission rate, filtering events and packets, dropping packets, aggregating flows, detecting attacks and access control [21].
- Hash collision attack on the flow table or data structure in the controller is another attack.
- Attackers exploit flow aggregation to detect network phenomenon. For instance, attackers observe the differences in controller response time to derive information about network state or active flow rules. These types of attacks can be eliminated by randomizing the observable system parameters so that attackers cannot expose the internal system state [21].

Table 1: The threat sources affecting SDN-IoT

Threat Source	Definition
Non SDN-IoT system	A system that is not within the SDN-IoT framework
Rogue SDN-IoT system	An unauthorized system within the SDN-IoT architecture
Malicious SDN applications	An application engaging in malicious activities or a malicious user using it
Malicious SDN controller	A controller engaging in malicious activities or a malicious user using it

Malicious network device	A network device engaging in malicious activities or a malicious user using it
Malicious IoT device	An IoT device engaging in malicious activities or a malicious user using it
Malicious management console	A management console that is engaged in malicious activities or a management console that is used by malicious users

Table 2: The sources of vulnerabilities in SDN-IoT framework

Source of Vulnerability	**Description**
Application	An application accessing the resources provided by the SDN controllers
SDN controller	A machine that controls network devices
Network device	Devices in charge of traffic forwarding
Management console	A console for applications, controllers, and network devices; supports remote management tasks
Northbound interface	Communication channel between applications and the SDN controller
Southbound interface	Communication channel between the SDN controller and network devices
East/west interface	Communication channel between distributed SDN controllers
Management interface	Communication channel between the management console and applications, controllers and network devices in each plane

Table 3: The threats which violate the security

Threat	**Description**	**Security violation**
Spoofing	Impersonating something or someone's identity	Authentication
Tampering	Changing something	Integrity
Repudiation	Denying what you did or claiming you did not do something	Non-Repudiation
Information Disclosure	Unauthorized person accesses information	Confidentiality
Denial of Service	Making resources unavailable to legitimate users	Availability
Elevation of Privilege	Allowing someone to perform unauthorized tasks	Authorization

Table 3 lists the threats based on the STRIDE model along with the type of security that it violates [15]. Table 4 presents a list of threats that works in SDN-IoT architecture with detailed description [14 - 15].

Table 4: The threat actions in the SDN-IoT

Threat Action	**Details**
Spoofing	Unauthorized access gained by impersonating legitimate SDN-IoT element
Information Disclosure	Unauthorized disclosure of information resulted by a threat source obtaining information not intended for it
Repudiation	A threat source claiming to have not performed an action or have been a victim or manipulating the logs
Tampering	Unauthorized modification or destruction of data being acquired, transmitted or analyzed
Denial of Service	Service disruption caused by a threat source disrupting a SDN-IoT element
Elevation of Privilege	Unauthorized access gained by a threat source to a SDN-IoT element for which it does not possess the access clearance

4.2 Security Challenges and Discussions

In this section, we present security challenges of SDN-IoT, which are related to the following sources.

4.2.1 Authentication and Authorization

Authenticating IoT devices and applications is a challenge in SDN-IoT environment for a number of reasons such as (i) SDN allows multi-tenancy and allows Greenfield communication, i.e., allows everyone to access into the network and reprogram the network according to needs, (ii) the SDN controllers are dynamically assigned to switches all the time and switches are handed over from one controller to another (iii) different devices join and leave the network frequently and (iv) different SDN-IoT applications share the same infrastructure [8] [12].
According to [12], the key methods for authentication and authorization are access control and cryptography, but frequent handover using these methods induces latency which can lower the efficiency of the network. One solution for implementing faster, efficient and robust authentication and authorization is utilizing user-specific attributes as a shared non-cryptographic security context. The controller, which is a program running in a data center could be equipped with an authentication handover module that is in charge of authentication and handover.

4.2.2 Data Confidentiality for Privacy Protection

IoT applications require data confidentiality and reliable authentication as they acquire, transmit and process data from variety of sources which cause the increase in threat vectors [13]. Also, since cloud provides data mining platforms, visualization software, and virtual machines for storage, there is always an opportunity for an attacker to attempt to acquire and analyze big data collected from IoT, thereby threatening the privacy of data.
The work done in [12] proposes a solution in which the controller transmits pieces of data stream via multiple network paths and only the receiver can decrypt the data using its primary key then reorganize the data stream. This method can avoid privacy leakage by; the privacy level, in conjunction with other elements such as system complexity.

4.2.3 Threat Detection

Liu et al. [10] point out that one drawback of SDN of today is that it is unable to deep-inspect every packet. OpenFlow is limited in that the match fields are applied to the packet headers only and that it cannot look into the

data portion of the packet to determine suspicious flow. In order to detect malicious devices which utilize decentralized entry points among IoT as well as SDN, effective and strong detection mechanisms are needed. The controller program capable of deep packet inspection and malware detection could be implemented, but it should not cause undue overhead.

4.3 Other Challenges

Network Management for Reliability and Scalability in SDN-IoT pose a great challenge. The problem of scalability consists of the controller scalability and network device scalability facing latency raised by data transmission between: a single controller and many network devices; and the controllers and other controllers. Latency and overhead in the SDN network caused by flow handing capacity and bandwidth between switches and controller must be resolved for reliable performance and scalability of the network. The work done in [8] states that the main resources consumed in the SDN-IoT are: the link bandwidth; the controller's computational power to handle the flow; and the switch's capability to handle the flow. Therefore, in order to adequately provision for these resources, you must deal with the type and length of flow, network topology,

If the link bandwidth between the switches and controllers are not well provisioned or overused, with the increasing flow size, it may result in the increase of collision rate, jitters and delays in the network. In [10], various reliability and scalability issues are raised: conflict resolution and optimization for multiple applications that share the same platform must be worked out; traffic scheduling should be QoS enabled; resource mapping in cloud needs to delegate application service requests to the physical device, therefore needs to determine where to store data and which servers are to be used for data processing; and how to design the control layer with objectives such as scalability, performance and robustness.

Distribution of multiple controllers is important for replication, scalability, lower rate of delay and preventing a single point of failure. Since logically centralized controller is to be physically distributed, we must consider the appropriate number of controllers and their placement [3].

The limited flow handling table capacity, if pushed, may result in packets dropped while setting up new rules due to dynamic change in the applications [8]. Or, Denial of Service attacks can be launched against the flow table as high volume traffic can consume the table capacity. Secure flow automation needs central management of data forwarding.

5. DISCUSSION AND CONCLUSION

In this paper, we present several architectures of software defined networks in Internet of Things (SDN-IoT) and identify challenges in designing SDN in IoT. Heterogeneity, scalability, interoperability, designing efficient routing protocols, security and privacy pose a great challenge in SDN-IoT. The security challenges in SDN-IoT were also analyzed along with threat modeling (by identifying threats and vulnerabilities). It is worth mentioning that the security challenges in SDN-IoT can be handled implementing the following strategies: proper trust relationship management, access policies enforcement in real-time based on the network and device behavior, dynamic traffic rerouting and network reconfiguration, application of cryptography, greater network intelligence and data analytics and sharing capacity, fault tolerance, auto system restoration and mitigation to keep up with real-time requirements [13].

Despite many challenges, the SDN has advantages in mitigating security challenges in IoT. Since the data plane is decoupled from the control plane, should the data plane be breached, attackers are unable to use the data from which the controls have been removed. The SDN controller can adapt automatically to the changed environment when security breaches happen and also quarantine malicious actors. Dynamic service provisioning can ensure load balancing and QoS. The programmability of the SDN controller should be able to open doors for further development in threat detection mechanisms and deep packet inspection. Hence, IoT is expected to grow rapidly with the contribution of SDN to its wide adaptation.

REFERENCES

[1] D. Evans. "The Internet of Things. How the next evolution of the internet is changing everything". Cisco IBSG San Jose, CA. 2011

[2] H. Wang, S Chen, H Xu et al. "SoftNet: a software defined decentralized mobile network architecture toward 5G". IEEE Network, Vol 29. P 16-22. 2015.

[3] B. Nunes, M Mendonca, X. Nguyen,J et al. "A survey of software-defined networking: past, present and future of programmable networks," in IEEE Communications & Tutorials, vol. 16, no. 3, pp. 1617-1634. 2014.

[4] A. Caraguay, A Peral, L. Lopez et al. "SDN: Evolution and opportunities in the development IoT applications". University of Madrid, Madrid, Spain. 2014.

[5] S. Nigam "The perfect storm", in Software Defined Planet with Internet of Things. Hopkinton, MS, EMC Corp. 2015.

[6] A. Qin, G. Denker, C. Giannelli et al. "A software defined networking architecture for the internet-of-things". IEEE Network Operations and Management Symposium (NOMS). P 1-9. 2014.

[7] A. Hakiri, P. Berthou, A. Gokhale et al. "Publish/subscribe enabled software defined networking for efficient and scalable IoT Communications". IEEE Communications Magazine. Vol 53. P 48-54. 2015.
[8] K. Sood, S. Yu, Y. Xiang, "Sofware Defined Wireless Netorking Opportunities and Challenge for Internet of Things: A Review," in IEEE Internet of Things Journal, pp. 1-11, 2015.
[9] M-K. Shin, Y. Hong. "A software defined approach for end-to-end IoT Networking". SDNRG meeting. IETF. Honolulu, HW
[10] J. Liu, Y Li, M Checn. "Software defined internet of things for smart urban sensing," in IEEE Communications Magazine, vol 53, no. 9, pp. 55-63, 2015.
[11] Z. Qin, G. Denker, C. Giannelli, P. Bellavista, N. Venkatasubramanian, "A Software Defined Networking Architecture for the Internet-of-Things," in IEEE Network Operations and Management Symposium, 2014.
[12] X. Duan and X. Wang, "Authentication Handover and Privacy Protection in 5G HetNetsUsing Software-Defined Networking," in IEEE Communications Magazine, vol. 53, no. 4, pp. 28-35, 2015.
[13] J. Oltsik, "The Internet of Things: A CISO and Network Security Perspective," pp. 1-9, 2014.
[14] J. Hizver, "Taxonomic Modeling of Security Threatsin Software Defined Networking," in BlackHat Conference, 2015.
[15] A. Shostack, Threat Modelng: Designing for Security, 1st ed, Indianapolis, IN: Wiley, 2014.
[16] Routing Protocol Definition. Routing Protocol Definition by the Linux Information Project LINFO, Nov 2015, http://www.linfo.org/routing_protocol.html
[17] Network Switch. Wikipedia. Wikimedia Foundation, accessed on: 26 Nov, 2015.
[18] Margaret Rouse, "What Is SDN Controller (Software-defined Networking Controller)? – Definition from WhatIs.com", SearchSDN. TechTarget, Nov 2012, Accessed Web on 26 Nov, 2015.
[19] 2014 Threats Predictions: Software Defined Networking Promises Greater Control While Increasing Security Risks, by McAfee Labs,https://blogs.mcafee.com/mcafee-labs/2014-threats-predictions-software-defined-networking-promises-greater-control-while-increasing-security-risks/, Accessed on December 29, 2015
[20] Patrick Hubbard, Will SDN pose network security vulnerabilities? It depends,http://searchsdn.techtarget.com/opinion/Will-SDN-pose-network-security-vulnerabilities-It-depends, Accessed on Dec 29, 2015
[21] Kloti, Rowan, Vasileios Kotronis, and Paul Smith. "Openflow: A securityanalysis." In Network Protocols (ICNP), 2013 21st IEEE International Conference on, pp. 1-6. IEEE, 2013
[22] Y. K. Chen, "Challenges and opportunities of internet of things," 17th Asia and South Pacific Design Automation Conference, Sydney, NSW, 2012, pp. 383-388.