Software-Defined Networking Security - A Survey

Nandini Krupa Krishnamurthy
Clemson University
nandink@clemson.edu

***Abstract*—SDN has revolutionised networking domain by introducing a new of interacting between devices and has imparted to the architecture a better control. It has improved performance, increased reliability and programability and also have given good scalability. Legacy networks are way too complex and are integrated vertically. SDN overcomes this by using decoupling. SDN is unique in the sense that it decouples data planes from the control plane. But along with all these, it also brings in its own set of security issues. Its Network Security is a major debate. This paper begins with giving a basic understanding of SDN architecture and subtly moves on explain techniques to provide security in these systems. Furthermore, it also talks about the current issues faced by SDN security and also the future challenges present in this field.**

I. Introduction

Software-Defined Networking or SDN is now one of the widely used networking technologies. Like we already know the popularity of this technology lies in the fact that the control plane is separated from the forwarding plane [1]. SDN has two basic properties decoupling of control plane and data planes and direct programmability. It is also built upon simple networking devices the forwarding plane or the data plane is controlled by the control plane by its network instructions which then forward the data packets. The entire network is managed by centralized control plane with no regard to the underlying devices [2]. This makes the mechanism simpler by treating all devices equally and Lets us first discuss briefly on the architecture of SDN before we move forward.

A. SDN Architecture

As illustrated in the Fig 1 below, SDN architecture consists of three layers the bottom-most is the infrastructure layer, the middle one is the control layer and on top lies the application layer [3].

The bottom-most is the infrastructure layer, also called data plane. Its basic functionality is to set up data path for the network. Essentially it consists of network elements whose basic job is to collect, monitor and forward local data. Due to this, it is also called as forwarding plane.

On top of this layer lies the Control layer, also called control plane. The main task of this layer is to program and control the forwarding plane. The forwarding plane delivers the data to the control plane which in turn uses this information to for network routing and function. There are some interfaces running between these two layers. These are called as south-bound interfaces and their task is to mainly carry out the communication between control and data layers. OpenFlow is the most widely used south-bound interface. These interfaces

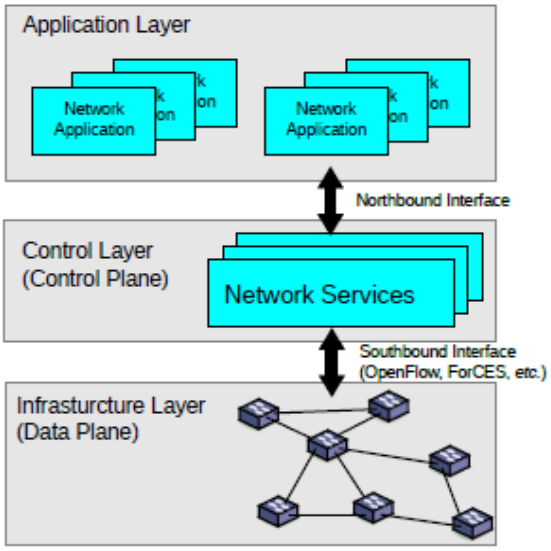

Fig. 1. SDN Architecture clearly showing the three layers and interfaces [3]

mainly consist of network components such as switches and routers.

The topmost layer is the application layer. This layer, as shown in the figure basically comprises of network application that deals with the introduction of new network features as and when necessary. These network features may include security, some networking schemes or some other feature aiding in the maintenance of the architecture.In addition to this, application layer also carries out the function of assisting the control layers in its tasks buy providing guidance. The application layer and the control layer communicate using interfaces called north-bound interface. There is no well defines API for these interfaces and control layer provides its own API to applications.

The separation of the network and data planes helped in implementing network abstraction where we just have to program the SDN network controller and all the logic will go down to individual nodes through southbound APIs and protocols. Here, controller pushes the instruction from controller and virtually control the network. All applications work on the top of the software layer but sends all the instructions down to the components. This helps in adapting to changing business requirements very fast as all individual components dont need

to be physically programmed but just the network controller. This saves a lot of time as well as abstracts the network view.

B. Advantages of using of SDN

There are various uses of using SDN. Some of them are defined as follows [4] [5] [6].

- **Central control:** SDN is centrally controlled. This means that SDN control software can control interfaces by any seller rather than differing in the management by a group of vendors.
- **OpenFlow advantage:** SDN using OpenFlow reduces automation tasks because it is flexible enough to create tools for automation of many manual tasks being done.
- **Less complexity:** As this can be automated, complexity of networks can be reduced. Operational costs may also be lessened due to this.
- **Greater performance:** SDN does not need to configure network every time a device is added or removed from the network. This increases its robustness, reliability, optimum working environment, dynamic nature and security.
- **Consistency:** SDN allows the changed to be implemented at the granular level. This helps in maintaining consistency.
- **Adaptability:** Since SDN is dynamic and adaptable to better user needs it gives a better user experience.

C. Advantages of using of SDN architecture for Security

- **Effective monitoring of abnormal traffic:** The SDN controller can handle the entire traffic simultaneously. Therefore it is easier for it to detect the abnormal behavior of traffic . It may also be easy to detect the abnormality of the traffic flow.
- **Timely dealing with vulnerabilities:** The network operators can develop new software to treat the threat without having to wait for a notice from the operating system. This considerable reduces the time spent for detecting, analyzing and treating a security threat.
- **Scalability:** SDN controller can implement policies up to 2-7 layers of OSI architecture (OSI Open Systems Interconnection) and can provide more security.

II. Main Techniques

In this section, two techniques of security control in SDN are described. They are:

- SecControl Architecture
- Policy-Based Security Architecture

A. SecControl

SecControl combines the existing security tools and SDN technologies to form a protection framework. It gathers evidence of attacks, gauges the security activities and generates a proper defense response. Fig 2 shows the SecControl architecture.

As seen in the figures, it has three layers in its architecture. These three layers are - Threat Collecting Layer, SecControl Layer, and SDN Controller Layer. Each layer has its own role in this [7]

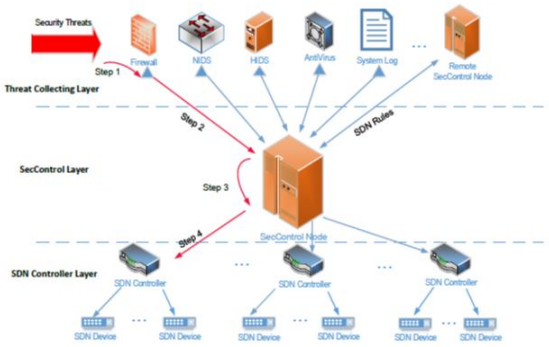

Fig. 2. The SecControl Architecture [7]

- **Threat Control Layer:** The Threat Collecting Layer as the name tells, is used to collect threats using various tools and agents. So, its basic function is to collect threats and send this information to SecControl layer.
- **SecControl Layer:** Now that SecControl layer has all this information, it runs a series of steps which include converging all the details of the security events, trying to build a relation among these, analyzing this information, and looking for evidence of these events. And, accordingly it determines a defense response .
- **SDN Control Layer:** The responses from SecControl Layer will be translated into SDN rules and sent to SDN Control Layer. Then this SDN Control Layer will enforce these rules.

1) SecControl Components: The SecControl has 4 components Treat Collecting Agent, Threat Analyzer, SDN Rule Engine, SDN rule distributer [7] as shown in the Fig 3.

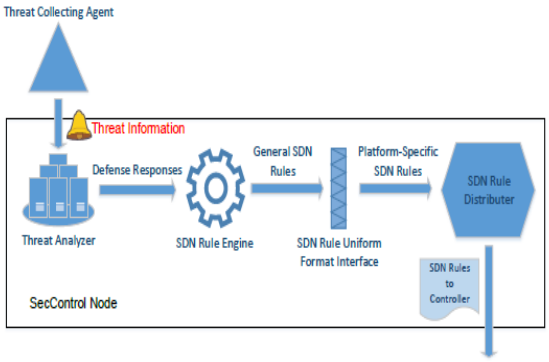

Fig. 3. The SecControl Components [7]

The functions of each of there are as explained below:

1) **Threat Collecting Agent (ThreatCA):** ThreatCA gets threat information from various security tools in various formats and gives output as the structured uniform information which can be used as the direct input to the Threat Analyzer. If existing security tools are separately analyzing various security threats, then it is preferred to have separate specialized ThreatCAs for each security tool. The main goal of a ThreatCA is to provide personalized information to Threat Analyzer. This can be done using a preprocess function. The main use of this preprocess function is to transform the obtained raw input to a to a format that is easily analyzed by the Threat Analyzer. There may be also a presence of a unified interface for the threat Analyzer for the analysis of various threat information.
2) **Threat Analyzer:** Threat Analyzer obtains the preprocessed information from ThreatCA. The main jobs of threat Analyzer is to analyses threat information, gauge the security state and decide on the defense mechanisms to encounter these threats. It is scalable and adaptable. There are various algorithms also for all the analysis Threat analyzer does. Once a threat is identified, the analyzer will search for a preexisting defense mechanism as a response for the threat. For various protection scenarios, the defense responses may be different.
3) **SDN Rule Engine:** The main function of SDN Rule Engine is to generate rules for SDN based on the defenses generated. These rules instruct at how to adjust networks devices. SDN primitives are used which achieve the desired goals by using a systematic method. This method is a combination of five methods Drop, Forward, Reflect, Isolate and copy. These five SDN Primitives are explained below:

   - Drop is used to reject a network traffic which may seem to be malicious. So, this primitive is basically sued to block traffic.
   - Forward is used to pass on the network traffic to their destination. This process should follow SDN rules. So basically, this is used when no operation is need on the network traffic.
   - Reflect is used to change the destination of the network traffic, both for incoming and outgoing network traffic.
   - Isolate limits a particular network traffic to a single destination/area.
   - e. Copy creates another copy of the network traffic. This is normally used for logging or monitoring the traffic flow.

   These five primitives can be used in combination and in any sequence repeatedly to come up with a line of defense. Each SDN rule may possess an SDN Primitive.
4) **SDN Rule Distributor:** SDN Rule Distributor acts as a bridge between SDN controllers and SDN Rule Engine. The generated SDN rules are sent to SDN Controllers using this. The network devices that are present in the SDN network will be formed into groups. Each group will have a controller which controls and manages them. Its the controller which has the right to send SDN rules to the associated SDN devices. To make sure that the related rules are sent to groups, the controller should have a map of the network devices.

B. Policy based Security architecture

Policy based architecture can be run as a part of SDN controller or as an application on top of it. This paper explaining the later type and calls it PbSA [8]. Policy based security architecture is shown in the Fig 4.

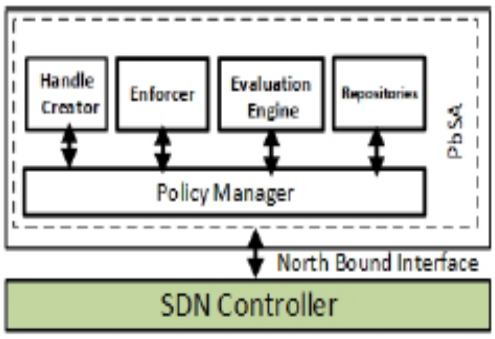

Fig. 4. Policy based Security Architecture for SDN [8]

PbSA is implemented in northbound interface of the SDN and can be used on a single/multiple host. The basic assumption here is that each Controller has a Policy Server. There are 5 components that make up a Policy Server. They are the Topology and Policy Repositories, a Policy Manager, a Policy Evaluation Engine, a Policy Enforcer and a Handle Creator.

1) **Topology Repository and a Policy Repository:** These are present with every controller which takes up the role of maintaining and updating them. The topology repository, as the name suggests, maintains information about the topology of the network and policy repository policy details explained in a simple language.
2) **Policy Manager:** This is responsible in managing every single operation of the security system.
3) **Evaluation Engine:** This is used to gauge the incoming traffic into the networks system and to see if the available security policies are sufficient for the traffic. After evaluation, the Policy Manager again determines the flow policies which are then transferred to the Enforcer system.
4) **Enforcer:** To implement the rules, the Enforcer fetches the data from south bound interface to the switches and enforces the rules from Policy Manager.
5) **Handle Creator:** Packet Handle Creators are used to the required handles from the controller and these handles are in turn used to measure the authenticity of both the packet and the policy enforcements.

III. Issues and Problems

Now we will see the security threats or issues with the SDN architecture. SDN threats are concentrated comparison to the traditional networks. These threats are as summarized below [9] [10] [11].

1) **Spoofing:** A user fakes MAC or IP address or also ARP packets and acts as an authentic user of in the network. In this way they fool the system. They can obtain data illegally through this procedure. Though this does not cause data loss, it definitely causes data theft.
2) **Tampering:** Here, the attacker modifies the data according to his need. This will change the original data. The process to do this may be by making the controller use different flow rules and modify/provide new packets of data in the system.
3) **Repudiation:** In repudiation the attacker changes the source address and sends it to the attackers desired destination. In this the receiver system cannot accurately determine the senders actual source address.
4) **Information Disclosure:** If an attacker has the information from the network system that is not intended for him to be knows, then it is called as information disclosure. This could mean side channel attacks in the concept of SDN.
5) **Denial of Service:** These types of attacks are called as DOS attacks. In DOS attacks, an attacker limits the systems ability to send/receive data packets This may be done by depleting system bandwidth or/and the resources.
6) **Know Your Enemy:** Know Your Enemy (KYE) is a technique in which an attacker gathers information about a network, that can showcase attack detection capabilities for network security tools. It can also show quality of the network or its policies.
7) **Vulnerable Controller:** The controller in an SDN network performs majority of the important actions including gathering of information, routing actions and configuration of networks. If attackers can somehow seize the controller actions, then it would be very easy for the them for data theft.
8) **Risks by Open programmable interfaces:** The interfaces in SDN are quite open. This openness causes security threats. Firstly, the software threats are present as its exposed to attacker and secondly, SDN has huge number of interfaces to application layer which can cause the infection of an evil code (may be virus).
9) **SDN Switch:** SDN switch is composed of both hardware and software which makes it more vulnerable to attacks.
10) **Application software:** Application software sits right on top of the controller and is associated with it through the north-bound API. So, when this calls a function with a malicious code, this can get into the controller and cause an evil attack. So, its always easy to attack through application software.

IV. future trends

SDN increases security due to central intelligence and programmability of the network. But these can also be a source of risk . therefore, in this section we will see the areas which need to be addressed before SDN can be commercially viable. So here we will see some challenges in research and future trends [12] [13].

1) **Programming and Developing models:** SDN allows users to create/modify applications on the go. Though this brings in a new innovative technology into action, it also builds up security threats. The applications have been given too much liberty and also the components of SDN are loosely held. This independence can cause severe security challenges. Integration in these systems are difficult as the environment in which the control plane is developed is totally different from the one in which the applications are developed. This might cause security collision if vulnerabilities from data plane are considered. Therefore, these models, interfaces and environments have to be standardized.
2) **Class-Based Application Security:** The security applications in SDN need a direct view of the network statistics. But the control plane in SDN provides these applications with its own view of network statistics. This may cause incoherency with the actual network view. Hence, these applications require direct access to the network statistics. But in the current environment, providing a direct view may be challenging in terms of security. Here, the need arises to categorize the applications into classes according to their requirements.
3) **Scalability and Security:** The control plane in a SDN architecture is logically centralized. Therefore, as the network size grows, the volume of traffic on the controller grows. But controller may not be capable of handling the huge amount of traffic. This inconsistency between controller and switch may cause attacks by starting malicious communication between them. Therefore, multiple controllers may be suggested. But yet again, adding too many controllers may cause failure. Therefore load balancing is an important concept along with maintaining security.
4) **Control- Data planes Intelligent Trade off:** In SDN availability of a single controller and multiple forwarding units may cause latency in network information transfer/exchange. If the number of switches are increased, the time taken by the controller to respond to network traffic increases. Therefore another research challenge is to intelligently build a convincing trade-off between control plane and the data-plane to manage network flow without latency.
5) **Network Security Automation:** As the users get added to the systems, the network flow increases. This leads to development of more intricated model and hence the complexity of the network increases. Along with it the issues of reliability, availability and scalability also

increases. The system becomes prone to errors if the communication between the devices of the network are not managed properly. All these lead to increased responsibility for network administrators whose mistakes, even if minor, may cause huge security breach. Therefore, the need arises to create automated systems with minimal human intervention. These automated systems may also be used to protect data and also to recover data loss.

6) **Secure Network Map:** It is a huge concern to match the virtual and physical mappings of network components. It becomes important to know and retrieve the state of network elements at a given point of time. Virtual networks are mapped on to physical networks. The applications gets details from these to carry out a necessary action. Therefore the challenge here is to how to give out the information of the underlying network to the applications to carry out necessary function by upholding the security of the system.

REFERENCES

[1] A Survey of Security in Software Defined Networks, Scott-Hayward, S., Natarajan, S., & Sezer, S. (2016). A Survey of Security in Software Defined Networks. IEEE Communications Surveys and Tutorials, 18(1), 623-654. https://doi.org/10.1109/COMST.2015.2453114.

[2] Yao, Zhen & Yan, Zheng. (2016). Security in Software-Defined-Networking: A Survey. 10066.319-332. 10.1007/978-3-319-49148-6_27.

[3] Wolfgang Braun, and Michael Menth, Software-Defined Networking Using OpenFlow: Protocols, Applications and Architectural Design Choices. Department of Computer Science, University of Tuebingen, Sand 13, Tuebingen 72076, Germany, May 2014.

[4] Software-Defined Networking: The New Norm for Networks.ONF White Paper, April 13, 2012

[5] Wolfgang Braun, and Michael Menth, Software-Defined Networking Using OpenFlow: Protocols, Applications and Architectural Design Choices. Department of Computer Science, University of Tuebingen, Sand 13, Tuebingen 72076, Germany, May 2014.

[6]] Kamal Benzekki, Abdeslam El Fergougui and Abdelbaki Elbelrhiti Elalaoui, Software-defined networking (SDN): a survey, Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/sec.1737, Feb 2017.

[7] SecControl: Bridging the Gap Between Security Tools and SDN Controllers, Li Wang and Dinghao Wu, College of Information Sciences and Technology, The Pennsylvania State University, lzw158,dwu@ist.psu.edu

[8] Karmakar, Kallol & Varadharajan, Vijay & Tupakula, Udaya & Hitchens, Michael. (2016). Policy based security architecture for software defined networks. 658-663. 10.1145/2851613.2851728.

[9] Shu, Zhaogang & Wan, Jiafu & Li, Di & Lin, Jiaxiang & Vasilakos, Athanasios & Imran, Muhammad. (2016). Security in Software-Defined Networking: Threats and Countermeasures. Mobile Networks and Applications. 21. 10.1007/s11036-016-0676-x.

[10] Eskca EB, Abuzaghleh O, Joshi P, Bondugula S, Nakayama T, Sultana A. Software Defined Networks Security: An Analysis of Issues and Solutions, International Journal of Scientific & Engineering Research, Volume 6, Issue 5, May-2015 1270 ISSN 2229-5518

[11] Conti, Mauro, Fabio De Gaspari and Luigi V. Mancini. Know Your Enemy: Stealth Configuration-Information Gathering in SDN. GPC (2017).

[12] Scott-Hayward, S, Natarajan, S & Sezer, S 2016, 'A Survey of Security in Software Defined Networks' IEEE Communications Surveys and Tutorials, vol. 18, no. 1, pp. 623-654. https://doi.org/10.1109/COMST.2015.2453114

[13] I. Ahmad, S. Namal, M. Ylianttila and A. Gurtov, "Security in Software Defined Networks: A Survey," in IEEE Communications Surveys & Tutorials, vol. 17, no. 4, pp. 2317-2346, Fourthquarter 2015. doi:10.1109/COMST.2015.2474118