www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume – 5 Issue -02 February, 2016 Page No. 15668-15672

A Survey on SDN: An Unprecedented approach in Networking

Mr. Nithin Kumar[1], Ms. Nidhi K N[2], Mr. Sachin Acharya T[3]

[1]P.E.S College of Engineering,
Mandya-571401, India
nithingowda021@gmail.com

[2]P.E.S College of Engineering,
Mandya-571401, India
nidhi.kn@gmail.com

[3]P.E.S College of Engineering,
Mandya-571401, India
Sachinacharya71@gmail.com

Abstract: ***Software Defined Networking (SDN) is an emerging architecture in the field of networking in which the control plane and forwarding plane of traditional networking devices (e.g. Switches, Routers) are decoupled. The network-wide traffic flow can be directly programmed. SDN plays an important role in today's enterprises and applications with drastically changing requirements which are monitored and adapted by the change in traffic flows through the networking devices. This survey paper on SDN provides an outline on the standard communication interface, characteristics of SDN and the pros and cons that are associated with SDN architecture.***

Keywords: Networking Architecture, SDN, OpenFlow™, Threat Vectors

I. INTRODUCTION

A network is a collection of nodes that allows us to exchange data among them. A set of nodes in a network can be communicated by using either physical transmission medium or wireless medium. The well-known network is the *internet*. The task of maintaining and operating an intrinsic computer network is enormous.[2] To signify the required high level network policies, the network operator configures every network device individually using low level commands (i.e. vendor specific).

In this dynamic networking environment, operators have limited mechanism to automatically respond to network events. Enforcing the required network policies is hence difficult. Separating control plane from data plane, then gathering and controlling all control planes from a logically centralized location is the key aspect of Software Defined Networking. Here the network switches become the forwarding devices.

"Software-Defined Networking (SDN) is an emanating architecture in networking. SDN is controllable, intensive, cost-effective, adaptive, and provides high-bandwidth to today's dynamic applications. This architecture splits the network control plane and forwarding plane and makes the control of network to be directly programmable. Also, applications make use of abstraction of underlying infrastructure to provide network services. Open Flow™ protocol is a basic building block of SDN solutions." [3] [4]

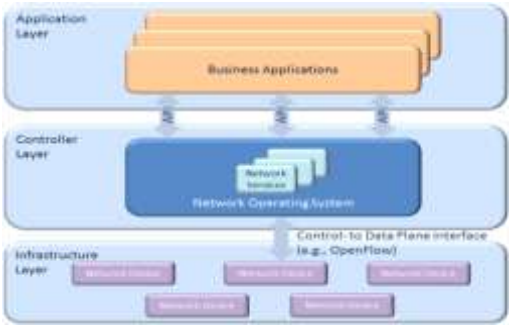

Figure 1:- layered view of SDN architecture

As shown in figure 1, this logically centralized SDN controller system offer several fore deals. Like it's simpler, and less error prone to modify network policies through SDN software. Additionally, spuriously changing states are automatically handled. [5] For example, considering an application, in which, the traffic flow becomes unpredictable with changing needs of the user. Here in case of heavy traffic, during certain peak hours, the path of storage needs to be varied else the network tends to slow down. Also, switching over of states must be updated in a faster mode. For such cases where changing over of "east-west" machine to machine traffic before running the data into the end users in a classic "north-south" traffic patter SDN is beneficiary .

Certain key trends drive the need for a new network paradigm. First, the rapid increase in usage of personal mobile devices, to access corporate network .The corporate data need to be used in a protected manner. Second, with the rise of cloud services the complexity of network becomes voluminous. This is due to the demand of resources, which is the actual fundamental in cloud services. Third, handling today's "big data" where scaling of data is in high demand. Scaling of network occurs

here because of increasing demand for additional network capacity. And scaling of network is expensive.

In comparison with the existing traditional networking system, SDN could be a better approach for these changing needs .SDN trying to meet these demands is the main objective of this proposal. The organization of the survey paper is as follows: section II deals with the characteristics of SDN, section III provides an overview on standard communication interface (OpenFlow™) of SDN [7], section IV explains the threats that are associated with SDN technologies, section V provides secure and dependable nature of SDN, section VI provides the benefits that are associated with SDN technologies, finally this survey paper ends with conclusion and few references.

II. CHARACTERISTICS

The characteristics of SDN architecture are as follows:

• **Programmable**: Control plane of the network can be directly programmable because it is uncoupled with forwarding functions. [1]

• **Active**: The abstraction of Control plane from forwarding function makes the network administrators to dynamically monitor the network traffic flow so as to satisfy changing requirements.

• **Centrally managed**: Intelligence of network is logically centralized in SDN controllers. This is done in order to provide a global view of network, to applications, as a single logical switch.

• **Programmatically configured**: SDN makes network managers and administrators to configure, control, and secure network resources very easily through automated SDN programs. [4] These programs can be written easily by managers and administrators, relying on proprietary software are not required.

• **Open standards and vendor-neutral**: SDN implementation when it's through open standards, it simplifies design and operations of the network. This simplification occurs because the specifications are provided by SDN controllers instead of multiple vendor-specific devices and protocols.

Traditional networking architectures are not suitable to meet the changing requirements of now a day's/present/current applications, enterprises, and end users. So that *Open Networking Foundation (ONF)* is an industry that made its maximum efforts in bringing Software-Defined Networking (SDN) which may create future era in networking field.

III. OPEN FLOW

OpenFlow™ is the first protocol that defines standard communications interface between the control plane and forwarding plane of SDN architecture. [7] This protocol allows to directly access the forwarding plane inorder to manipulate the forwarding plane of network devices such as switches and routers. The absences of open standard communication interface in the forwarding plane have led today's networking devices to behave monolithic and closed. This closed and monolithic nature of networks can be removed by the use of OpenFlow™ protocol that provides an open standard communication interface in networking. OpenFlow™ protocol is most needed protocol to move network control out of the networking devices to logical control software.

The OpenFlow™ protocol should be implemented on both sides of the interface such as network devices and the SDN controller. It analyzes the network traffic flows to identify network wide traffic based on pre-defined criteria's. These criteria's are statically or dynamically created and programmed by the SDN controller. [9] OpenFlow™ protocol also allows enterprises to define the traffic flow through network devices based on certain criteria's such as application, usage analysis, and resources. Also, OpenFlow™ protocol in SDN architecture allows the application's to program the network on the basis of traffic-flows. Hence, OpenFlow™ protocol based SDN architecture provides extreme granular control, and makes the network to adapt to real-time changing requirements of the applications, enterprises, and end users. Currently IP-based routing does not provide the required level of control, and adaptive nature for changing different requirements.

The OpenFlow™ protocol is the basic building block for software-defined networks and presently it is the one and only standard communication interface protocol available for SDN that allows us to direct access and manipulation of the forwarding plane of network devices. [9] Networking devices can support OpenFlow™ protocol based forwarding together traditional forwarding, and making easy for any applications and enterprises to progressive adaption of SDN architecture. The Open Networking Foundation (ONF) is chartered to standardize OpenFlow™ protocol, that work through the technical group of people responsible for the protocol configuration, testing, and other processes and ensuring the interoperability between network devices and control software from different vendors.

IV. THREAT VECTORS

Two important properties of Software-defined networks make them vulnerable to malicious users, are as follows

- The ability to control the network by means of software, hence always subject to bugs.
- SDN is a centralized control, where if attackers gain access to the control software may lead to access over entire network.

Seven major threat factors/vectors are found in SDN. These different threats need to be dealt differently. But if SDN is properly designed and deployed, the new network environment would be one of the best network advancement not only in terms of functionality but also in resilience.[10] [11]

1. Forged or faked traffic flows

These threats could be triggered by faulty devices or by a malicious user. An attacker may use network elements to launch a DoS attack against OpenFlow switches and controller resources. In addition to this, if the attackers take control over application server, they can forge the entire network. As MAC address and authentication ports are genuine it appears to be more realistic. Use of *intrusion detection systems* (IDS) is one possible solution to these type of threat vectors.

2. Attacks on vulnerabilities in switches

A single switch could drop or slow down packets in the network. Also it could clone, deviate, inject network traffic or even forge requests to overload the controller or neighboring switches. Software attestation and monitoring the abnormal behavior of network devices are a few probable solutions to overcome these kinds of attacks.

3. **Attacks on control plane communications**

Usage of TLS/SSL does not assure secure communication, and that compromises the controller device link.TLS/SSL model is does not establish and assure trust between controllers and switches. Attackers once gain accessibility to the control plane; they may get capable of aggregating enough power force to launch DoS attacks. This mistrust may lead creation of a virtual blackhole network allowing data leakage while the normal production traffic flows. Use of oligarchic trust models with multiple trust-anchor certification authorities and securing communication with threshold cryptography across controller replicas could be possible solutions. Furthermore the use of dynamic, automated and assured device association mechanisms may be considered so as to guarantee trust between the control plane and data plane devices.

4. Attacks on and vulnerabilities in controllers

This is the most severe threats to SDNs. A faulty or malicious controller could cause extreme problems to the entire network. Malicious application could do as they please in the network, since controllers only provide abstractions that translate into issuing configuration commands to the underlying infrastructure.

5. Lack of mechanisms to ensure trust between the controller and management applications

Controllers and applications lack the ability to establish trusted relationships. Here the techniques used to certify network devices are different from those used for applications. Autonomic trust management mechanism could be used to guarantee that the application is trusted during its lifetime.

5. Attacks on and vulnerabilities in administrative stations

Administrative stations used in SDNs to access the network controller. The threat surface as seen from a single compromised machine increases dramatically in SDNs. Use of protocols requiring double credential verification could be one such possible solution. Furthermore, could be the use of assured recovery mechanisms to guarantee a reliable state after reboot.

6. Mistrusted resources for forensics and remediation

Forensics, be it for the use of science and technology or investigating and establishing facts about an incident, finally it's the information of the network components that are taken. And that very data/info must be genuine enough to put in use. Trustworthiness of data is counted here. Likewise, remediation requires safe and reliable system snapshots to guarantee a fast and correct recovery of network elements to a known working state. *Logging* and *tracing* are the most common mechanisms that are obligatory in data and control planes. Nevertheless, for more efficiency, they should be indelible. Moreover, logs should be stored in remote and secure environments.

V. Security and dependability

Considering the security and dependability perspective, there is always a need to find out faults and detect intrusions. The two main fault models are crash and byzantine. Crash model is a narrow subset of arbitrary model which addresses the faults with respect to crashed process and operating system. Additionally, byzantine fault tolerant model addresses the abnormal issues, intentional and unintentional faults. Machine replication [12] can be used to mask the faults automatically as soon as they occur. Errors in a system can also be removed using techniques such as self-healing and proactive-reactive recovery techniques [13]. Intrusion tolerant systems help in maintaining the reliability, confidentiality and integrity of the system even though the system has faults or effected by successful attacks. Thus, an efficient technique has to be adopted to ensure that the system is dependable and fulfills all security goals of a system. Some of the precise techniques which can be used to promote dependability and security in networks will be discussed below.

1. **Replication:** Here generally the controllers are replicated to increase the dependability. In certain cases, even the applications are replicated. A mixed approach of replicating both controller and application can tolerate both hardware and software faults, accidental or malicious. Replication masks the failures to isolate faulty controllers and/or applications.

2. **Diversity:** This technique helps in increasing the robustness of security and dependability[12][13]. The basic principle of this mechanism is to avoid common-mode faults such as software bugs or vulnerabilities. For example, it is known that off-the-shelf operating systems, from different families, have few intersecting vulnerabilities [14], which means that OS diversity constrains the overall effect of attacks on common vulnerabilities. Whereas in SDNs, the same management application could run on different controllers.

3. **Self-healing mechanism:** under adversary circumstances, proactive and reactive mechanisms can bring the system to healthy state compromising with the components and keep it virtually active forever. Exploring diversities in the recovery mechanism strengthens the system.

4. **Dynamic device association:** If a switch is associated with a single controller then its control plane cannot tolerate the faults. Once the controller fails, the control operation of the switch fails and the switch will need to associate with another controller. A switch associated with different controllers would be able to automatically tolerate faults (crash or Byzantine, depending on the

configuration). This method also increases the control plane throughput and reduces control delay [15].

5. **Trust between devices and controllers**: control plane trustworthiness is always gained by establishing trust between the devices and controllers. Network devices should be allowed to associate with controllers dynamically which can be done by having authenticated white lists of known trusted devices kept at controllers. Malicious behavior could be reported by other switches or controllers, based on anomaly detection algorithms. Once the trustworthiness of a switch or a controller would go below an accepted threshold, the switch would be automatically quarantined by all devices and controllers.

6. **Trust between applications and controllers software**: A dynamic trust model is usually required as the software component's present/current behavior changes due to aging, exhaustion, bugs, or attacks. Autonomic trust management is used in component-based software systems. They use a holistic notion of trust to allow a trusted component to assess the trustworthiness of the trustee component. This is basically done by observing its behavior and measuring it based on quality attributes, such as availability, reliability, integrity, safety, maintainability, and confidentiality.

7. **Security domains**: Isolated security domains are most commonly used technique. Security domains in SDN control platforms can be explored using techniques such as sandboxing and virtualization. These techniques enable the design of strong isolation modes, through well-defined interfaces that allow minimal (only restricted and strictly necessary) set of operations and communication between different domains.

8. **Secure components**: These components are one of the essential building blocks of a secure and dependable system as they assure confidentiality. Some of the security components (like trusted computing bases) can be used to store sensitive security data (e.g., crypto private keys) and execute basic operations on it. Thus, the sensitive data will have its confidentiality assured even if the system is compromised.

VI. BENEFITS

OpenFlow-based SDN technologies enable IT to address the high-bandwidth and dynamic natures of different applications, adapt the network to ever-changing business needs and significantly reduce operations and management complexity.

OpenFlow-based SDN architecture benefits the enterprises and includes the following:

• **Centralized control of multi-vendor environments**: The switches, routers, and virtual switches from any vendors can be used to control any OpenFlow-enabled network device by SDN control software. Rather than having to manage groups of devices from individual vendors, IT can use SDN-based orchestration and management tools to quickly deploy, configure, and update devices across the entire network.
• **Reduced complexity through automation**: A flexible network automation and management framework, which makes it possible to develop tools that automate many management tasks, are offered by OpenFlow-based SDN. They promote IT-as-a-Service and self-service provisioning models.
• **Higher rate of innovation**: IT network operators literally program—and reprogram—the network in real time to meet specific business needs and user requirements as they arise. Adoption of SDN accelerates this business innovation.
• **Increased network reliability and security**: OpenFlow-based SDN architecture eliminates the need to individually configure network devices each time an end point, service, or application is added or moved, or a policy changes, which reduces the likelihood of network failures due to configuration or policy inconsistencies. They ensure that access control, traffic engineering, quality of service, security, and other policies are enforced consistently across the wired and wireless network infrastructures, including branch offices, campuses, and data centers. They have a reduced operational expenses, more dynamic configuration capabilities, fewer errors, consistent configuration and policy enforcement.
• **More granular network control**: The policies are applied at a very granular level, including the session, user, device, and application levels, in a highly abstracted, automated fashion.

CONCLUSION

In this paper, we have surveyed on the SDN architecture which is centralized and directly programmable. The characteristics of the above discussed approach has made it elegant and promising. OpenFlow™ can be considered as the one and only standard communication interface available for SDN. Considering the threats, security and dependability issues, SDN can be accepted to be an emerging and efficient technology in the field of networking.

References:

[1] Bernardo and Chua (2015). Introduction and Analysis of SDN and NFV Security Architecture (SA-SECA). 29th IEEE AINA 2015. pp. 796–801.
[2] The Future of Networking, and the Past of Protocols, Scott Shenker (video of talk at Ericsson, slides of talk at ONS'11)
[3] Li Erran Li, Z. Morley Mao, Jennifer Rexford, EWSDN 2012, Towards Software-Defined Cellular Networks
[4] Mohammad Banikazemi, David Olshefski, Anees Shaikh, John Tracey, Guohui Wang, IEEE Communications Magazine Feb 2013 Meridian: An SDN Platform for Cloud Network Services
[5] IETF Software Driven Networks http://nerdtwilight.wordpress.com/2012/01/30/sdn-double-vision/
[9] Canini, Marco and Venzano, Daniele and Peresini, Peter and Kostic, Dejan and Rexford, Jennifer; et al. (2012). A NICE Way to Test OpenFlow Applications. NSDI. pp. 127–140.

[6] Giotis, K and Argyropoulos, Christos and Androulidakis, Georgios and Kalogeras, Dimitrios and Maglaris, Vasilis (2014). "Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments". Computer Networks 62: 122–136.
[7] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, Jonathan Turner , OpenFlow: Enabling Innovation in Campus Networks, CCR 2008
[8] What OpenFlow is (and more importantly, what it's not) http://networkheresy.com/2011/06/05/what-openflow-is-and-moreimportantly- what-its-not/ & What Should Networks Do For Applications? http://networkheresy.com/2013/04/13/what-shouldnetworks- do-for-applications/#comment-922
[9] ONF, Brocade, "Open Flow Products - Brocade MLX Series",https://www.opennetworking.org/sdn-openflow-products/662-brocademlx-series
[10] Braga, Rodrigo and Mota, Edjard and Passito, Alexandre (2010). "Lightweight DDoS flooding attack detection using NOX/OpenFlow". Local Computer Networks (LCN), 2010 IEEE 35th Conference on. pp. 408–415
[11] Improving Network Management with Software Defined Networking, *Hyojoon Kim, Nick Feamster*, IEEE Communications Magazine Feb 2013
[12]Kampanakis, Panos and Perros, Harry and Beyene, Tsegereda. SDN-based solutions for Moving Target Defense network protection (PDF). Retrieved 23 July 2014.
[13] A Clean Slate 4D Approach to Network Control and Management, Albert Greenberg, Gisli Hjalmtysson, David A. Maltz, Andy Myers, Jennifer Rexford, Geoffrey Xie, Hong Yan, Jibin Zhan, Hui Zhang, CCR 2005
[14] Self-Organizing Networks (SON): http://www.3gpp.org/SON
[15] Bram Naudts, Ghent University – iMinds, " Techno-economic analysis of SDN", 2012 IEEE EWSDN

Author Profile

Mr, Nithin Kumar, Teaching Assistant under TEQIP, AICTE New Delhi has completed Bachelor of Engineering in Computer science and Engineering at B.G.S College of Engineering, Mandya in the academic year 2014 and is perusing Masters in Technology in Computer Science Engineering at P.E.S College of Engineering, Mandya.

Ms. Nidhi K N, has completed Bachelor of Engineering in Computer science and Engineering at Government Engineering College, Hassan. in the academic year 2013 and is perusing Masters in Technology in Computer Science Engineering at P.E.S College of Engineering, Mandya.

Mr, Sachin Acharya T, Teaching Assistant under TEQIP, AICTE New Delhi has completed Bachelor of Engineering in Computer science and Engineering at P.E.S College of Engineering, Mandya. in the academic year 2014 and is perusing Masters in Technology in Computer Science Engineering at P.E.S College of Engineering, Mandya.