Software Defined Networking: Techniques, Issues and Future Trends

Praarthana Ramakrishnan
Department of Computer Science
Clemson University
Clemson, US
Email: *praartr@clemson.edu*

***Abstract*—Software Defined Networking (SDN) referred as "radical new idea in networking has evolved as an efficient network technology that supports dynamic nature of network functions while enabling innovation through network programmability. In this article, the techniques, issues and the future trends of Software Defined Networking are discussed.**

Keywords—Software-Defined Networking, Security, Issues, programmable networks, data plane, control plane, trends.

I. INTRODUCTION

Software Defined Networking allows flexible behavior of the network and is designed to use application programming interfaces (API). It is made flexible by decoupling hardware from the control decisions. The API allows easy interface of the network applications [2]. SDN has a centralized system that collects information from the system and improve the policies dynamically. It allows developers to depend on network resources same as they do on storage resources.

Software Defined Networking was designed to facilitate simple programmatic control of the network. Usually data communication networks consists of hosts connected by switching elements such as switches and routers [1] as well as communication data to carry the data. These connecting elements (routers, switches) are "closed systems" with limited control interfaces. So, it is quite difficult to make changes and use new versions of existing protocol.

SDN deploys separation of hardware from the control logic that makes it easier to deploy new protocols and applications. In an SDN environment, the application programming interfaces are used by applications to reach through to the network switches. The Software Defined Networking is used to simplify forwarding hardware and the decision making network controllers. SDN is used to manage networking elements using software running on an external server. This is met by an architecture split between the forwarding element and the control element.

The vast growth in the traffic volume over the network makes it necessary to manage paths circulating the network and fast recovery if there are network problems. But, these technologies are restricted due to the closed connection between the hardware and the software of networking devices. SDN architecture proposes to separate the data and the control plane for independent development. The paper is organized as follows: In Section II, the main techniques used in SDN are described. Section III provides an overview of the issues and problems involved in SDN. Finally, Section IV describes the future trends in Software Defined Networking.

II. MAIN TECHNIQUES

SDN is described in the article with the Open Networking Foundation (ONF): "*In the SDN architecture, the control and data planes are decoupled, network intelligence and state are logically centralized, and the underlying infrastructure is abstracted from the applications.*" [5]
Thus, Software Defined Networking focuses on the following important four features:

- Separating data and control planes
- Centralized controller
- Open interfaces between the devices
- Programmability of the network

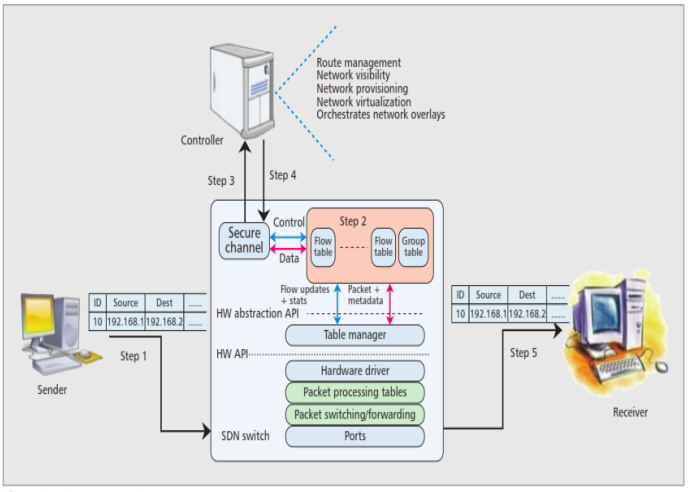

Figure 1 : The operation of SDN (Controller Switch)

A. *ForCES:*

The ForCES (Forwarding and Control Element Separation) Working Group, proposes the network device's internal architecture as separation of the control element from the

forwarding element. There are two logic entities called the Forwarding Element (FE) and the Control Element (CE).

The Software Defined Networking has two popular architectures namely ForCES and Openflow. Both these techniques follow separation of data and control planes; and information transfer between planes is standardized. However, they differ in terms of design, architecture, forwarding model and the protocol interface. The operation of SDN is illustrated in the Fig. 1 [5].

The FE is responsible for providing the per-pocket handling. The CE executes control and signaling functions and utilizes ForCES protocol to instruct FE about how to handle packets. Master-Slave model is used where Fes are slaves and CEs are masters. The building block called the LFB (Logical Function Block) is a functional block residing on the Fes that is controlled by the CEs through the ForCES protocol.

B. OpenFlow:

In OpenFlow, the forwarding device or the OpenFlow switch contains one or more flow tables and layer that communicates with the controller via OpenFlow protocol securely. The elements of the OpenFlow architecture is shown is Fig. 2 [3].

The Flow tables consists of flow entries that consists of: (1) match fields, used to match incoming packets, it may contain information found in the packet header, ingress port, and the metadata. (2) Counters, collects statistics for the particular flow. (3) A set of instructions, which is applied upon the match. As soon as a packet arrives, the packet header fields are extracted and tried to match with the matching fields. If a match is found, appropriate set of instructions are applied. If not, the action depends on the instruction as defined by the table-miss flow entry.

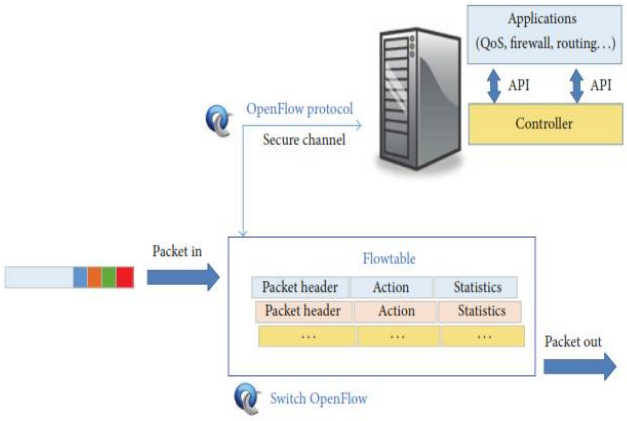

Figure 2 : Elements of the OpenFlow architecture

OpenFlow supports multiple tables and pipeline processing. In the case of switches that have both OpenFlow and non OpenFlow ports, is to send non-matching packets using regular IP forwarding methods.

C. Network Interfaces

SDN has a three-tier architecture, where the applications and the high-level instructions are placed at the top-tier, a controller appears at the middle tier and the third tier consists of physical and virtual switches. There are two interfaces that represent the communication between the tiers. The northbound API interface allows communication with the higher level component, and the southbound API interface allows network component to communicate with the low level component. The SDN functional architecture is shown in the Fig. 3 below [5].

- *Northbound APIs*

The Northbound API interface enables applications to program the network and allows services to be requested. The top tier includes automation and data management and provides network functions such as routing and security.

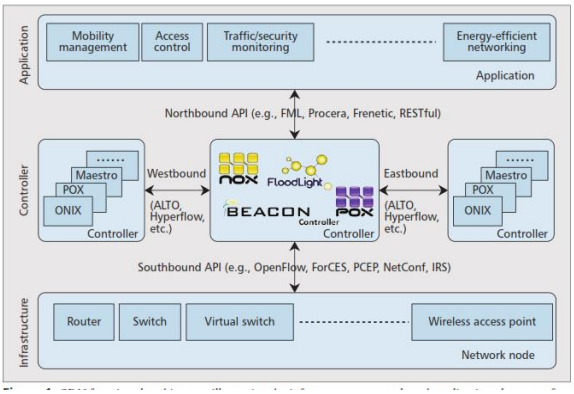

Figure 3 : SDN functional architecture

- *Southbound APIs*

OpenFlow is used as the southbound API that consists of a set of commands for data forwarding. These commands are used to identify the network topology and the behavior of physical and virtual switches are defined. There are other southbound API other than OpenFlow that any organization can use.

D. Network Operating Systems (NOS):

Network Operating Systems is run on the southbound interface. NOX is an example of NOS and consists of two elements: processes of controller and network view. The users make decision and set network behavior through these processes. The traffic is handled such that the packets with the same headers are treated equally.

NOX uses events with different priorities when an event occurs in the network, due to the dynamic nature of traffic. NOX implementation is done in C++, thereby providing high performance. However, there are some problems that creates malfunctions in NOS.

III. ISSUES AND PROBLEMS

There are two factors that appears as threats to SDN. The first factor is that the network can be controlled by software, which is always prone to bugs and other vulnerabilities. The second factor is that the "network intelligence" in the controller is centralized. Therefore, anyone who has access to the server can control the entire network. Fig. 4 shows the various threats involved in SDN [4]. This sections provides the various issues or threat factors that is associated with the SDN [4].

- Forged Traffic Flows, can be used to attack switches and controllers. An attacker can launch a DoS attack against OpenFlow switches using network elements. The possible solution is use of intrusion detection systems that could help abnormal flows identification.
- Attacks on vulnerabilities in switches: Packets can be slowed down or network traffic can be deviated using a single switch. The use of automatic trust management solutions is a possible solution.
- Vulnerabilities in Control Plane communications can be used to generate DoS attacks or data theft. There are many man-in-the-middle vulnerable implementations of SSL being used in critical systems. Communication can be secured with threshold cryptography in order to avoid this threat.
- Lack of trusted resources for forensics, allows to understand the cause of problem and continue to secure mode recovery. Logs can be stored in secure environments to avoid this threat.
- Vulnerabilities in Administrative stations are used to access network controllers. The use of recovery methods to a reliable state after reboot is one of the possible solutions.

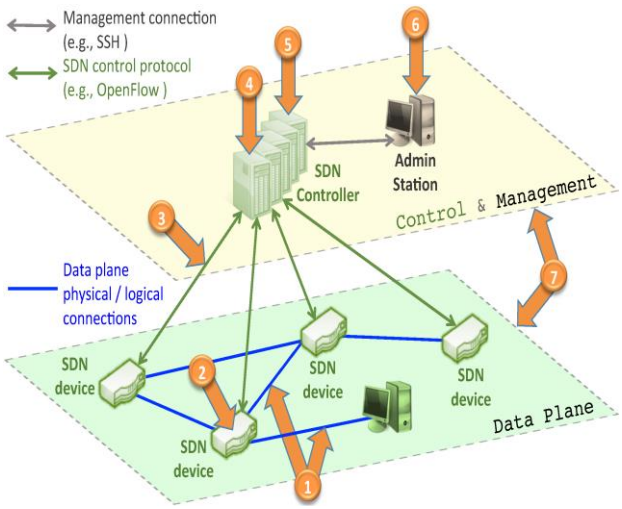

Figure 4 : SDN main threat vector map

- Attacks on controllers are severe threats to SDNs. An entire network could be compromised by a faulty controller. A vulnerable controller could affect the entire network. Securing all sensitive elements inside the controller (crypto keys/secrets) is a way to avoid this threat.

IV. FUTURE TRENDS

The future trends of Software Defined Networking is listed as follows:

I) Information - Centric Networking (ICN):

ICN is a new paradigm which increases the efficiency of delivery and availability of the content. The motivation was the current internet is information-driven, but the idea of location-based addressing and host-to-host communication is still being focused on the networking technology. [1]

The separation of the data and the control plane is associated with the decoupling between the information processing and forwarding in ICN. The future question that is being proposed in number of project is "how to combine ICN and SDN towards Software-Defined-Information-Centric Networks".

II) Heterogeneous Network Support:

Future networks will be more heterogeneous, connecting users from wired, infrastructure-based wireless (wireless mesh networks) to infrastructure-less wireless networks (mobile ad-hoc networks). Users will demand high quality communication, as mobile devices with multiple network interfaces become common. Self-organizing networks will become an important part of the future hybrid internet. The efficient utilization of resources will be a major challenge in future networks.

The use of shared physical medium and absence of managed infrastructure is the main cause of this challenge. SDN facilitates the deployment and management of network applications and services with greater efficiency. But, OpenFlow targets largely target infrastructure based networks (WiMAX, Wi-Fi access points). The OpenRoads project has envisioned in which users freely moves between wireless infrastructure while also providing support to network provider. [1]

V. REFERENCES

[1] A. Bruno Astuto Nunes, M. Marc, N. Xuan-Nam, O. Katia, and T. Thierry, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks.," IEEE Communications Surveys & Tutorials, vol. 16, no.3, third Quarter 2014.

[2] Keith Kirkpatrick, " Software Defined Networking." Communications of the ACM, vol. 56, no. 9, September 2013

[3] Angel Leonardo Valdiviso Caraguay, Alberto Benito Peral, Lorena Isabel Barona Lopez, and Luis Javier Garcia Villalba," SDN: Evolution and Opportunities in the Development Iot Applications", International Journal of Distributed Sensor Networks, vol., May 2014.

[4] K. Diego, Fernando M.V. Ramos and Paulo Verissimo, " Towards Secure and Dependable Software-Defined Networks" , ACM, August 2013.

[5] Sakir Sezer, Sandra Scott-Hayward, and Pushpinder Kaur Chouhan, Queen's University Belfast Barbara Fraser and David Lake, Cisco Systems Jim Finnegan and Niel Viljoen, Netronome Marc Miller and Navneet Rao, Tabula, " Are We Ready for SDN? Implementation Challenges for Software-Defined Networks," IEEE Communication Magazine, July 2013.