A Survey on Software Defined Networking Technology

Farheen Fatma Ansari[1], Prof. D.P.Mishra[2] and Prof. Sumit Kumar Sar[3]
[1,2,3]Department of Computer Science and Engineering,
Bhilai Institute of Technology,
Durg, Chhattisgarh, India.

[1]faruansari21@gmail.com ,[2] dpmishra@bitdurg.ac.in , [3]sumitsar@gmail.com

Abstract

In the early of mid 1990s, Software Defined Networking were introduced, but recently now a days have come more into existence and became well-established industry standards. The basic concept of SDN (Software Defined Networking) has introduced the expansive change to the conventional networks with the integration of the network by decoupling the forwarding hardware (data plane) from the control logic of the network (control plane). Software Defined Networking (SDN) is a network's platform based on a centralized control plane architecture with standardized interfaces between control and data planes. SDN enables fast configuration and reconfiguration of the network to enhance resource utilization and service performances. This new approach enables a more dynamic and flexible network, which may adapt to user needs and application requirements. To this end, systemized solutions must be performed in network software, proving to provide secure network services that meet the required service performance levels. Many network systems and network architecture adopted SDN, and vendors are choosing SDN as an alternative option to the fixed, predefined, and inflexible protocol stack. SDN permit us with dynamic, flexible and programmable functionality of network systems, as well as many other advantages such as better user experience, centralized control, reduced complexity, and a dramatic decrease in network systems and equipment costs. However, SDN capabilities and characterization, as well as workload of the network traffic that the SDN-based systems handle, determine the level of these advantages. Moreover, the allowed flexibility of SDN-based systems comes with a performance penalty. The capabilities and design of the underlying SDN infrastructure influence the performance of common network tasks, compared to a dedicated solution.

Keywords: Planes,Controllers,Forwarding.

1. INTRODUCTION

Today computer networks are very complex as more and more devices are increasing day by day along with the content they access[2]. Communication networks are growing in size and complexity at an ever-increasing rate, with the conventional infrastructure, protocol stack and network systems, which hardly provide adequate solutions to the contemporary networking demands[3]. The kind of equipment used in networks like Intrusion Detection system, switches, firewalls, Load balancers are typically very hard to manage by network administrator individually, the solution for this is Software Defined Networking. This triggered the emergence of a different approach to network systems architecture, called Software-Defined Networking (SDN). SDN, has been present for the last 20 years. Recently, OpenFlow succeeded in establishing itself as an SDN industry standard. It has changed the way we used to manage the networks.

Following are the basic principles of Software Defined Networking (SDN) are[2]:

- Control plane referred as a brain of the network which has a direct control over the Data plane, all the elements in the Data plane can be manipulated as per the needs, there is no need to configure each and every element of data plane individually.

- It separates the control plane from data plane (control plane contains the intelligence, control logic while data plane contains the physical infrastructure or low level network elements that are used for packet forwarding and switching).

Software Defined Networking (SDN) is a paradigm that is related to idea of offering the network resources to end users as a service (Naas) over an open.

2. BACKGROUND

In the traditional networks, both control plane and data plane are coupled inside the proprietary hardware. In a dedicated appliance network functionality is mainly implemented, 'dedicated appliance' refers to one or multiple switches, routers and/or application delivery controllers [2]. Within this appliance Most of the functionality is implemented in dedicated hardware only and for this purpose, Application Specific Integrated Circuit (or: ASIC) is often used [5].

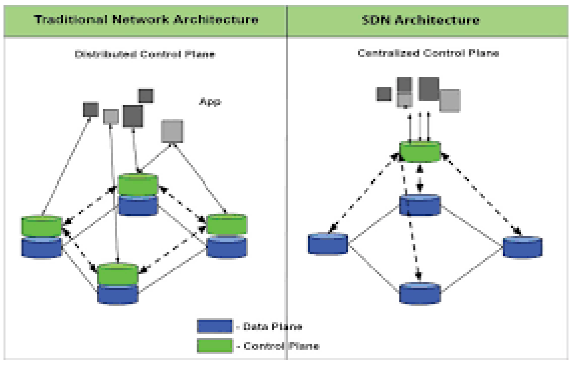
Traditional Network Architecture
SDN Architecture
Distributed Control Plane
Centralized Control Plane
App
- Data Plane
- Control Plane

Fig 1. Traditional Versus SDN Architecture

Limitations of Traditional Networks:

- **Network configuration was time consuming and Fickle:** Whenever an IT administrator needs to add or remove a single device in a traditional network many steps are needed. Firstly, the manual configuration of multiple devices used in the network like switches, routers, firewalls etc. The next step which he has to follow is to update numerous configuration settings, such as ACLs, VLANs and Quality of Service using device-level management tools. This approach makes it that much more complex for an administrator to deploy a set of policies which are consistent [5].

- **Multiple vendors:** As there includes multiple physical devices in traditional networks so it implies for multiple vendor environment which ultimately needs high level of expertise and extensive knowledge of all the devices present in the network.

- **Distributed control plane:** The intelligence of the network resides in the control plane in case of traditional networks it is residing in multiple places because of coupling of both data plane and control plane in network devices. It becomes very difficult to manage the network for a network administrator as configuration was a bit complex [5].

3. SDN ARCHITECTURES

The vital elements of SDN are separation of network control logic from network hardware operation and simplification of network device logic. This enables automatization of network management processes and logic above the control layer, centralization of network control logic, and openness of network resources through open standards and to end users.

SDN is defined as a three layered architecture; the main layer is control layer because controller resides in it, and controller acts as a brain to the network because it manages the flow of traffic from switches using flow tables.

Features of SDN architecture are as follows:

- **Programmability is Direct:** Because it is decoupled from forwarding functions network control is directly programmable [6].

- **Agility:** In order to meet network changing needs by dynamically adjusting network wide traffic flow is used. In software-based SDN controllers that maintain an overall view of the network, which appears to applications, policy engines as a single, logical switch, and network intelligence is logically centralized.

- **Configuration is programmable:** SDN lets network administrators to secure, configure, manage and optimize network resources very fast via dynamic, automated SDN programs, which they can be written by themselves because there is no more dependency on proprietary software [6].

- **Open standards-based and no more vendor-dependency:** Through open standards when SDN is implemented, it makes the network design and operations performed in a very simple manner because most of the instructions instead of multiple vendor-specific devices protocols, are provided by SDN controllers (like POX, Ryu, Opendaylight etc.) [7].

All three layers are dependent to each other and communicate with one another through some interfaces. The best advantage of SDN architecture is that it provides abstraction view of entire network for the applications it provides; this makes the network even more “Smarter”.

SDN Architecture contains the following three layers:

- **Application Layer:** It is composed of the applications which are communicating with controller in control layer through some interfaces called as Northbound APIs.The commonly used API in providing Northbound API is REST (Representation State Transfer) API. Applications in SDN can be like Firewall, Load balancer etc. [8].

- **Control Layer:** It is the middle layer of the SDN architecture and constitutes the SDN controller which acts as a brain of the network and has a global view over the network also known as Control plane.

- **Physical Layer:** It contains the infrastructure used in the network like switches, also known as Data plane. They provide packet forwarding and packet switching According to the controller, switches can only perform actions. The interface they use to communicate with controller situated in control layer is called as Southbound APls. The most common protocol used in providing Southbound APIs is OpenFlow Protocol [8].

Network Interfaces used in SDN:

SDN is a 3-layered architecture top layer includes the high level instructions, controller resides in middle layer and the third layer constitutes all the physical & Virtual switches used in the network. Within a network each control device is equipped with some interfaces (one or more), every control device is able to communicate with other components through these interfaces. A network interface is a software or protocol which provides the communication medium through its interface between two equipment’s or computer networks. The types of API’s used in SDN are as follows:

- **Southbound Application programming interface (API):** The communication between control layer and physical layer is done through this interface. For this many protocols are used like OVSDB, NETCONF, SNMP etc. but mainly OpenFlow protocol is used, it provides the programmatic control of forwarding rules from the data path given by network elements present in the physical layer [9].

- **Northbound Application programming interface (API):** The communication between control layer and application layer is done through this interface.

- **Westbound Application programming interface (API):** This interface acts as a channel for providing the interface between SDN control plane and different network domains [9].

- **Eastbound Application Programming interface (API):** communication is done from control plane to non SDN domains. Depends upon the technology used in non SDN domains its implementation is proportional [9].

OpenFlow Protocol:

The OpenFlow protocol is basically used protocol for the southbound interface SDN, which separates the data plane from the control plane. OpenFlow was originally proposed by Stanford University, and it is now standardized by the ONF. OpenFlow is an open interface and best suited for remotely controlling forwarding tables in network routers, switches and access points.

OpenFlow architecture constitutes the three basic concepts: -

- With the help of OpenFlow-compliant switches (that compose the data plane.) network is built.
- More than one OpenFlow controller is constituted in control plane of SDN network.
- A secure control channel connects the switches with the control plane [10].

4. SDN CONTROLLERS

The vital element of SDN network is studied to be its controller. It is defined as a platform which manages the flow of control to the routers and switches via Southbound OpenFlow protocol and applications via Northbound APIs. A collection of Pluggable part is contained by controller which performs different network tasks. Five most important commonly used controllers which are opensource.POX [13], Ryu [14], Trema [15], Floodlight [16], OpenDaylight [17] apart from these above mentioned controllers there are many others controller like Jaxon, NOX, Beacon, Maestro etc. because of less usage and poorly documented these controllers are not used.

- **POX:** It is developed and inherited from NOX controller. POX is python based SDN controller. Pythonic OpenFlow interface runs anywhere – Can bundle with installing free Py runtime for easy distribution. And the similar visualization tools and GUI as NOX are used.

- **RYU:** It gives the component based platform for SDN, for managing the network flow and applications it uses different APIs. Ryu helps in providing software components with well-defined API that make it easy for developers to create new network management and control applications. For managing different types of network devices, such as OF-config, OpenFlow, Netconf, etc. Ryu supports plenty of protocols. About OpenFlow, versions 1.0, 1.2, 1.3, 1.4, 1.5 and Nicira Extensions all are supported by RYU. Under the Apache 2.0 license all of these codes are freely available.

- **Trema:** For developing different controllers which use OpenFlow protocol for configuring and connection to the network devices (switches, routers) through Southbound APIs called as OpenFlow Controller, Trema provides a framework (open source) to them in the programming language like c and ruby.

- **Floodlight:** It is a java based OpenFlow controller, managed by ONF (Open Networking Foundation) and licensed by Apache. It specifies a “Forwarding instruction set” in which a remote controller can make changes in network behavior through some defined protocols through switch.

- **OpenDaylight:** It the largest open source SDN controller, managed ONF (Open Networking Foundation). A flexible common platform is provided by OpenDaylight which serves many purposes like Automated Service Delivery, NFV and cloud, Network Visibility and control, Network Resource Optimization. Model-driven service abstraction platform that allows users to write applications that easily work across a wide variety of hardware and south-bound protocols is provided by OpenDaylight. The OpenDaylight Controller is able to deploy in a variety of production network environments. Upcoming protocols and other SDN standards are supported by this modular controller. The OpenDaylight Controller show open northbound APIs, which are used by applications.

The Controller is used by the applications to collect information about the network and then algorithms are run to conduct analytics, and then again make use of OpenDaylight Controller to create new rules throughout the network. Within its own Java Virtual Machine (JVM) OpenDaylight is kept and implemented singly in software.

5. CONCLUSION

Due to the highly management of traffic in networks provided by SDN technology, more bandwidth is available to the users. No more dependency is there on dedicated hardware which is a cost effective way too. An abstracted view of network is provided. SDN is considered to be the best solution for meeting the new demands in networking. As SDN is an emerging technology so, research is still going on in order to make it more efficient way of networking. It is hoped that introduction about SDN its architecture and Controllers discussed here will prove to be helpful for the researchers working in this area.

6. ACKNOWLEDGEMENT

We would like to thank almighty for his constant blessings. Then we would like to dedicate our gratitude towards parents, teachers, family, friends, and in essence, all sentient one beings.

7. REFERENCES

[1] T. Galinac Grbac, C.M. Caba, J. Soler. “Software Defined Networking Demands on Software Technologies”, University of Rijeka/Faculty of Engineering, Rijeka, Croatia.
[2] Sumit Badotra, Japinder Singh, “A Review Paper on Software Defined Networking”, Volume 8, No. 3, March – April 2017, International Journal of Advanced Research in Computer Science.
[3] Alexander Gelberger, Nev Yemini, Ran Giladi, “PerformanceAnalysisofSoftware-DefinedNetworking(SDN)”, 2013 IEEE 21st International Symposium on Modelling, Analysis & Simulation of Computer and Telecommunication Systems.
[4] Jammal, Manar, Taranpreet Singh, Abdallah Shami, Rasool Asal, Yimming Li.” Software Defined Networking: State of the Art and Research Challenges “Elsevier Computer networks 72(2014)74-98
[5] Wickboldt, Juliano Araujo, Wanderson Paim de Jesus, Pedro Heleno Isolani, Cristiano Bonato Both, Juergen Rochol, and Lisandro Zambenedetti Granville “Software-Defined Networking: Management Requirements and Challenges “, IEEE Communications Magazine, January 2015.
[6] N. Feamster, J. Rexford, and E. Zegura, “The Road to SDN: An Intellectual History of Programmable Networks,” SIGCOMM Comput. commune. Rev., vol. 44, no.2, Apr. 2014, pp. 87–98.
[7] Kim, H. and N. Feamster, “Improving Network Management with Software Defined Networking,” IEEE Commune. Mag., vol. 51, no. 2, Feb. 2013, pp. 114–19.
[8] Wenfeng, Xia, Yonggang Wen, Senior Member, IEEE, Chuan Heng Foh, Senior Member, IEEE, Dusit Niyato, Member, IEEE, and Haiyong Xie, Member, IEEE.” A Survey on Software Defined Networking.”
[9] Jarschel, Michael, Thomas Zinner, Tobias Hoßfeld, Phuoc TranGia, and Wolfgang Kellerer.” Interfaces, Attributes, and Use Cases: A Compass for SDN”. IEEE Communications Magazine, June 2014.
[10] Nick, McKeown, Tom Anderson, Hari Balakrishnan Guru Parulkar, Larry Peterson, Jennifer Rexford. Scott Shenker. Jonathan Turner “OpenFlow: Enabling Innovation in Campus Networks”, March 14, 2008.
[11] OpenFlow Switch Specification, March 26, 2015.
[12] OpenFlow, http://www.openflow.org/
[13] Python at https://www.python.org/
[14] Ryu at https://osrg.github.io/ryu/
[15] Trema at htttps://github.com/trema/trema
[16] Floodlight http://www.projectfloodlight.org/floodlight/
[17] OpenDaylight at http://www.opendaylight.org/
[18] Figuerola, A.;” OpenDaylight as a Controller for Software Defined Networking” Spring 2014/2015.