ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

IJESRT
INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

SOFTWARE DEFINED NETWORKS (SDN): APPROACHES NEEDED FOR UP-GRADATION OF SDN'S

Deepak Kumar*, Manu Sood
[*] Department of Computer Science Himachal Pradesh University, Summer Hill, Shimla
Department of Computer Science Himachal Pradesh University, Summer Hill, Shimla

DOI: 10.5281/zenodo.322481

ABSTRACT

SDN has changed the way of thinking about networks. SDN networks are flexible, scalable and easily manageable. Application that runs on the management plane actually utilize the network efficiency, So application developed should be capable of handling the controllers functionality. Each of the application that runs should be free from viruses, so that it does not affect the controller. Also we need to increase the level of security in all layers of the SDN architecture. SDN is growing faster and faster and it has reached the platform where we can also apply other fields concepts. Now SDN researcher are looking for the overcome the existing drawback. So that it can even support heterogeneous networks in a very smooth and flexible way.

KEYWORDS: SDN, OpenFlow, Resilience, Security.

INTRODUCTION

Software Defined Networks (SDN) is the latest trend in networking field. It has the ability to solve many of the unsolved problems of traditional networks. In order to make changes in traditional networks all the hardware devices and software components needs to be configured individually. Due to absence of global view as in traditional networks; it takes huge amount of time to make any changes or to troubleshoot the network. With the emergence of cloud and virtualization services are the point of interest for networking industries. At the same time there is a lack of flexibility and adaptive nature in the current networking scenario. SDN [1] addresses all of the issues of traditional networks e.g. poor scalability, complexity and vendor dependence etc. The main concept that has made SDN successful is the separation of the control plane from data plane and to put the whole intelligence in the centralized controller.

By doing so we can get benefits like flexibility and the global view of the whole network (as shown in Figure1). Global view of the network helps us in easy management of the network system. In SDN architecture, as shown in **Figure2** the **bottom** layer is known as the data plane, which constitutes of the hardware devices like switches, routers and access points etc. These are the network elements which are responsible for the flow of packets in the data plane. The **middle** layer commonly known as the control plane, it is the place where the controller resides and performs the actual logic operations. All of the rules and policies are decided by the controller and are implemented in the data plane through programmatic control. At present there are wide choices of controllers available like NOX [3], POX [4], Beacon [5], RYU [6], MUL [7] and Floodlight [8] as shown in table 1.

Table 1: Controllers

Serial No.	**Controllers**	**Platform**	**Open Source**
1.	NOX	C^{++}/ Python	Yes
2.	POX	Python	Yes
3.	Beacon	Java	Yes
4.	Ryu	Python	Yes
5.	Mul	C	Yes
6.	Floodlight	Java	Yes

The **topmost** layer is called as management plane or application layer (where all applications runs), and these applications interacts with the controller through south bound interface and performs the actual management task related to network services. The OpenFlow is a protocol that acts as a interface between the control plane and data plane in SDN architecture. The decoupling of data and control plane becomes possible; only because of the OpenFlow protocol. It provides the direct access for the management of hardware devices of the data plane. OpenFlow based SDN helps us to address various parameters [9] like higher bandwidth, lively nature of the applications and to adapt the network for the dynamic business needs. In this paper we will discuss the architectural SDN components in section 2nd, In section 3rd we are discussing OpenFlow evolution, In 4th section we are discussing controller components, In section 5 we explained the need of distributed controller in SDN, In 6th section we are discussing SDN resiliency. In section 7 we will discuss the SDN for wireless mesh networks. In section 8 we are discussing SDN security levels, In section 9 we are finally concluding and the section 10 includes future scope.

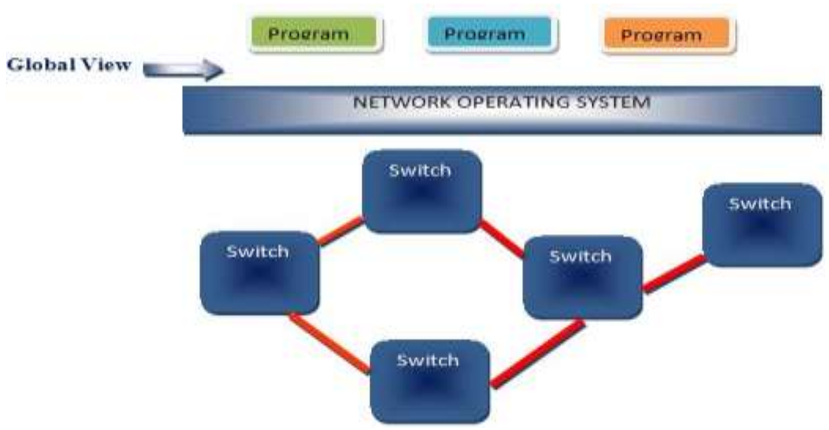

Figure1: Global view of the network

SDN ARCHITECTURE COMPONENTS

Architectural components [10] of SDN are as follows:

I). SDN Application: SDN Applications are the programs that does the management task on the controller to configures the network elements in data lane. Usually SDN application consists of one application logic and one or many North Bound Interface (NBI) drivers.

II). SDN Controller: The controller in control plane is a logically centralized and is able to receive the requirements from the SDN application layer and bring it down to the SDN data plane and therefore providing the SDN applications with an abstract view of the whole network. SDN controller consists of North Bound Interface (NBI) agents (which can be one or more), control logic and the Control Data Plane Interface (CDPI) driver.

III). SDN Datapath: It is the component of the data plane and is a logical part of the network device, which uncover visibility and are incomparable in its control over forwarding and the data processing capabilities. An SDN Datapath consists of components like CDPI agent, set of one or more traffic forwarding engines and zero or more traffic operating functions.

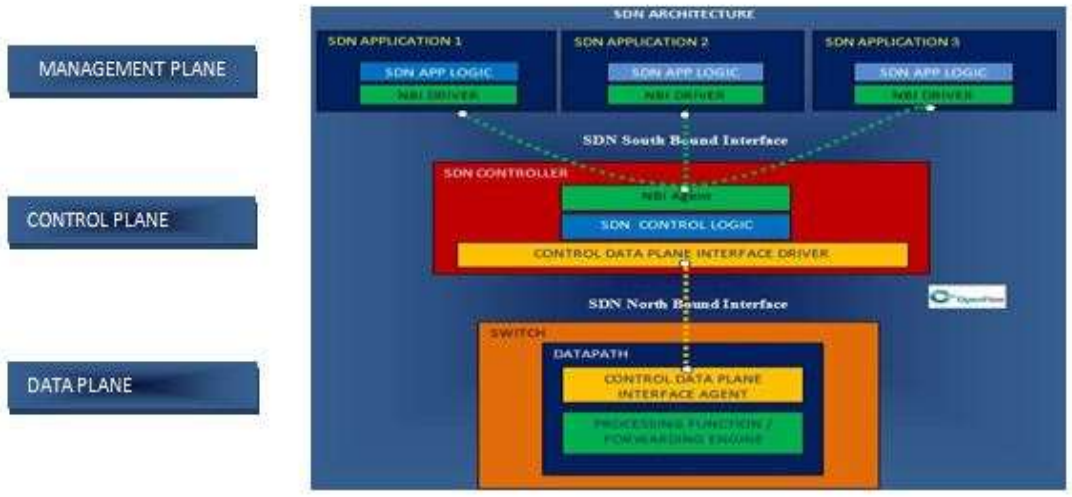

Figure2: Architecture components of SDN

SDN Control to Data-Plane Interface (CDPI) : It is the interface through which control logic implements rules and policies or drivers which are captured by the control data plane interface agents in the data plane.
V). SDN Northbound Interface: OpenFlow is a protocol that acts as the north bound Interface, enabling the communication between the components of the data plane and control plane.

OPENFLOW EVOLUTION

As SDN is growing, it has enclosed various software elements and protocols. OpenFlow is the primary protocol involved in the evolution of SDN standard. In last few years researchers has focussed on the current state of the OpenFlow in SDN framework. The evolvement of SDN take place with the development of OpenFlow in the year 2008-09 [11]. The first version of OpenFlow was released in the year 2009, and continuous work has been made on OpenFlow, soon the new updated version was released in 2012. Open Networking Foundation is focussed on the development of SDN and usually manages the OpenFlow standard. The OpenFlow protocol is firstly proposed by the N. McKeown with the intention of performing network experiments in a campus environment [12]. During the development of OpenFlow initial aim was of creating the programmable network which is controlled via centralized controller, after some researches it was discovered that the aim of controlling the network through programmatic control is actually a way that enables the development of the field; so called as the Software Defined Networking. OpenFlow supports the modern switches, routers etc as these contain the flow tables which are important for the networking function such as; stats of packet flow, routing and sub-netting etc. Each entry of the packet in the flow table consist of three main fields: **1st** is **"header"** the main function of the header field is to match the received packet. **2nd** field is **"action"** this specifies what to do for the matched packet and **3rd** is **"statistics"** for those packet which are matched. OpenFlow protocol also offers services for the manipulation of flow table for a intelligent controller to insert, delete or to modify and remotely check the flow table entries through a secure TCP channel. OpenFlow supports three message types; **Asynchronous**, **symmetric** and between the **controller** to **switch [13]**. **Asynchronous** type of message is initially sent by the switch, the purpose of this message is to modify the switch state or to update the controller for the network events. **Controllers to switch message is initiated by the controller and are** used directly to inspect or manage the switch. **Symmetric messages** can be initiated either by the switch or controller.

CONTROLLER COMPONENTS

We have wide variety of controllers available. The main components of controllers are as shown in Figure 3.
I). Network Layer: The main purpose of the **network layer [13]** is to establish the communication with switching devices. It is the most inner or bottom layer of each of the controller to determine its performance. The main task of the **network layer** is:
1). To read the incoming OpenFlow messages from the channel. The network layer is dependent on the runtime of programming language chosen. To establish the faster communication with devices like network interface card (nic), we can use fast packet processing framework such as netmap [14] etc.

II). OpenFlow Library: The main functionality of the OpenFlow library is to check the correctness and provides the parsing to the OpenFlow messages, and depending upon the type of the packet it can produce new events such as port_status and packet_in etc.

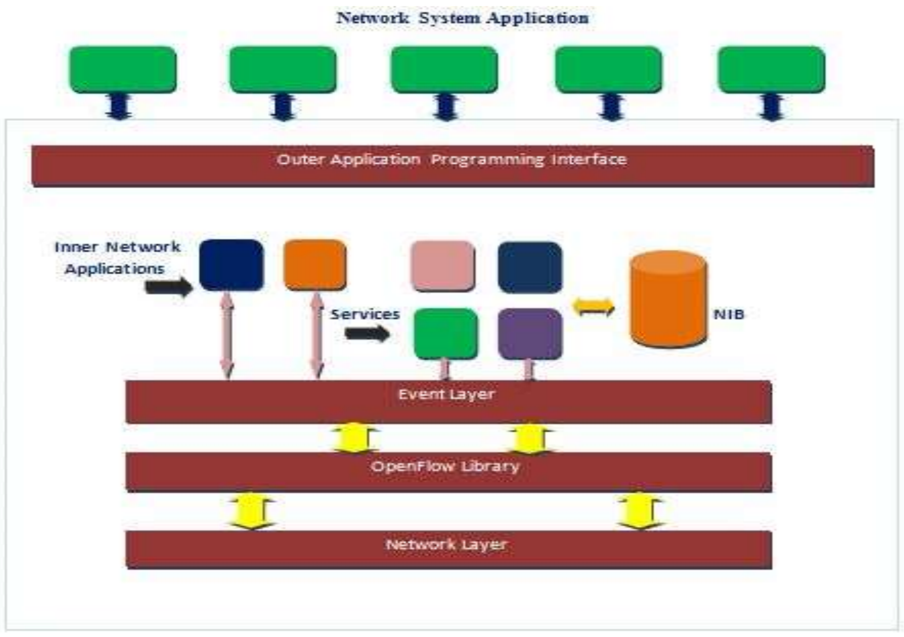
Network System Application
Outer Application Programming Interface
Inner Network Applications
Services
NIB
Event Layer
OpenFlow Library
Network Layer

Figure 3: Controllers inbuilt components

III). Event Layer: The main work of this layer is the handling of events between the core, the services and the internal network applications. The network applications subscribes the events from the core or bottom layer and which produces other events to which other applications may or may not be subscribed. This is usually done with the mechanism known as subscription mechanism.

NEED OF DISTRIBUTED CONTROLLER IN SDN

Only a single controller is not enough for the management of whole network system. There are two measure issues:-

Less Reliable:- The centralizing controller is the only point of interest for attackers. It is the only single point of failure. If controller fails, the whole network system fails.

Scalability Issue:- Today's networks are growing at a very fast rate. Resources provided by the current controller are limited and are not capable of handling or to maintain the states of the all of the network elements. As networks become wider, so latency also increases.

In order to solve the scalability and reliability issue, there is a need of more than one distributed controllers. In this the network is divided into various sectors, each of which is controlled by the individual dedicated controller. Network sectors may overlap to certify the flexibility of network in case if any of the controller stops working. Each of the controllers is connected with distributed data storage to provide global view of the network. The main purpose of this data storage is to store all the information regarding all of the applications and hardware devices. State of each of the application is kept under distributed data storage just to promote the migration between the switches and for the recovery of controller failure. There are many unanswered research questions such as; how can we reduce the overhead in distributed data storage, how the migration between switches takes place, how can we run applications on distributed controllers and what is the best controllers place etc.

SDN RESILIENCY

In the current network scenario whenever the logic of switch fails, only the traffic that passing through it is affected. The neighbouring switches are preinstalled with the backup path whenever any failover occurs. When there is a detection of failure optional backup paths are activated. If control logic in SDN fails, then the forwarding elements are in down mode; resulting in to the drop of packet, or undelivered data packets. How strong the topology is, this can be measured on terms of connection in a network after removing the hardware elements and links. The keyword resiliency means that the ability to redistribute the optional paths with in the specified time period. The resistance [15] of the individual or group of distributed controller (s) can be defined in terms of their robustness in handling of faults in controller before it fails. Resilience also means the ability to recover the prestate control program as soon as possible after failure occurs. From the point of view of the designer the definitions for the resilience and robustness can be different. In the case of having a single controller and if it fails, then result in to un-controllable behaviour of the network.

Earlier detection of fault and faster failure recovery are the primary requirement of an efficient network system. Therefore by using the distributed set of controllers and dividing the work load across them, we can create a robust network system, capable of handling failures.

SDN FOR WIRELESS MESH NETWORK

The disposition of OpenFlow based SDN in a Wireless Mesh Network (WMN) presents some problems such as to setup a vigorous control framework that enables the network element to communicate with the controller, and also to face the worst conditions like unavailability of the controller due to the partitioning of the network or a complete controller failure. In WMN the routing mechanism is not based on the Spanning Tree algorithm but is based on the routing protocols like OLSR. WMN consists wireless mesh routers, whose purpose is to provide connectivity to the set of access networks through wired or wireless interface. Each of the subset of WMR can be operated as gateways and provides connectivity to the internet. In OpenFlow based SDN controller is connected to WMR via wired or wireless connection.

SDN SECURITY LEVEL

One of the purpose to provide security in network is to provide the controlled network access, partition between the users and to protect the network system from attackers or intruders etc. As SDN is a new demanding networking technology, it is the point of interest for researchers, and it is expected that applications which provides network security can be easily applied to the networks control logic. There are mainly **two** levels of security**:-**

In the **first** level of the security; it requests for the logical connections among the hosts within the network. The **SSL** (Secure Socket Layer) protocol and the technique which is used for the packet encryption are used to make sure that connection must be secured. In SDN the standard protocol i.e. OpenFlow which has the inbuilt feature to provide secure network connection. It is the responsibility of the controller to provide the secure connection with the network elements. In the distributed controller there is no such mechanism for the link security. When there is a lack of the link security, then the malicious code across any node can take the whole control of the switches.

The main function of the **second** level of security; is to protect servers, end hosts in the network and the network elements. Any malicious software can interrupt the whole network system, can collect secret information and can also infect the hosts. If DOS (Denial of Service) attack made by the attackers then it can disable the whole network system or it can also overload the network elements and controller. Securities mechanism such as; firewall to find out the malicious packet flow and to block or re-route traffic.

One of the approaches such as Net-Watch is firewall component for OpenFlow to restrict the fake packet flow. It mainly includes two types of functionality; **First:** Policy file includes the IP address or ports. It allows only those port and IP addresses, which are already specified in order to transmit packets**. Second:** Those packets which are specified in the rules are only allowed for the transmission.

CONCLUSION

SDN has brought a lot change in the networking industry, furthermore is yet to come. Researchers are doing their best to utilize SDN features. There is a need of increasing security level across all the three layers in the SDN architecture. By providing security at each layer we can minimize security related problems. Also, the choice of using multiple controllers makes the SDN networks much more reliable and capable of handling multiple faults at the same time. Also the use of multiple controllers makes it much more scalable. Also there is a need of modification in the networking elements of data plane in SDN so that they become resistant to any type of attack. If one of the network elements fails, other elements should have back up path, which makes network flexible enough to handle fault. The concept of the SDN is applicable for mobile networks, data-centres and wireless mesh networks. SDN offers wide variety of solutions to network problems, so there is a need of SDN like networking environment.

FUTURE SCOPE

There is a wide scope of future in SDN. It is the future of networks. It can easily manage the whole network through programmatic control. Technology in actual never becomes outdated, if it becomes, then it act as a bases for the new updated technology. SDN is newly established networking platform, each day there is research going on, in current SDN there are many breaches, so there is wide scope of improvement and research. We can apply concept of the other fields in SDN like software engineering, databases etc. So this makes SDN much more interesting for researcher.

REFERENCES

[1] [ONLINE] Article available at: https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/SDN-architecture-overview-1. 0.pdf (Figure1)66
[2] Gude, N., Koponen, T., Pettit, J., Pfaff, B., Casado, M., McKeown, N., and Shenker, S. NOX: towards an operating system for networks. SIGCOMM Computer Communication Review 38, 3 (2008), 105-110.
[3] Pox documentation, http://www.noxrepo.org/pox/about-pox/
[4] D. Erickson, The Beacon OpenFlow controller. *In Proc. HotSDN* 2013.
[5] Ryu documentation, http://osrg.github.com/ryu/
[6] Mul documentation, http://sourceforge.net/p/mul/wiki/Home/
[7] Floodlight documentation, http://floodlight.openflowhub.org/
[8] https://www.opennetworking.org/sdn-resources/openflow
[9] [ONLINE] Article available at: https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/SDN-architecture-overview-1.0.pdf
[10] [ONLINE] Article available at: https://www.opennetworking.org/images/stories/downloads/sdn-resources/special-reports/Special-Report-OpenFlow-and-SDN-State-of-the-Union-B.pdf
[11] W. Xia, Y. Wen, C.H. Foh, D. Niyato, H. Xie, A Survey on Software Defined Networking. *IEEE COMMUNICATION SURVEYS & TUTORIALS, VOL. 17, NO. 1*
[12] A. Shalimov, R. Smeliansky, On Bringing Software Engineering to Computer Networks with Software Defined Networking. Applied *Research Centre for Computer Networks, Moscow State University.*
[13] L. Rizzo, netmap: a novel framework for fast packet I/O, *Usenix ATC'12, June 2012*
[14] B.J.V. Asten, N.L.M. Van Adrichem, F.A. Kuipers. Scalability and Resilience of Software Defined Networking: An
[15] Overview, Network Architectures and Services, Delft University of Technology *Mekelweg 4, 2628 CD Delft, The Netherlands.*