International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 04 Issue: 07 | July -2020 ISSN: 2582-3930

A Survey on Software Defined Networking

Juhika Azmeen[1], Mr. N.Rana Singha[2]

[1]PG Student, Department of Information Technology, Kaziranga University, Assam, India
[2]Assistant Professor, Department of Information Technology, Kaziranga University, Assam, India

---

Abstract -

Software Defined Networking (SDN) came into picture from the last couple of years, decouples the network control and forwarding functions. It enabled the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services. The controller is now responsible for making the decisions which is a major advantage for the Software Defined Networks (SDN). Here, we will make a general survey on Software Defined Networks (SDN) considering the following parameters such as scalability, reliability, fault tolerance, high availability and security. The survey aims to rate most of the solutions that came up so far and the end of the survey a proposed solution for the load balancing of Software Defined Networking (SDN) factor in cellular network has been driven out.

Key Words: SDN , load-balancing, Mininet

1. INTRODUCTION

Since networks have been increasing a lot in size and in requirements, moving around hardware switches has become a burden. Even manually setting up each individual switch has become an error prone ad hectic task which needs a lot of man power and such manual configuration for large scale companies with a strong virtualized network is a difficult task. This is where the need of Software Defined Networking came into being in the early 1990s.

"SDN" is a like "cloud computing": It is a trendy topic in computing, and is often considered as a solution to the traditional network infrastructure system. But beside marketing, it also corresponds to a new type of network architecture. "Software defined" means that the whole network is programmable and it doesn't only uses virtual switches.

Software-Defined Networking (SDN) is an emerging architecture that decouples the network control and forwarding functions enabling the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services. SDN is cost effective, centralized administration and is highly manageable.

1.1. Purpose:

To make a general survey on Software Defined Networks (SDN) solutions implementations and management's frameworks that has been proposed so far by considering the following five parameters such as scalability, reliability, fault tolerance, high availability and security in hand.

1.2. Technologies Used:

a) Mininet
b) Python
c) AWS
d) Network Operating System (Controller):

2. SDN Architecture and protocol

Software-Defined Networking (SDN) is an emerging architecture that decouples the network control and forwarding functions enabling the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services. SDN is cost effective, centralized administration and is highly manageable. The OpenFlow protocol is the main building block of the SDN. The core components that contribute to the SDN architecture are as follows:

a) Forwarding Devices (FD): These are hardware- or software-based data plane devices that perform a set of operations. The forwarding devices have well-defined instruction sets (flow rules) used to take actions on the incoming packets (e.g., forward to specific ports, drop, forward to the controller, rewrite some header).

b) b) Data Plane (DP): The data plane (or forwarding plane) consists of the basic forwarding elements like the switches or routers and is the high-speed path in SDN. Packets that pass through the device use the data plane, as opposed to packets directed to the device. Therefore, the data plane is also called the forwarding plane.

c) c) Southbound Interface (SI): It defines the communication protocol between forwarding devices and control plane elements. This Southbound Interface protocol formalizes the way the control and data plane elements interact.

d)Control Plane (CP): The forwarding devices such as switches and routers are programmed by control plane elements through well-defined SI embodiments. The control plane is referred to as the" network brain" in the SDN architecture and all the control logic lies in the applications and controllers that contribute to form the control plane..

e)Northbound Interface (NI): The Northbound Interfaces offers API to application developers. This API represent a common interface for developing applications. Typically, a northbound interface abstracts the low-level instruction sets used by southbound inter-faces to program forwarding devices..

f)Management Plane (MP): The management plane is the set of applications that leverage the functions offered by the NI to implement network control and operation logic. This includes applications such as routing, fire-walls, load balancers, monitoring. A management application defines the policies, which are ultimately translated to southbound-specific instructions that program the behavior of the forwarding devices.

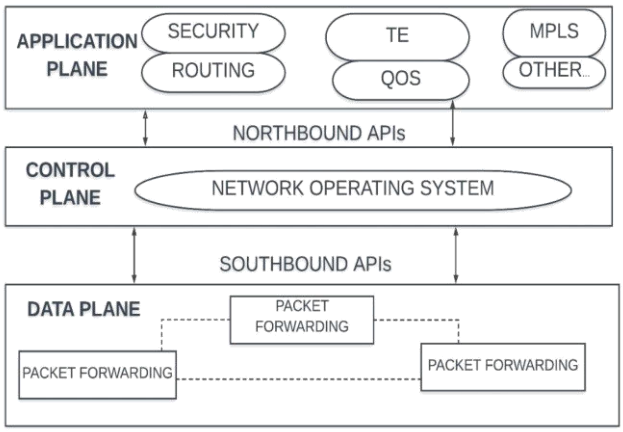

Fig -1: SDN Architecture

2.1: SDN Architecture Benefits:

By using SDN one can have several benefits such as:

- Intelligent routing decisions.
- Simplification of information collection, analysis and decision making.
- Visibility of network resources-network management is simplified based on user, device and application specific requirements.
- Intelligent traffic pattern analysis and coordinated decisions.

2.2. SDN Overview

The term SDN was originated at Stanford University, Stanford, CA, USA. SDN refers to a network architecture where the forwarding state in the data plane is managed by a remotely controlled plane de-coupled also known as the controller.

A SDN architecture can be described as a composition of different layers and each layer has its own specific functions while some of them are always present in an SDN deployment, such as the southbound API, NOSs, northbound API, and network applications, others may be present only in particular deployments, such as hypervisor- or language-based virtualization.

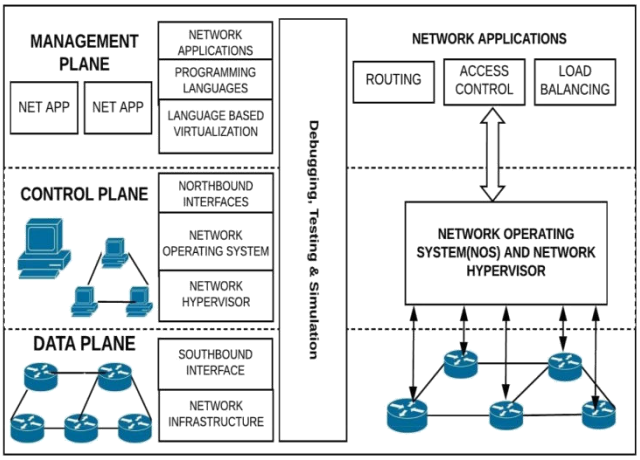

Fig -2: Software-Defined Networks in (a) planes, (b) layers, and (c) system design architecture.

Fig. 2 (a) presents a trifold perspective of SDNs. The SDN layers are represented in Fig. 3.1(b), as explained above. Fig. 3.1(a) and 3.1(c) depicts a plane-oriented view and a system design perspective, respectively.

Layer I: Infrastructure

SDN infrastructure is to a traditional network and is composed of a set of networking equipment (switches, routers, and middle box appliances and the difference is that the whole network is programmable and is decoupled. The control unit is removed from the data plane and the whole network is controlled by a device logically centralized control system.

Layer II: Southbound Interfaces

The connecting bridges between control and forwarding elements are known as the Southbound APIs. These APIs are tightly tied to the forwarding elements of the underlying physical or virtual infrastructure. For

commercialization typically, a new switch can take two years to be ready if built from scratch and with upgrade cycles that can take up to nine months. And also for a new product the software development can take from six months. To one year and the initial investment is considered to be high and risky. The southbound APIs represent one of the major barriers for the introduction and acceptance of any new networking technology.

Layer III: Network Hypervisors

In modern computers virtualization is already a consolidated technology. Virtualization has been the made computing platforms mainstream due to the fast developments of the past decade .The number of virtual servers has already exceeded the number of physical servers based on recent reports.The main function of a hypervisor is that it enables distinct virtual machines to share the same hardware resources. Each user can have its own virtual resources in a cloud infrastructure-as-a-service (IaaS), from computing to storage. This enabled new revenue and business models where users allocate re-sources on demand, from a shared physical infrastructure, at a relatively low cost.

Layer IV: Network Operating Systems/Controllers

For accessing lower level devices, manage the concurrent access to the under-lying resources (e.g., hard drive, network adapter, CPU, memory) the traditional operating systems provide abstractions (e.g., high-level programming APIs) and provide security protection mechanisms. These functionalities help in increased productivity, making the life of system and application developers very easier. This has significantly led to the evolution of various ecosystems (e.g., programming languages) and also for the development of a myriad of applications.

Layer V: Northbound Interfaces

The two key abstractions of the SDN ecosystem are the northbound and southbound interfaces. The southbound interface has already a widely accepted proposal (OpenFlow), but a common northbound interface is still an open issue. In these ecosystems, the implementation is commonly the forefront driver, while standards emerge later and are essentially driven by wide adoption.

Layer VI: Language-Based Virtualization

The capability of expressing modularity and of allowing different levels of abstractions while still guaranteeing de-sired properties such as protection are the two essential characteristics of virtualization solutions. The virtualization techniques may allow different views of a single physical infrastructure. A virtual ‘‘big switch’’ could represent a combination of several underlying forwarding devices can be taken as an example.

VII: Programming Languages

For decades programming languages have been proliferating. From low-level hardware-specific machine languages, such as assembly for x86 architectures, to high-level and powerful programming languages such as Java and Python both academia and industry has been evolved.

3. LITERATURE REVIEW:

In [1] existing cellular networks suffer from vendor specific configuration of switches or routers, lacks centralized administration and manual configuration. In this paper the author argues that SDN can solve the issues that are faced on the existing cellular network architecture however proposing several extensions to the controller platforms, base stations and switches so that it can help in a) Apply real-time fine-grained control through local agents on the switches. b) Perform deep packet inspection. c) Remotely manage shares of base station.

Conclusion: The author sketches out the changes in the existing architecture however the proposed architecture for the SDN cellular network architecture has been left out for future work.

In [2] paper the author highlights a deep survey of the SDN-Iot solutions over the time period 2010-2016 by focusing on different parameters such as scalability, security, limitations and benefits.

Conclusion**:** Several solutions and papers of SDN-IoT have come up so far after 2016 which has not been defined.

In [3] the author divided the whole cellular network into clusters and the whole clusters are controlled by separate controller. The CCC (controller controller communication) is the main component of the proposed architecture which is responsible for maintaining the status of the controllers. Whenever a cluster is heavily loaded it send the packet to the OFSW which is common between the two clusters. The OFSW with the help of CCC ensures that the traffic is forwarded to another less crowded cluster so that the controller overwhelming is minimized.

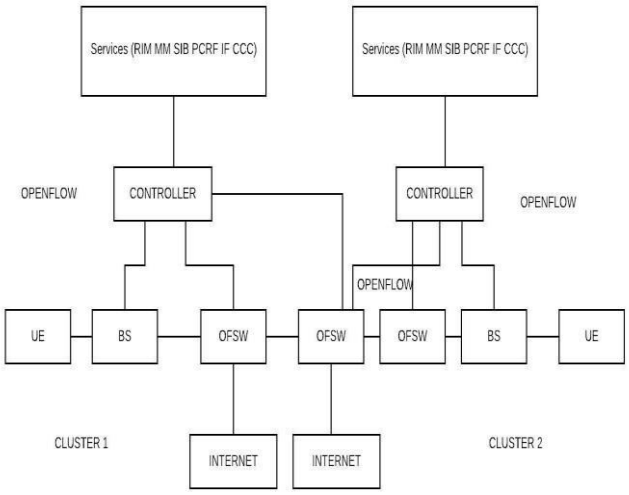

Fig-3: A novel SDN-Architecture

In [4] the author describes that sensor networks are typically purpose-built, designed to support a single running application. However, there might be a scenario where we want the same architecture to be shared among different sensing applications WSN integration in SDN is seen with a three-layer architecture. It consists of master node/controller node, central node (OpenFlow enabled switch) and a normal node. The master node defines a routing policy for the normal node. The author et.al uses flow Visor as virtualization engine to make independent user slice in between controller and switch. Data forwarding is done by OpenFlow switch. The configuration and management are done by OpenFlow protocol, which also identifies the existing routing protocol, and work in congruence. The placement of central node is important. In this proposal, the distance is calculated based on cosine similarity formula. The central node locates in the physical center of the cluster architecture, helps in maintaining network topology and help increasing network convergence. They name their architecture as SDWSN. Neighboring node status is taken into account for the assigning role. Even though the author has verified their architecture through comparing its result with existing WSN protocol and find improvement in the result; the conceptual details are not very clear.

A number of sensing applications were deployed including applications for monitoring office, occupancy and room environmental conditions however the performance seemed to be low, overall cost is reduced.

In [5] the author describes that traditional IP networks are very hard to configure and manage and are difficult to configure the network according to predefined policies, and to reconfigure it to respond to faults, load, and changes. It makes it more difficult as in the current network the control plane and the data plane are coupled together. Software-defined networking (SDN) the whole network is rebuilt and the switches and routers controlling and forwarding unit is separated to make the network directly programmable. This paper performs a deep survey on SDN, how it is different from the traditional network architecture and the benefits of SDN, challenges of SDN and the future aspects of SDN.

A SURVEY ON SOFTWARE DEFINED NETWORKING:

In this section survey of few of the papers have been done based on scalability and the main purpose of it.

Architecture	Virtua Management	Interface API	Virtualization	Control/Data Plane	Traffic Engineering	Scalability (High)	Scalability (Low)
SoftRAN	Resource Management, Mobility Support and Traffic of loading	Controller API/Femto API	Big Base Station	Centralized Controller & Local Agent at eNBs Slicing forming Bit Station	Load balancing Interface Management	High	
Softcell	Fine grain policies management	OpenFlow API	Minimum virtualization	Logically centralized controller, local agent, SD-RAN(BS)	MPLS and slandered routing as in OpenFlow		Low
SoftAir	Distributed traffic classification, network management	OpenFlow and CPRI	Fine grain Virtualization	SD-BS, SD-Switch, BS-clustering	Collaborative processing, scheduling and mobility management	High	

Fig -3.1 Survey on SDN-I

Name of the paper	Interface and benefits	Control /Data Plame	Scalability (High)	Scalability (Low)
Towards-Software Defined Cellular Networks	Openflow API • Perform deep packet inspection • Centralized administration	Centralized controller	√ High	
Software Defined Network (SDN) Based Internet of Things (IOT): A road Ahead	A deep survey on SDN upto 2016		√ High	
SenShare: Transforming Sensor Networks into Multi-Application Sensing Infrastructures	Deployment of Senshare where the same sensor network is shared among multiple sensing application with the integration of SDN			√ Low
A Novel Architecture for SDN based cellular network	Load balancing between two different clusters	SDN Controller with Open Flow Enabled Switch		√ Low

Fig -3.2: A survey on SDN II

4. PROPOSED ARCHITECTURE:

In this architecture the whole cellular network into clusters and the whole clusters are controlled by separate controller. The CCC (controller controller communication) is the main component of the proposed architecture which is responsible for maintaining the status of the controllers. So, whenever a cluster is heavily loaded instead of sending a packet to the CCC to ask about the status of the controller to the cluster in which it resides to as in paper "A Novel Architecture for SDN based cellular network". Here, both the CCC can communicate with each other regarding their status and hence the extra time is reduced and the CCC is used efficiently.

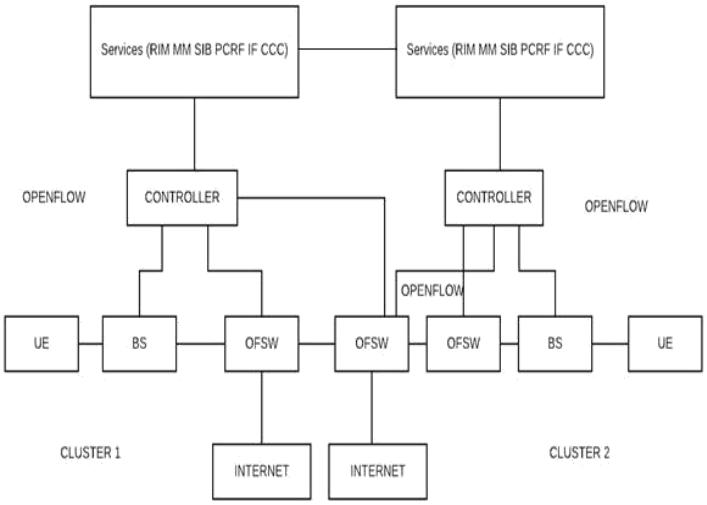

Fig -4: Proposed Architecture

5. RESULTS AND DISCUSSION

Software-Defined Networking (SDN) is an architecture that decouples the network controlling and forwarding functions enabling the network control. SDN is cost effective than the traditional network architecture. The proposed architecture is implemented on Mininet by creating AWS instances. Ryu has been installed in both the clusters and whenever a cluster is overloaded the traffic is directed to the cluster with the controller that has less traffic. In this paper, a survey on few of the papers has been done based on scalability reliability, performance and the proposed architecture for the load balancing scenario in SDN architecture has been implemented. However, the communication between the CCC in the proposed architecture and the time taken by them to share the status has been left for future work.

6. CONCLUSIONS

Since networks have been increasing a lot in size and in requirements, moving around hardware switches has become a burden. Even manually setting up each individual switch has become an error prone ad hectic task which needs a lot of man power and such manual configuration for large scale companies with a strong virtualized network is a difficult task. Software-Defined Networking (SDN) is an emerging architecture that decouples the network control and forwarding functions enabling the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services. SDN is cost effective then the traditional network architecture. In this paper, a survey on few of the papers has been done based on scalability reliability, performance and the proposed architecture for the load balancing scenario in SDN architecture has been implemented.

ACKNOWLEDGEMENT

I have taken efforts to present out the paper. However, it would not have been possible without the kind support and help of many individuals. I would like to extend my sincere thanks to all of them. I am highly indebted to The Assam Kaziranga University for giving me an opportunity to do this project on the topic "A survey on Software defined Networks (SDN)" which also help me in doing a lot of research and I came to know about so many new things. I would like to express my special gratitude and thanks to **Mr. N. Rana Singha** for his guidance and constant supervision as well as for providing necessary information regarding the project and also for his support in completing the project. My special thanks and appreciations to all the people in

developing the project who have willingly help me out with their abilities.

REFERENCES

[1] Li, L. E., Mao, Z. M., & Rexford, J. 2012. "Toward software-defined cellular networks. In SoftwareDefined Networking (EWSDN)", 2012 European Workshop on. (2012) 7-12.

[2] M. H. Kabir, —" A Novel Architecture for SDN-based Cellular Network", ‖ Int. J. Wirel. Mob. Networks, vol. 6, no. 6, p. 71, 2014.

[3] J. Medved, R. Varga, A. Tkacik, and K. Gray, —" Open daylight: Towards a model-driven sdn controller architecture", ‖ in Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014, 2014.

[4] kyildiz, I., Wang, P. and Lin, S. 2015. "SoftAir: A software defined networking architecture for 5G wireless systems. Computer Networks". 85, (2015), 1-18.

[5] J. Liu, S. Zhang, N. Kato, H. Ujikawa, and K. Suzuki, —" Device-to device communications for enhancing the quality of experience in software defined multi-tier LTE-A networks", ‖ IEEE Netw., vol. 29, no. 4, pp. 46– 52, 2015.

[6] J. Liu, S. Zhang, N. Kato, H. Ujikawa, and K. Suzuki, —" Device-to device communications for enhancing the quality of experience in software defined multi-tier LTE-A networks", ‖ IEEE Netw., vol. 29, no. 4, pp. 46– 52, 2015

[7] I. Leontiadis, C. Efstratiou, C. Mascolo, and J. Crowcroft, —" SenShare: transforming sensor networks into multi-application sensing infrastructures, ‖ in European Conference on Wireless Sensor Networks", 2012, pp. 65–
81

[8] Kevin´ Phemius, Mathieu Bout and Jer´emie´ Leguay, ---- "DISCO: Distributed Multi-domain SDNControllers".

[9] Amin Tootoonchian, Yashar Ganjali— "HyperFlow, a distributed event-based control plane for OpenFlow".

[10] Software-Defined Networking: A Comprehensive Survey by Diego Kreutz, Member IEEE, Fernando M. V. Ramos, Member IEEE, Paulo Esteves Verı´ssimo, Fellow IEEE, Christian Estev Rothenberg, Member IEEE, Siamak Azodolmolky, Senior Member IEEE, and Steve Uhlig, Member