ISSN: 0975-766X
CODEN: IJPTFI
Research Article

Available Online through
www.ijptonline.com

A SURVEY OF SOFTWARE DEFINED NETWORK

Nithya.S, M.Asha Jerlin, C.Jayakumar,
VIT University, Vellore, TN, India.
Email: nithya.s@vit.ac.in

Received on 25-10-2016 *Accepted on 02-11-2016*

Abstract

Software defined network (SDN) is the revolutionary technology in the Networking atmosphere. It finds its roots in programmable network and control-data-infrastructure plane separation which provides network management as well as network automation. This technology encourages programmability of the network which in turns reduces the cost of operation and increases efficiency. The objective of this paper is to study the various investigations going to understand and evaluate SDN characteristics and benefits. In this paper we will be exploring SDN architecture, SDN related taxonomy and challenges SDN implementation face now and in future.

Keywords; Software Defined Networks, Open Flow, programmable networks, management, taxonomy.

Introduction

Traditional Network technologies have existed from the inception of networking even though various modifications has been done to the underlying architecture and devices, they have processed and routed packets almost in the same format resulting to limited efficiency and high cost of maintenance. Consequence to this there was need to change the technique used in designing network architecture, thus the born of "SDN", this new technology brings a flat platform in which various vendors manufacture devices that are not proprietary in nature. Present day network architecture is made up of Control Plane, Data Plane and Management Plane where the Control and Data planes are merged into a machine generally known as "Inside the Box". To avoid these limitations a new set of networks known as "Programmable Networks" have emerged generally referred known as "Out of the box". The main aim of SDN is to separate the Control and Data Plane and to transfer the network intelligence and state to the Control Plane. Some technologies that have exploited these concepts include Routing Control Platform (**RCP**), Secure Architecture for the Network Enterprise (**SANE**) and recently Ethane. SDN is often related to the Open Flow Protocol. Currently, the

Open Networking Foundation (**ONF**) takes the task of advancing SDN and standardizing OpenFlow whose latest version is 1.5.0.

Software Defined Networking Genesis

SDN is born from the Programmable Networks and Control Data plane separation. The main objective of programmable networks is to present more flexible and dynamically customizable networks. To make this concept come to live, two major path of thought have been defined: OpenSig (Presented by the community of Telecommunication) and Active Networks (Presented by community of IP Networks). Active Networks is based on customized programs to be carried by packets and executed by network elements while OpenSig is based on controlling networks through a set of network programming interfaces and distributed programming environments.

Active Networks and OpenSig presented a concept whereby packets need to process individually by Network Node which in turn generates performance degradation. Due to this challenge, the research was slowed down because Network Devices Vendors didn't give full support. To expand the concept of control and data plane separation researchers explored Clean Slate 4D Architecture. This involves separating into two components. The first which is a controller that controls and has knowledge of the global Network Topology and second component is set of simple and dumb switches. The success of SDN can be traced to the existence of OpenFlow. OpenFlow-enabled switches contain numbers of tables containing various Packet Handling Rules.

I. SDN: DETAILED ARCHITECTURE AND RESEARCH WORK

Here we shall be presenting the underlying architecture of SDN and its principal components. The architecture presented by ONF is split into three main layers vertically.

1. Infrastructure Layer: This is commonly known as Data Plane, it comprises of the Forwarding Elements (FEs) which are physical and virtual switches.
2. Control Layer: This commonly known as the Control plane which is the brain of the network, it comprises of Software Based SDN Controllers that determine the flow path of packets.
3. Application Layer: It comprises of End User applications that consume SDN Network services.

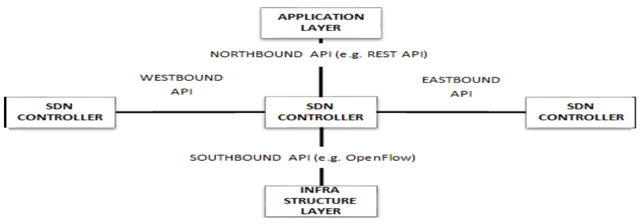


Fig.1 Layered Architecture and Interfaces of SDN [1].

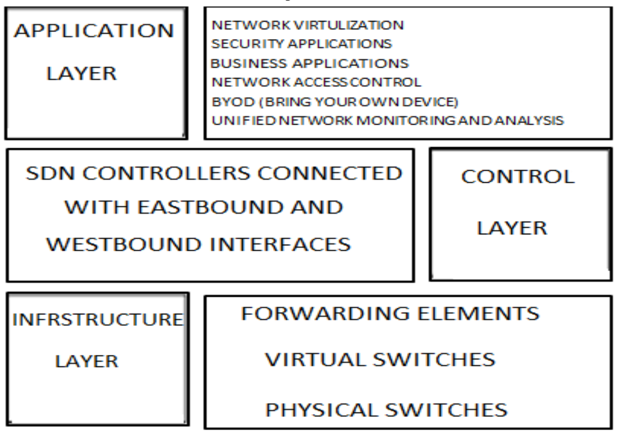


Fig. 2 Detailed Architecture of SDN [1].

Interaction between the above layers is carried out by three open interfaces as shown in Fig. 1.

- Southbound: This is the interface in charge of communication between Infrastructure and Control Layer
- Northbound: This supports the programming of the controllers.
- East/West Bound: This is in charge of communications between the controllers in the control Plane.

Looking at SDN from another point of view other logical layers have been presented for the control and data layers. These layers present simple presentation of the SDN visions. They are:

- Physical Forwarding Plane: this describes he set of physical forwarding nodes.
- Network Virtualization (or Slicing): This layer is in charge of configuration of the physical forwarding nodes.
- Logical Forwarding Plane: This provides an end-to-end forwarding model which is not dependent on the Physical Forwarding Plane.

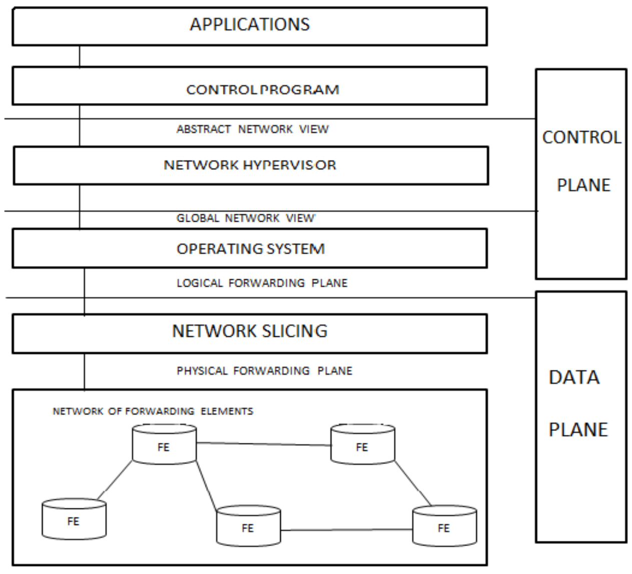


Fig.3 Abstraction of Layered SDN.

- Network Operating System: This is software which defines the operation of the network.

- Global Network View: This comprises of a pictorial graph of the network that is generated through an API.
- Network Hypervisor: This maps the Abstract network view to the global network view.
- Abstract Network View: This presents to the applications information for management policies.

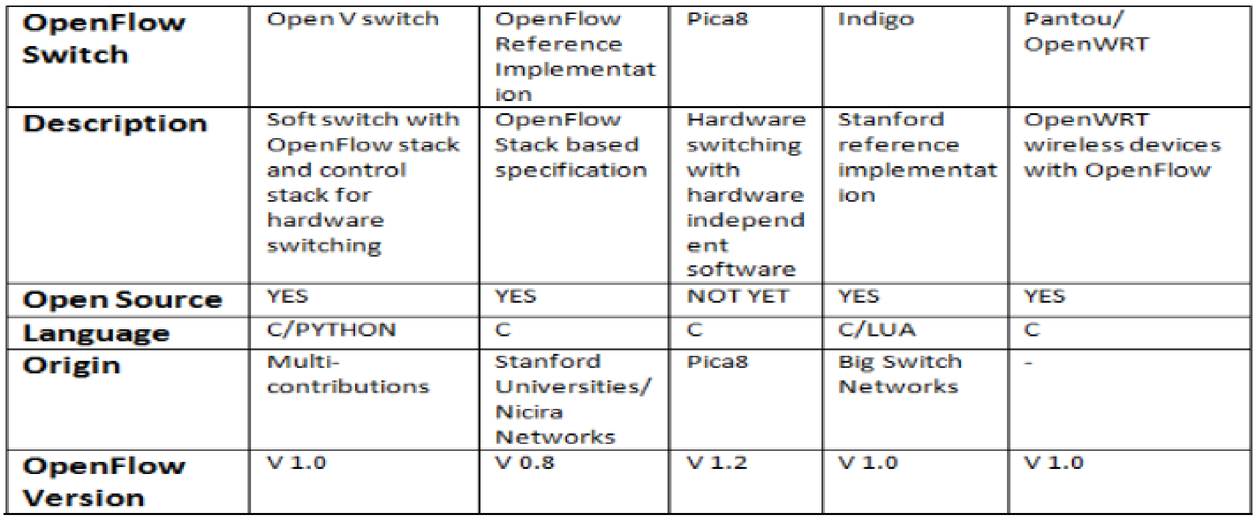
DETAILED EXPLANATION OF EACH LAYER

FORWARDING ELEMENTS:

Due to the fact that we are considering OpenFlow Protocol, we shall be looking at forwarding elements from the OpenFlow view. OpenFlow presents two types of switches: Software-based (example includes Open V Switch OVS) and Open Flow enables Hardware-Based (example include NetFPGA). Software switches come with complete features where hardware based is application specific integrated circuits (ASICs).

OpenFlow enabled switches are generally divided into hardware layer (data path), software layer (control path) and Open flow protocol.

- The data path comprises of flow tables and group table which executes the task of packet look ups and forwarding. Flow entries are filled in flow table and this defines how the switch is processed the flow. Flow table are filled by the controller. The group entries are found in group table.
- The switch is connected to the controller via the controller path for signalling and programming purposes.



OpenFlow Switch	Open V switch	OpenFlow Reference Implementation	Pica8	Indigo	Pantou/ OpenWRT
Description	Soft switch with OpenFlow stack and control stack for hardware switching	OpenFlow Stack based specification	Hardware switching with hardware independent software	Stanford reference implementation	OpenWRT wireless devices with OpenFlow
Open Source	YES	YES	NOT YET	YES	YES
Language	C/PYTHON	C	C	C/LUA	C
Origin	Multi-contributions	Stanford Universities/ Nicira Networks	Pica8	Big Switch Networks	-
OpenFlow Version	V 1.0	V 0.8	V 1.2	V 1.0	V 1.0

OpenFlow protocols are in charge of presenting a means of communication from controller to switch and vice versa. Information exchange generally is done by OpenFlow.

The flow table comprises of various fields which are listed below.

- Match fields: this is based on a 15-tuple packet header, ingress port, and not often packets Meta-data.

The fig.4 presents various packet data fields based on the OSI 1 to 4 layers.

- Counters: Its responsibly is to track numbers of packets and bytes for each flow.
- Timeout: this defines max. Amount of time in which the flow gets expired in the switch.

There exist three major messages in OpenFlow

1. Control to switch
2. Asynchronous
3. Symmetric

A. Controllers:

This is the brain of SDN network and it also serves as the main part of the network operating system. The controller's task amongst other is to install flow entries on switch nodes.

Above is the table consisting is of major existing OpenFlow controllers.

B. Programming SDN Applications

SDN Applications make use of APIs to easily configure the network and its components to pull information based on its needs. SDN programming framework comprises of programming languages and specific tools used for compiling and validating the rules (OpenFlow) that generates various applications. Comparison of SDN programming languages is based on:

1. Level of abstraction
2. Programming
3. Policy logic

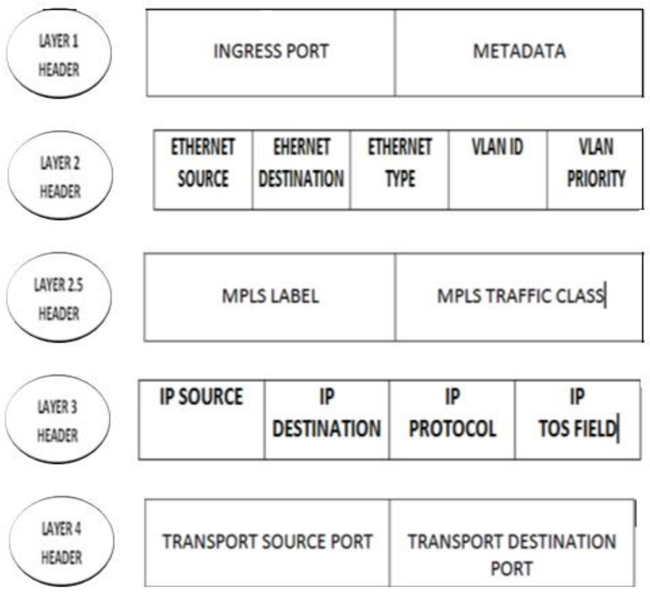


Fig.4 Identification Flow in OpenFlow.

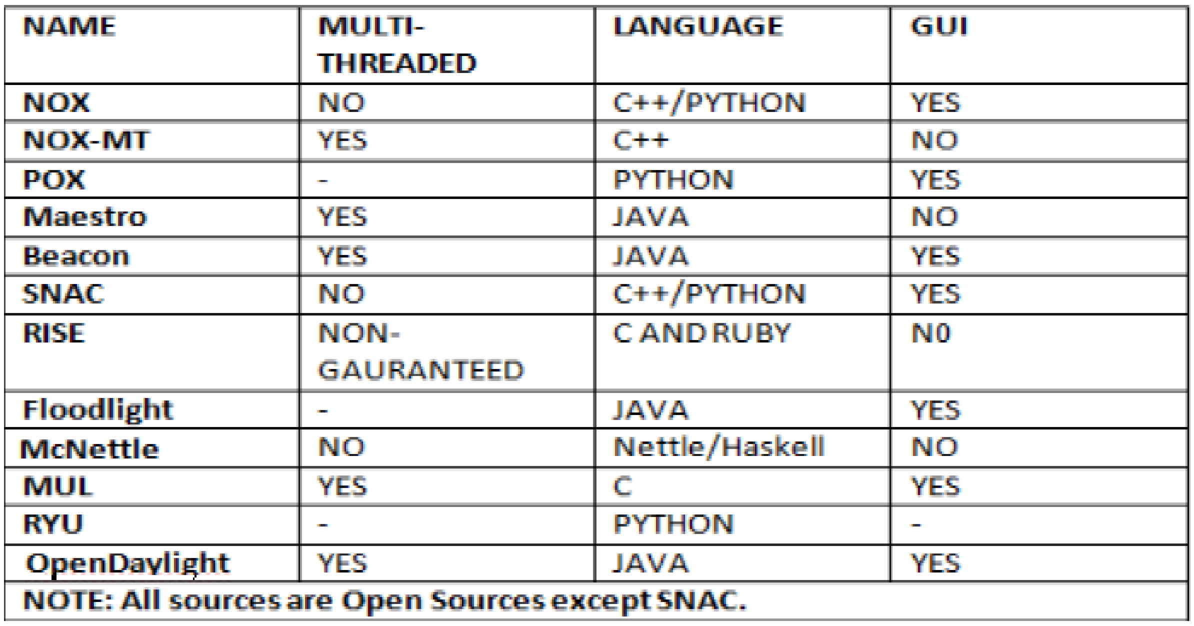
SDN TAXONOMY

Here we are concerned about the major issues raised by the introduction of SDN in the present network structure. They are majorly two taxonomies which face on single aspect of SDN. The first is based on switch level SDN deployment by Gartener in non-public report, where the 2nd focuses on the control plane. The 3 proposed control plane abstractions are given below:

1) FORWARDING ABSTRACTION:

A dynamic forwarding planning to be compatible with different kind of forwarding without exploring information about supporting hardware.

Table II: Controllers For SDN



NAME	MULTI-THREADED	LANGUAGE	GUI
NOX	NO	C++/PYTHON	YES
NOX-MT	YES	C++	NO
POX	-	PYTHON	YES
Maestro	YES	JAVA	NO
Beacon	YES	JAVA	YES
SNAC	NO	C++/PYTHON	YES
RISE	NON-GAURANTEED	C AND RUBY	N0
Floodlight	-	JAVA	YES
McNettle	NO	Nettle/Haskell	NO
MUL	YES	C	YES
RYU	-	PYTHON	-
OpenDaylight	YES	JAVA	YES
NOTE: All sources are Open Sources except SNAC.			

2) DISTRIBUTED STATE ABSTRACTION:

A single coherent global view of the network through a noted network graph accessible for control via an API.

3) SPECIFICATION (OR CONFIGURATION) ABSTRACTION:

Determines the behaviour of required controls like access controls, isolation, QoS, etc.

Following hierarchal taxonomy consists of three-levels: SDN layers, recognised problems related to SDN layer and the proposed solutions for SDN problems.

A. Infrastructure Layer

Here we shall be considering mainly the performance and scalability of the nodes also the correctness of the of each flow entry.

1. Performance and Scalability of the Forwarding Devices: To address this issue there exist some solutions listed below:
   - Utilizing Switch Resources
   - Lookup Procedure

B. Control Layer

The control layer faces specific challenges which are its performance also security of the layer:

1. Performance: In the scenario were there exist a single controller, this can be the determine factor for the performance of SDN networks in a large network. To solve this issue there have been various suggestions:
   - Control Plane dividing: Here suggestion has been made to the splitting of the control plane in either vertical or horizontal format.
   - Placement of controllers in distributed forms: This would solve performance issue but determine the amount of controllers needed to serve one SDN network is an issue
2. Security: There have been suggestions to the introduction of intelligence to the whole network.

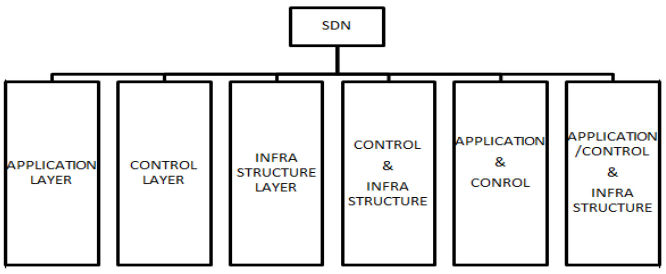


Fig.5 Overview of SDN Layers.

C. Application Layer

At the application layer, two major researches are analyzed which are:

1. SDN applications:

Interaction with controllers to obtain a particular function related to network such that network operators requirements are completed. To understand we can have example like quality of service (QoS), traffic engineering (TE) and load balancing (LB).

2. SDN Use cases:

In any specified scenario, to perform a particular use, applications are defined. There are several SDN uses cases like cloud computing, Information Content Networking (ICN), network virtualization, mobile networks and network function virtualization (NFV).

D. Control/Infrastructure Layers

Here we shall be looking at the challenges that might arise from the control and infrastructure layers.

1. Performance and Scalability: SDN sole aim was to keep the data plane very simple and assign the control task to a central controller, this leads to latency in the switch, controller connection due to the utilization of the channel. Addressing this issue has been addressed by Control Load Devolving
2. Network Correctness: The controller bears the task of defining how the flows re handled by the switch, there is need to keep network correctness in order for the network to remain efficient.

E. Application/Control Layers

There exist presently diverse applications with the aim of programming the controller in order to manage the network. This solution has posed two challenges which are: Policy correctness and Security of the Northbound Interface

F. Application/Contrrol/Infrastructure Layers

Deploying of SDN applications at the application layer has effect on the OpenFlow rules existing at the Infrastructure layer. Below we shall be considering the challenges the three layers face:

1. Policy Updates Correctness:
2. Network Correctness

SDN ISSUES AND RESEARCH DIRECTIONS

A. Infrastructure Layers

1. Performance and Scalability: The major challenges faced by switches that support SDN are listed below:
   - Flow Table Size
   - Lookup Procedure
   - CPU Power
   - Packet Buffer Size
2. Correctness of Flow Entries: Major errors in network are generated from Misconfiguration. Misconfigurations could be classified into: Intra-Switch and Inter-federated. Misconfiguration in turn affects the security and level of efficiency the network provides. Various tools have been presented to solve this issue, such as FlowChecker **[7]** which analyzes, validates and ensure a closed OpenFlow configuration in the existing infrastructure.

B. Control Layer: To solve the issue of performance, reliability and scalability there have been various solutions presented ranging from Control Plane partitioning to diverse placement of Controllers.

1. Control Plane Partitioning **[8]-[12]** have stated various methods for the division of the controllers. HyperFlow **[8]** for instance uses various Physical controllers but provides a central network control.

   Onix **[9]** is also another platform that distributes the control plane. It runs multiple servers known as ONIX controllers and these operate on a general view of the network where the present state of such controllers is stored in the Network Information Base (NIB).

   FlowVisor **[13], [14], [15]** distributes the network thereby allowing various network users to utilize the same physical infrastructure. It operates as a proxy between OpenFlow switches and various foreign network OS.
2. Security: **[16]** suggested the introduction of AVANT-GUARD whose function is to tackle the diverse SDN security threats, this introduces intelligence to the Data Plane.

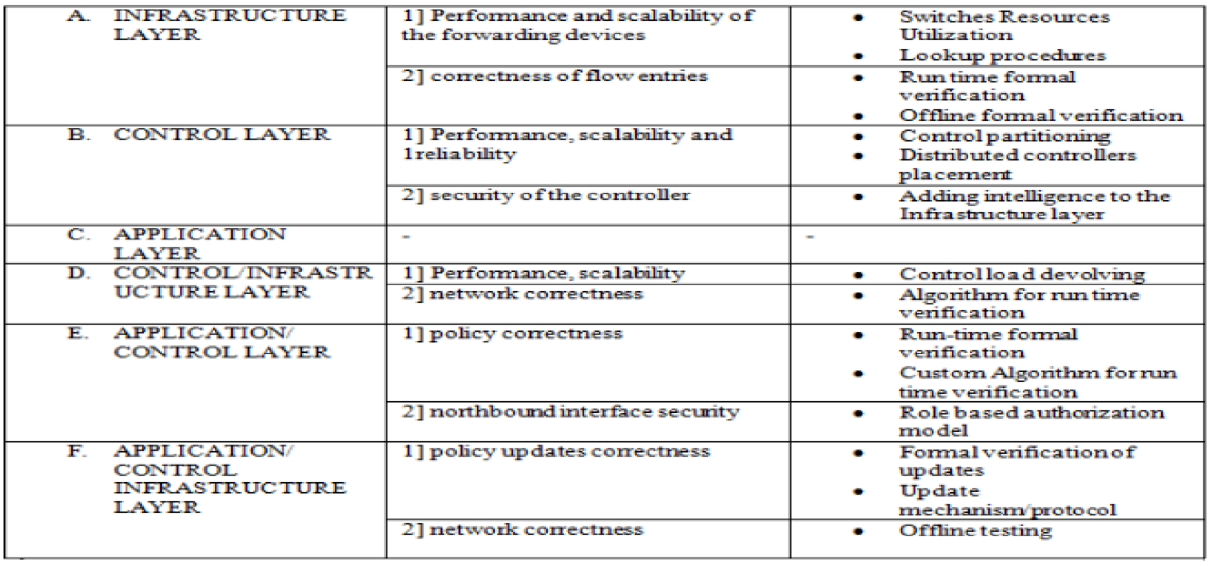
C. Application Layer

1. Here we shall be considering some major services and how SDN provide solutions to the challenges they faced in present day networks

- Security: SDN has provided a global view of the network, a logically intelligence unit and also a standard procedure for programming the network devices, these in turn has opened new doors for battling present day network security problems. Looking at the current day networks, Anomaly Detection Systems (ADS) are installed in the Service

  Providers Core Network but recent research output by **[17]** has proposed moving the ADS from the Core Network to the Home Network. They proved a more accurate policy in security when deployed to the Home Networks unlike the Core Networks.\
- Also, with regards to Distributed Denial of Service (DDOS) [**18]** suggested the installation of Security Applications in the NOX controllers. Here the controller collect information that relates to DDOS attacks after which it is been processed by an artificial neural network to predict irregularities in the network traffic and on which decision is been made. **[19]** Proposed a SDN based mechanism against network scanning which causes an IP address mutation therefore stopping hackers from predicting the specific IP addressing that is made on the network. **[20]** Proposed a mechanism which detects DDoS Attacks as well as Port Scan Attacks.

- Quality of Service: In this field **[22]** has suggested OpenQoS which is dedicated for the QoS delivery of multimedia traffic.



A. INFRASTRUCTURE LAYER	1] Performance and scalability of the forwarding devices	• Switches Resources Utilization • Lookup procedures
	2] correctness of flow entries	• Run time formal verification • Offline formal verification
B. CONTROL LAYER	1] Performance, scalability and 1reliability	• Control partitioning • Distributed controllers placement
	2] security of the controller	• Adding intelligence to the Infrastructure layer
C. APPLICATION LAYER	-	-
D. CONTROL/INFRASTRUCTURE LAYER	1] Performance, scalability	• Control load devolving
	2] network correctness	• Algorithm for run time verification
E. APPLICATION/ CONTROL LAYER	1] policy correctness	• Run-time formal verification • Custom Algorithm for run time verification
	2] northbound interface security	• Role based authorization model
F. APPLICATION/ CONTROL INFRASTRUCTURE LAYER	1] policy updates correctness	• Formal verification of updates • Update mechanism/protocol
	2] network correctness	• Offline testing

- Traffic Engineering: This is also known as traffic management and it is aimed at optimizing the network performance based on analysis of the traffic that transverses the network. Some solutions are implemented in organizations such as Google.
- Universal ACL Management: This is concerned with the universal configuration of policy updates on the nodes present in the SDN by the administrator.

Table III

- Load Balancing: Various papers have been published such as **[23]** which propose methods for balancing the traffic load in the networks.

2. SDN Use cases: Various papers have suggested the implementation in environments such as Cloud computing, mobile networks and some other ones for maximum performance, we shall be looking at some major environments

- Cloud Computing: Data centers that provide cloud computing face challenges such as cost of deployment and operation**[24]** suggested the using OpenFlow to virtualize the physical network which allows for the operation of a common infrastructure supporting all participating cloud operators.

  Mobile networks: Due to the traffic generated by the mobile users, there has been need for the supply of more efficient networks to manage such traffics, OpenFLow wireless also known as OpenRoad has been proposed by **[25]** to support the Wireless

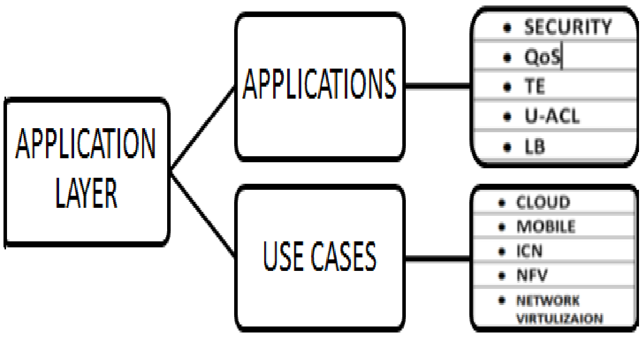


Fig.6.A

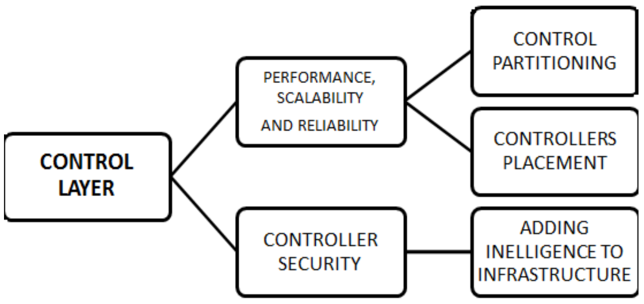

CONTROL LAYER
PERFORMANCE, SCALABILITY AND RELIABILITY
CONTROLLER SECURITY
CONTROL PARTITIONING
CONTROLLERS PLACEMENT
ADDING INELLIGENCE TO INFRASTRUCTURE

Fig.6.B

Networks. Also a method that involves programing of the data plane in Cellular Networks has been proposed by **[26]** known as OpenRadio, it suggests separation of the of the wireless protocols from their existing hardware and implementing of a Software abstraction layer which can be remotely programmed in various base stations.

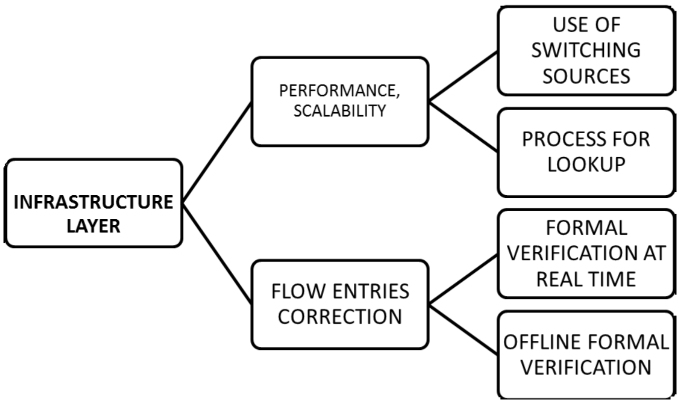


Fig.6.C

Fig.6 Summarized Classification of Research Works

A) Application Layer B) Control Layer and C) Infrastructure Layer

D. Control/Infrastructure Layer

Considering the interconnection existing between the control and data layers, we shall be considering two dynamic parameters.

Performance and Scalability: OpenFlow was designed to maintain simplicity within the data plane and for the controller to handle the control task. This concept therefore has led to the situation whereby the switch needs to consult the controller for dataflow instruction resulting to congestion between the switch to controller connection which in turn adds latency to the processing of the initial packets of a flow in the switch node. A new system for traffic management **[130]** suggested the replacement of each flow monitoring in the switches with an elephant flow detection existing between the end-hosts and the signaling sent to the controller.

Various suggestions have been made but they all require modifying either the OpenFlow protocol, end-host or switching nodes which in turn leads to the reduction of flow visibility in the Controller.

E. Application/Control Layers

1. Policy Correctness: The existing challenge of policy detection and re-conciliation of conflicting flow rules has been addressed by **[30]** where an extension known as FortNox was added to the controller. FortNox has an inbuilt Analyzer for conflict which detects any conflicts in flow rule insertion request made by various OpenFlow applications.

F. Application/Control/Infrastructure Layers

1. Policy Update Correctness: Network failure could be introduced to the network resulting from inconsistencies of network configurations. To solve this challenge a launch of KINECTIC which provides a per-packet abstraction in the NOX OpenFlow Controller is seen in **[28]**
2. Network Correctness: Considering the errors that arise from design and logics there were launch of various OpenFlow Testing bench such as OFTEN **[29]** which tests the SDN systems to avoid violating correction settings.

Conclusion

SDN presents a more dynamic method for packet switching for the evolving networks that would require faster and more secure channels. Also it comes with various challenges that Implementation would face.

In this paper, more light has been shed on some fundamental challenges SDN faces and proposed solutions for each of these challenges.SDN has envisioned by many organizations is the future of networking but to realize this vision we need to work on the various components ranging from the controllers, switches and application services and also the interfaces interconnecting them. Once the SDN structure has matured mass deployment would be experienced in the present Networks transitioning us from the traditional networks to a more brilliant and functional network.

References

1. A Survey And Layered Taxonomy Of Software Defined Networking, IEEE Communication Surveys & Tutorials, Vol. 16, No. 4, Fourth Quarter 2014.
2. A Survey on Software-Defined Network and OpenFlow: From Concept to Implementation, FeiHu, Qi Hao, and KeBao, IEEECOMMUNICATIONSURVEYS&TUTORIALS,VOL.16, NO.4, OURTHQUARTER2014.
3. A.T.Campbelletal.,"A survey of programmable networks," SIGCOMMComput.Commun.Rev.,vol.29,no.2,pp.7–23,Apr.1999.
4. Defense Advanced Research Projects Agency.(1997).Active network program. [Online].Available: http://www.sds.lcs.mit.edu/darpa-activenet
5. N.Feamster,J.Rexford,andE.Zegura,"TheRoadtoSDN:AnIntellectualHistoryofProgrammableNetworks,"ACMQueue,NewYork,NY, USA,Tech.Rep.,2013
6. A.Greenbergetal.,"Acleanslate4 D approach to network control and management, "ACMSIGCOMM Comput. Commun.Rev.,vol.35,no.5, pp.41–54,Oct.2005
7. E. Al-Shaer and S. Al-Haj, "FlowChecker: Configuration analysis and verification of federated OpenFlow infrastructures," in Proc. 3rd ACM Workshop SafeConfig., 2010, pp. 37–44.
8. A. Tootoonchian and Y. Ganjali, "HyperFlow: A distributed control plane for OpenFlow," in Proc. INM/WREN, 2010, pp. 3–3.
9. T. Koponen et al., "Onix: A distributed control platform for large-scale production networks," in Proc. 9th USENIX Conf. OSDI, 2010, pp. 1–6.
10. V. Yazıcı, M. O. Sunay, and A. Ö. Ercan, "Controlling a software- defined network via distributed controllers," in Proc. NEM Summit, Implementing Future Media Internet Towards New Horizons, 2012, pp. 16–21.
11. S. Hassas Yeganeh and Y. Ganjali, "Kandoo: A framework for efficient and scalable offloading of control applications," in Proc. 1st Workshop HotSDN, 2012, pp. 19–24.
12. C. Macapuna, C. Rothenberg, and M. Magalha es, "In-packet bloom fil- ter based data center networking with distributed OpenFlow controllers," in Proc. IEEE GC Wkshps, 2010, pp. 584–588.
13. R.Sherwoodetal.,"Cantheproductionnetworkbethetestbed?"in Proc.9thUSENIXConf.OSDI,2010,pp.1–6

14. B. Sonkoly et al., “OpenFlow virtualization framework with advanced capabilities,” in Proc. EWSDN, 2012, pp. 18–23.
15. R. Sherwood et al., “Carving research slices out of your production networks with OpenFlow,” Proc. SIGCOMM Comput. Commun. Rev., vol. 40, no. 1, pp. 129–130, Jan. 2010.
16. S. Shin, V. Yegneswaran, P. Porras, and G. Gu, “AVANT-GUARD: Scalable and vigilant switch flow management in software-defined networks,” in Proc. ACM SIGSAC Conf. CCS, 2013, pp. 413–424.
17. S. A. Mehdi, J. Khalid, and S. A. Khayam, “Revisiting traffic anomaly detection using software defined networking,” in Proc. 14th Int. Conf. RAID, 2011, pp. 161–180.
18. R. Braga, E. Mota, and A. Passito, “Lightweight DDoS flooding attack detection using NOX/OpenFlow,” in Proc. IEEE 35th Conf. LCN, 2010, pp. 408–415.
19. J. H. Jafarian, E. Al-Shaer, and Q. Duan, “OpenFlow random host muta- tion: Transparent moving target defense using software defined network- ing,” in Proc. 1st Workshop HotSDN, 2012, pp. 127–132.
20. K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras, and V. Maglaris, “Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environ- ments,” Comput. Netw., vol. 67, pp. 122–136, Apr. 2014.
21. S. Shirali-Shahreza and Y. Ganjali, “Efficient implementation of security applications in OpenFlow controller with FleXam,” in Proc. IEEE 21st Annu. Symp HOTI, 2013, pp. 49–54.
22. H. E. Egilmez, S. T. Dane, B. Gorkemli, and A. M. Tekalp, “OpenQoS: OpenFlow controller design and test network for multimedia deliverywith quality of service,” in Proc. NEM Summit, Implementing Future Media Internet Towards New Horizons, 2012, pp. 22–27.
23. R.Wang,D.Butnariu,andJ.Rexford,“OpenFlow-basedserverload balancinggonewild, ”inProc. 11thUSENIXConf. Hot-ICE,2011, pp.12–12.
24. J.Matias,E.Jacob,D.Sanchez,andY.Demchenko,“AnOpenFlowbasednetworkvirtualizationframeworkforthecloud, ”inProc.IEEE 3rdInt.Conf.CloudCom,2011,pp.672–678.
25. K.K.Yapetal.,“OpenRoads:Empoweringresearchinmobilenetworks,”SIGCOMMComput.Commun.Rev.,vol.40,no .1,pp.125–126, Jan.2010.
26. M.Bansal,J.Mehlman,S.Katti,andP.Levis,“OpenRadio:Aprogrammablewirelessdataplane,”inProc.1stWorkshopH otSDN,2012, pp.109–114.

27. A. Curtis, W. Kim, and P. Yalagandula, “Mahout: Low-overhead data- center traffic management using end-host-based elephant detection,” in Proc. IEEE INFOCOM, 2011, pp. 1629–1637.
28. M. Reitblatt, N. Foster, J. Rexford, C. Schlesinger, and D. Walker, “Abstractions for network update,” in Proc. ACM SIGCOMM Conf. Appl., Technol., Archit., Protocols Comput. Commun., 2012, pp. 323–334
29. M. Kuzniar, M. Canini, and D. Kostic, “OFTEN testing OpenFlow net- works,” in Proc. EWSDN, 2012, pp. 54–60.
30. P. Porras et al., “A security enforcement kernel for OpenFlow networks,” in Proc. ACM SIGCOMM Workshop HotSDN, Aug. 2012, pp. 121–126.