ISSN No. 0976-5697

Volume 8, No. 5, May-June 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Network Management using Software Defined Networking

Er. Jaspreet Singh and Er. Yadwinder Kaur
Assistant Professor
Department of Computer Science and Engineering
Chandigarh University, Gharuan (Punjab), India

Abstract: The Internet has led to the creation of a digital society, where (almost) everything is connected and is accessible from anywhere. Today's internet is growing very fast, devices to the network for accessing and providing the services. It is difficult to configure the traditional network according to predefined policies, reconfiguration, response to fault and load changes. The static nature of the conventional network makes it difficult to accomplish the dynamic computing and storage desires of big data centres, enterprises and campuses. Software defined networking is the new approach to networking. Software defined network model promises to simplify the network configuration and resource management. It will replace the current network and fulfil the business needs via software rather than hardware only. It makes it easier to create and introduce new abstractions in networking, simplifying network management and facilitating network evolution. This paper introduces the concepts of software defined network which will help to configure and manage the network needs.

Keywords: SDN, Network Management, Data Plane, Control Plane ,ONF, Open Flow

I. INTRODUCTION

Controlling and managing networks has become a highly complex and specialized activity. In the traditional approach to networking, most network functionality is implemented in a dedicated appliance i.e. switch, router etc. In addition, within the dedicated appliance, most of the functionality is implemented in dedicated hardware such as an ASIC (Application Specific Integrated Circuit).Networking organizations are under increasing pressure to be more efficient and agile that is not possible with the traditional approach to networking.

Software defined networking (SDN) is an emerging networking paradigm that gives hope to change the limitations of current network infrastructure[1].It gives assurance to dramatically reduce the complexity of network configuration and management as well as to make the introduction of innovation in the network operations possible. The challenges associated with the current traditional network are like an explosion of the cloud, growth of big data centers, the mobility that make the network very complex and putting pressure on the networks[2].Open Networking Foundation (ONF) is a nonprofit consortium dedicated to the development, standardization and commercialization of SDN.

ONF defined SDN as follows: Software defined networking is an emerging architecture that is dynamic, manageable, cost-effective and adaptable, making it ideal for the high-bandwidth, dynamic nature of applications[3].

SDN recommends split architecture by separating the device functionality into different planes like control plane and data plane, allowing them to operate independently (Figure 1). It aggregates the distributed intelligence spread across in different network elements of a large network into a centralized control layer that helps to improve the efficiency of networks and optimum utilization of resources.

SDN is flexible in its design to operate, in coexistence with legacy networking devices and the new genre of low cost switching infrastructure with tremendous advantage of programmability, thereby ensuring competitive benefits to service providers, data centers and enterprises with respect to scalability, performance and high availability of services. Although SDN and Open Flow started as academic experiments, they have gained significant importance in the industry over the past few years. Most vendors of commercial switches now include support for Open Flow API in their equipments[3].

Open Flow protocol is a foundational element for building SDN solutions. SDN momentum was strong enough to make companies like Google, Facebook, Yahoo, Microsoft, Verizon and Deutsche Telekom to fund Open Networking Foundation (ONF) with the main goal of promotion and adoption of SDN through open standard development.[4].

ONF owns Open Flow specification and standardization.

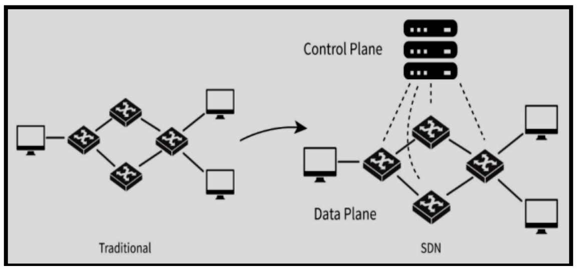

Figure 1: Traditional Networks and SDN

II. GENESIS OF SDN

SDN leverages on networking ideas with a longer history [5]. It builds on work done on programmable networks such as active networks, programmable ATM networks and on proposals for control and data plane separation such as the network control point (NCP) and routing control platform (RCP) [6] [7].

SDN has been evolving since1996 driven by the desire to provide user controlled management of forwarding in the network nodes, implementations by research and industry groups include Ipsilon (General Switch Management protocol, 1996), the Tempest (a framework for safe, resource-assured, programmable networks, 1998) and Internet Engineering Task Force (IETF) Forwarding and Control Element Separation, 2000 and Path Computation Element, 2004 [8].
Open Flow (2008) has brought the implementation of SDN closer to reality. Open Flow acts as interface between switches/Network element and Network OS. Open Flow-based SDN technologies enable IT to address the high-bandwidth, dynamic nature of today's applications, adapt the network to ever changing business needs and significantly reduce operations and management complexity[9].
Open Networking Foundation was founded in 2011 to promote SDN and Open Flow. By 2014 ONF had grown to over 150 members. By 2016, SDN is evolving in the industry as a marketing term. SDN addresses the fact that the static architecture of conventional networks is ill-suited to the dynamic computing and storage needs of today’s data centers, cloud services etc.

III. OPEN FLOW SPECIFICATION

Open Flow protocol provides a standardized protocol to separate control/ management plane from data plane. Open Flow specification defines standard collection of features that switches must provide as well as an interface that controllers can use to communicate with switches including instructions for installing, deleting forwarding rules, notifications about flows, topology and traffic statistics.

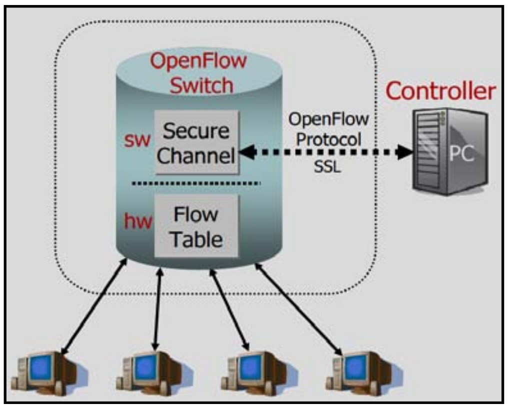

Figure 2: Open Flow Architecture

The Open flow architecture consists of three principal elements(figure 2):

- Open Flow controller (an external controller)

It process packet match, instruction, action set and pipeline processing.

- Open Flow switch

It includes two major components as
secure channel(software) and flow table(hardware)

- Open flow Protocol

It establishes the communication between switch and controller through a secure channel.

Switches/routers are modeled as forwarding elements with forwarding tables, containing flow entries.
Open Flow protocol affects the forwarding tables.
Open Flow provides a powerful tool for efficient control of enterprise networks. It is already applied to wide area networks & large datacenters.

IV. ARCHITECTURE OF SDN

Open Networking Foundation (ONF) is a user-driven organization dedicated to the promotion, adoption of SDN and implementing SDN through open standards where such standards are necessary to move the networking industry forward. ONF is developing open standards such as the Open Flow Standard and the Open Flow Configuration and Management Protocol Standard. Open Flow Standard is vendor-neutral standard communication interface between the control and forwarding layers of an SDN architecture. Open Flow provides an opened protocol to program the flow table in different switches and routers [9].
In the SDN architecture(figure 3), the control and data planes are decoupled, network intelligence and state are logically centralized and the underlying network infrastructure is abstracted from the applications. In traditional networking, forwarding plane, control plane, management and services plane are largely integrated into each network device. In contrast, the SDN model deploys the majority of management, services and path selection decisions either as an advanced function within the controller software or on northbound applications to the controller. An Open Flow agent is loaded on the network device to serve as an intermediary between the Open Flow controller and the underlying switch operating system.

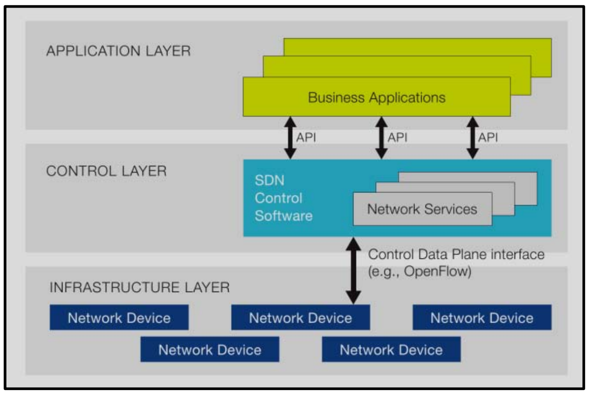

Figure 3: SDN Architecture

SDN architecture splits into various planes like

Data plane

Data plane comprises a set of one or more network elements, each of which contains a set of traffic forwarding/traffic processing resources. Forwarding devices are interconnected through wireless radio channels or wired cables. The network infrastructure comprises the interconnected forwarding devices, representing the data plane.

Controller plane

Forwarding devices are programmed by control plane elements through well-defined interfaces. The control plane can therefore be seen as the network brain. All control logic rests in the applications and controllers which form the control plane. The controller plane comprises a set of SDN controllers each of which has exclusive control over a set of resources exposed by one or more network elements in the data plane.SDN control software can provide network services which refers to the functionality that enables business applications to perform efficiently and securely. It includes wide ranges of security services such as firewalls, IDS/IPS and DDoS protection. Control plane uses various *API (Application Programming Interface)* like northbound API that enables communications between the control layer and the business application layer. There is currently no standard based northbound API. Southbound API enables communications between the control layer and the infrastructure layer. Protocols that enables communications include Open Flow, the extensible messaging and presence protocol (XMPP),the network configuration protocol etc.

Application plane

Application plane comprises one or more applications each of which has exclusive control of set of resources exposed by one or more SDN controllers. An application may invoke or collaborate with other applications. An application may act as an SDN controller in its own right.

Management Plane

Each application, controller and network element has a functional interface to a network manager. The minimum functionality of the manager is to allocate resources from a resource pool in the lower plane to a particular client entity in the higher plane and to establish reachability information that permits the lower and higher plane entities to mutually communicate. Network management centre is responsible for implementing various functions such as firewalls, custom policies and protocol implementations. It may also perform operations that the application, controller and data planes are restricted from doing by policy or for other reasons.

V. BENEFITS OF SDN

SDN approach provides several benefits which includes:

1) *Virtualization*

Use network resources without worrying about where it is physically located, how much it is, how it is organized etc. Controller provides a global view of network resources.

2) *Orchestration*

Able to control and manage thousands of devices with one command.

3) *Programmable*

Able to change network behavior on the fly.

4) *Dynamic Scaling*

Able to change network size, quantity etc.

5) *Visibility*

Monitor network resources, connectivity.

6) *Automation*

To lower or minimize manual involvement for network troubleshooting, reduce network downtime, policy enforcement, provisioning/re-provisioning or segmentation of resources, add new workloads, sites, devices and resources.

7) *Multi-tenancy*

Complete control over addresses, topology, routing, security as Tenants.

8) *Service Integration*

Load balancing, firewalls, Intrusion detection Systems, provisioned on demand and placed appropriately on the traffic path.

9) *Performance*

Optimize network utilization, traffic engineering or bandwidth management, capacity optimization, High utilization, Fast failure handling.

VI. ISSUES IN SDN

SDN faces many challenges during implementation. The major issues in its implementation are:

- SDN is a centralized solution, failure in the SDN controller will block the entire functionality of the network.
- The data and control planes are decoupled but they can progress independently as long as the API connects them. The decoupling process has its own drawbacks, such as having a standard API for both planes and the SDN controller become the bottle neck in a situation where the network scales the number of switches and number of nodes up.
- SDN controllers must be wisely configured and the SDN’s network topology authenticated to prevent manual errors and increase network availability.
- A security model must be evolved to take care of varying privilege requirements of application. The controllers are a particularly attractive target for attack in the SDN architecture, open to unauthorized access and exploitation.
- Performance of the network is another important area to look into. The networks should be flexible enough so that new features and capabilities can be programmed.
- In case of large-scale migration, transition to SDN requires coexistence of SDN and the legacy equipment. More developmental work is required to achieve a hybrid SDN infrastructure in which SDN enabled and traditional integrated nodes inter-operate. Such a solution would reduce the cost, risk and disruption for enterprise and carrier networks transitioning to SDN.

VII. SDN TOOLS AND LANGUAGES

Various tools & languages are used to monitor and implement software defined networks. Certain SDN initiatives have focused on forming a platform, Onix, to implement SDN controllers as a distributed system for flexible network management [10]. Other studies have presented a network debugging tool, Veriflow which is capable of discovering the faults in SDN application rules and hence preventing them from disrupting network performance[11].

Other initiatives have developed a routing architecture, Routeflow, which is inspired by SDN concepts provides an interaction between hardware performance and flexible open-source routing stacks[12]. Hence, it opens the door to migration from traditional IP deployments to SDN.

In addition to recent studies that developed physical SDN prototypes, other researchers have provided an efficient SDN innovation, Mininet[13]. Mininet is a virtual emulator which provides an environment for prototyping any SDN idea. Whenever the prototype evaluation is acceptable then it can be deployed in various research networks and for general use[14]. Mininet supports research, development, learning, prototyping, testing, debugging and any other tasks that could benefit from having a complete experimental network on a laptop or PC.

Mininet's services are hindered by certain imitations: poor performance at high loads and its lightweight virtualization approach.There are various simulation tools like NS-3.It is developed after ns-2 for network research and education. It is written in C++ & simulation programs are C++ executables or python scripts.NS-3 is free software, licensed under the GNU GPLv2 license and is publicly available for research, development and use. The goal of the ns-3 project is to create an open simulation environment for computer networking research community.

Table1: Various Tools for SDN [15]

Category	***Purpose***	***Software and tools***
Emulation and simulation	Emulating network topologies as well as providing reference for network event simulation	Mininet, ns-3, OMNeT++
Software switches and platforms	A software platform to test and validate switch-controller behaviour and southbound protocol working	Open vSwitch, Indigo, Pantou
Debugging and troubleshooting	Tool set to debug SDN behaviour at the switch and controller level	NICE, VeriFlow, OFRewind, NDB, Wireshark

Research has also been directed towards developing control support for SDN & describing new language approaches to program Open-Flow networks.SDN programmability enables developers to automatically program the network to support applications. Basic SDN programming involves three components: the network to be controlled, the controller that controls the network & programming environment.

A programming language can be used to develop either passive or active policies. The former can only observe the network state, while the latter can reactively affect the network wide state as a response to certain network events.

There are various languages which is used for the development of SDN.

a) Frenetic

It is a domain-specific language for programming Open Flow networks[16].It is embedded in Python and comprises of two levels of abstraction. A limited, but high-level and declarative network query language.

The query language provides means for reading the state of the network, merging different queries and expressing high level predicates for classifying, filtering, transforming and aggregating the packets streams traversing the network.

b) Procera

Procera is a high level network control language that allows network operators to express reactive network control policies, without having to resort to general purpose programming of the network controller [17]. Procera is both expressive and extensible so users can easily extend the language by adding new constructs.

c) Nettle

It is another language which was created to configure networks[18].It is embedded in the strongly typed language, Haskell. It can be understood as a language that exhibits electrical circuits where it transforms messages issued from switches into commands generated by the controller by defining various signal functions for them. Nettle also allows programmers to define their own functions. It is considered more appropriate for programming controllers because it is a relatively low level language compared to other languages[18].

Advances in high-level programming languages are fundamental component to the success of a prolific SDN application development ecosystem.

VIII. APPLICATIONS OF SDN

There are many applications of software defined networking

- Enterprise Networks
- Data Centers
- Infrastructure-based Wireless Access Networks
- Cellular Networks
- Optical Networks
- Home and Small Business

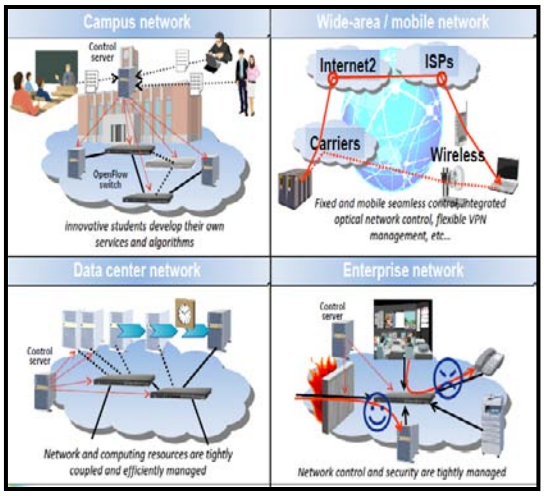
Campus network
Control server
OpenFlow switch
innovative students develop their own services and algorithms
Wide-area / mobile network
Internet2
ISPs
Carriers
Wireless
Fixed and mobile seamless control, integrated optical network control, flexible VPN management, etc...
Data center network
Control server
Network and computing resources are tightly coupled and efficiently managed
Enterprise network
Control server
Network control and security are tightly managed

Figure 4: Example Scenarios of SDN

IX. CONCLUSION

SDN is designed to address networking needs that are poorly addressed by existing networks.SDN market has attracted a great deal of attention over the past few years, although real-world adoption has been relatively slow. SDN's ability to greatly improve performance and efficiency while cutting costs, all signs indicate that its growth will skyrocket over the next several years.SDN provides tremendous opportunities for server platform vendors, network operating system developers, independent software vendors (Network and/or Business applications) to meet the desires of cloud operators, data centers and enterprises.

SDN also opens the way to mission-focused benefits, such as improved customer service as well as the creation of new network capabilities and services simply by building the appropriate software applications. SDN can also be a driving force for network innovation. The most successful commercial implementations of SDN includes Google, Amazon, Microsoft, Cisco, HP etc. Most organizations have yet to formulate a network transition strategy or even to start transition planning.

X. REFERENCES

[1] A.L. Barona Lopez, L.I.; Garcia,L.J., ‘‘Evolution and challenges of software defined networking,” IEEE,2012.

[2] A. Siamak, “Software defined networking with openflow”, Birmingham B3 2PB, UK,Oct. 2012.

[3] N. McKeown et al., ‘‘OpenFlow: Enabling innovation in campus networks,’’ SIGCOMM Computer Communication Review, vol. 38, no. 2, pp. 69–74, Mar. 2008.

[4] Open Networking Foundation: https://www.opennetworking.org/ (Available Online)

[5] N. Feamster, J. Rexford, and E. Zegura, ‘‘The road to SDN,’’ Queue, vol. 11, no. 12, pp. 20:20–20:40, Dec. 2013.

[6] D. Tennenhouse, J. Smith, W. Sincoskie, D. Wetherall, and G. Minden, ‘‘A survey of active network research,’’ IEEE Commun. Mag., vol. 35, no. 1, pp. 80–86, Jan. 1997.

[7] M. Caesar et al., ‘‘Design and implementation of a routing control platform,’’ in Proc. 2nd Conf. Symp. Networking. System Design Implement., 2005, vol. 2, pp. 15–28.

[8] N Feamster et al.," The Road to SDN: An Intellectual History of Programmable Networks", ACM SIGCOMM Computer Communication Review, Volume 44 Issue 2, April 2014,Pages 87-98.

[9] https://www.opennetworking.org/sdn-resources/openflow

[10] Koponen, T.,Casado, et al., "Onix: A Distributed Control Platform for Large-Scale Production Networks," Proceedings, Ninth USENIX Conference on Operating Systems Design and Implementation,Berkeley,2010.

[11] Khurshid, A.,Zhou, W.X.,et al., "VeriFlow: Verifying Network-Wide Invariants in Real Time," First Workshop on Hot Topics in Software-Defined Networks, NY, New York,pp.49–54, 2012.

[12] Nascimento, M.R., Rothenberg, C.E.,et al., "Virtual Routers as a Service: The Route Flow Approach Leveraging Software-Defined Networks," Proceedings, Sixth International Conference on Future Internet Technologies (CFI '11), New York, NY,pp.34–37, 2011.

[13] http://mininet.org (Available Online)

[14] Lantz, B., Heller, B., Mc Keown, N., "A Network in a Laptop: Rapid Prototyping for Software-Defined Networks," Proceedings, Ninth ACM SIGCOMM Workshop on Hot Topics in Networks, New York, NY,2010.

[15] (Online)https://www.hindawi.com/journals/wcmc/2017/7191647/tab5/

[16] N. Foster, R. Harrison, M. J. Freedman, C. Monsanto, J. Rexford,A. Story, and D. Walker, “Frenetic: a network programming language,”SIGPLAN Not., 2011.

[17] Voellmy, A.,Kim, H.J., Feamster, N., "Procera: A Language for High-Level Reactive Network Control," First Workshop on Hot Topics in Software-Defined Networks (HotSDN '12), New York, NY, pp. 43–48,2012.

[18] Voellmy, A., Hudak, P., “Nettle: Functional Reactive Programming of OpenFlow Networks,” Practical Aspects of Declarative Languages Symposium, January 2011.