International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No. (01), January 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

SOFTWARE DEFINED NETWORKS - A SURVEY

Dr.B.Sujatha, Mr.K.Ashokkumar[2]

[1]Professor/CSE, Sengunthar Engineering College
[2]Associate Professor/CSE, Sengunthar Engineering College

ABSTRACT
The plan of "Programmable Network" has been projected as a way to facilitate network evolution. The emergence of Software-Defined Networking (SDN) has played a significant role in designing and managing networks. Software-Defined Networking is an architecture that allows network administrator can programmatically initialize, change, control and manage the network's behavior dynamically using open protocols such as OpenFlow. There is a physical separation between network control planes from the forwarding functions in SDN architecture. This paper presents the concept of SDN, comparison of SDN with traditional networks, its architecture and its applications.
Keywords: Programmable Network, SDN, OpenFlow

I.INTRODUCTION

The imminent condition of rich multimedia contents and promising trends in ICT area [1], in particular, mobile, social, cloud [2] and big data [3], [4], [5] are influencing computer networks for higher bandwidth, accessibility, security and dynamic management. The increasing demand for big data analytics of a different kinds of data sources, are demanding higher network connection speed than ever. Next, a wider usage of mobile devices and social networks is demanding ubiquitous communications to fulfill the social needs of people. Social networks have also experienced a dramatic growth in recent years like Facebook, twitter etc. Finally, Cloud computing has more demands on the flexibility and agility of computer networks At the same time, with more computing and storage resources placed remotely in the cloud, efficient access to these resources via a network is becoming critical to fulfill today's computing needs. IT industry has enjoyed innovation such as virtualization in computing and storage. The end is nowhere; technology will continue to evolve in years to come. On the contrary, networking has experienced limited innovation over the past 20 years. This stagnation has led to overly complex and inflexible networks no longer meeting current business requirements. Today, business and technical network requirements include enhancing performance and realizing broader connectivity. Companies have to meet more and more industry-specific security regulations and there is a growing demand for mobility. In order to comply with all of these criteria, networking protocols have evolved significantly over the last few decades. However, the way traditional networks are set up; deploying one protocol to realize these needs organization-wide is quite the challenge.

Traditional network configuration: (figure.1)

The traditional networking is characterized by two main factors:

1) Network functionality is mainly implemented in dedicated appliances, i.e. Switches and

Routers.

2) Most functionality within this appliance is implemented in dedicated hardware.

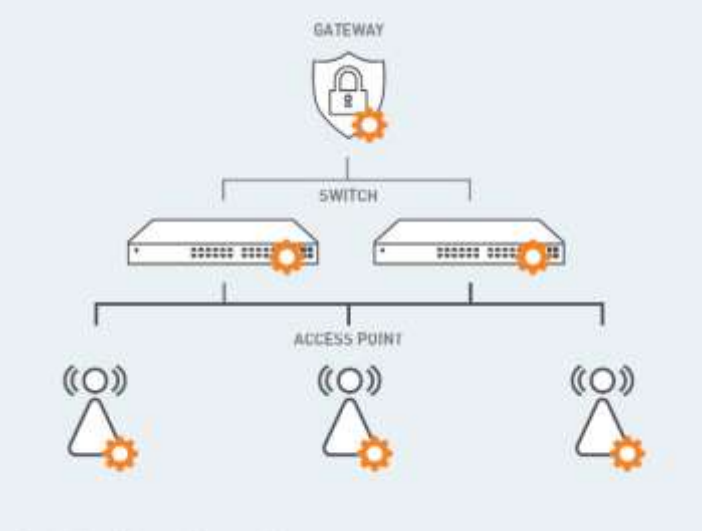


fig1 Traditional Networking

Nowadays organizations are increasingly confronted with the following limitations while running their systems in conventional networking:

1. Traditional networking is time-consuming and error-prone.
2. A high-level of expertise required for multi-vendor environments.
3. Traditional architectures complicate network segmentation.

To overcome this limitation, we can go for a new perspective on network management. The idea of "programmable networks" [6] has been introduced to facilitate network evolution. Software Defined Networking (SDN) is a new networking paradigm in which the forwarding functions are decoupled from control decisions. It simplifies network management and enables innovation and evolution. The main idea is to allow software developers to rely on network resources in the same easy manner as they do on storage and computing resources. In SDN, the network intelligence is logically centralized in software-based controllers (the control plane), and network devices become simple packet forwarding devices (the data plane) that can be programmed via an open interface (e.g. OpenFlow).

II. SOFTWARE DEFINING NETWORK

Data communication networks typically consist of end-user devices, or hosts interconnected by the network infrastructure. This infrastructure is shared by hosts and employs switching elements such as routers and switches as well as communication links to carry data between hosts. Routers and switches are usually "closed" systems, often with limited-and vendor-specific control interfaces. Therefore, once deployed and in production, it is quite difficult for current network infrastructure to evolve.

Software-Defined Networking was developed to facilitate innovation and enable simple programmatic control of the network data-path. The separation of the forwarding hardware from the control logic allows easier deployment of new protocols and applications, straightforward network visualization and management, and consolidation of various middleboxes into software control.

Definition of SDN:

Software-Defined Networking (SDN) is an emerging architecture that is dynamic, manageable, cost-effective, and adaptable, making it ideal for the high-bandwidth, dynamic nature of today's applications. This architecture decouples the network control and forwarding functions, enabling network control to become directly programmable and the underlying infrastructure to be abstracted from applications and network services. [7]

SDN is defined by two characteristics, namely decoupling of control and data planes, and programmability on the control plane. Decoupling these two planes involves leaving the data plane with network hardware and moving the control plane into a software layer. By abstracting the network from the hardware, policies no longer have to be executed on the hardware itself. Instead, the use of a centralized software application functioning as the control plane makes network virtualization possible.

The OpenFlow protocol can be used in SDN technologies. The SDN architecture is:

Directly programmable: Network control is directly programmable because it is decoupled from forwarding functions.

Agile: Abstracting control from forwarding lets administrators dynamically adjust network-wide traffic flow to meet changing needs.

Centrally managed: Network intelligence is (logically) centralized in software-based SDN controllers that maintain a global view of the network, which appears to applications and policy engines as a single, logical switch.

Programmatically configured: SDN lets network managers configure, manage, secure, and optimize network resources very quickly via dynamic, automated SDN programs, which they can write themselves because the programs do not depend on proprietary software.

Open standards-based and vendor-neutral: When implemented through open standards, SDN simplifies network design and operation because instructions are provided by SDN controllers instead of multiple, vendor-specific devices and protocols.

III. THE NEED FOR A NEW NETWORK PARADIGM

Changing traffic patterns:

Within the enterprise data center, traffic patterns have changed significantly. In contrast to client-server applications where the bulk of the communication occurs between one client and one server, today's applications access different databases and servers, creating a flurry of "east-west" machine-to-machine traffic before returning data to the end user device in the classic "north-south" traffic pattern. At the same time, users are changing network traffic patterns as they push for access to corporate content and applications from any type of device (including their own), connecting from anywhere, at any time. Finally, many enterprise data centers managers are contemplating a utility computing model, which might include a private cloud, public cloud, or some mix of both, resulting in additional traffic across the wide area network.

The "consumerization of IT":

Users are increasingly employing mobile personal devices such as smartphones, tablets, and notebooks to access the corporate network. IT is under pressure to accommodate these personal devices in a fine-grained manner while protecting corporate data and intellectual property and meeting compliance mandates.

The rise of cloud services:

Enterprises have enthusiastically embraced both public and private cloud services, resulting in unprecedented growth of these services. Enterprise business units now want the agility to access applications, infrastructure, and other IT resources on demand and à la carte. To add to the complexity, IT's planning for cloud services must be done in an environment of increased security, compliance, and auditing requirements, along with business reorganizations, consolidations, and mergers that can change assumptions overnight. Providing self-service provisioning, whether in a private or public cloud, requires elastic scaling of computing, storage, and network resources, ideally from a common viewpoint and with a common suite of tools.

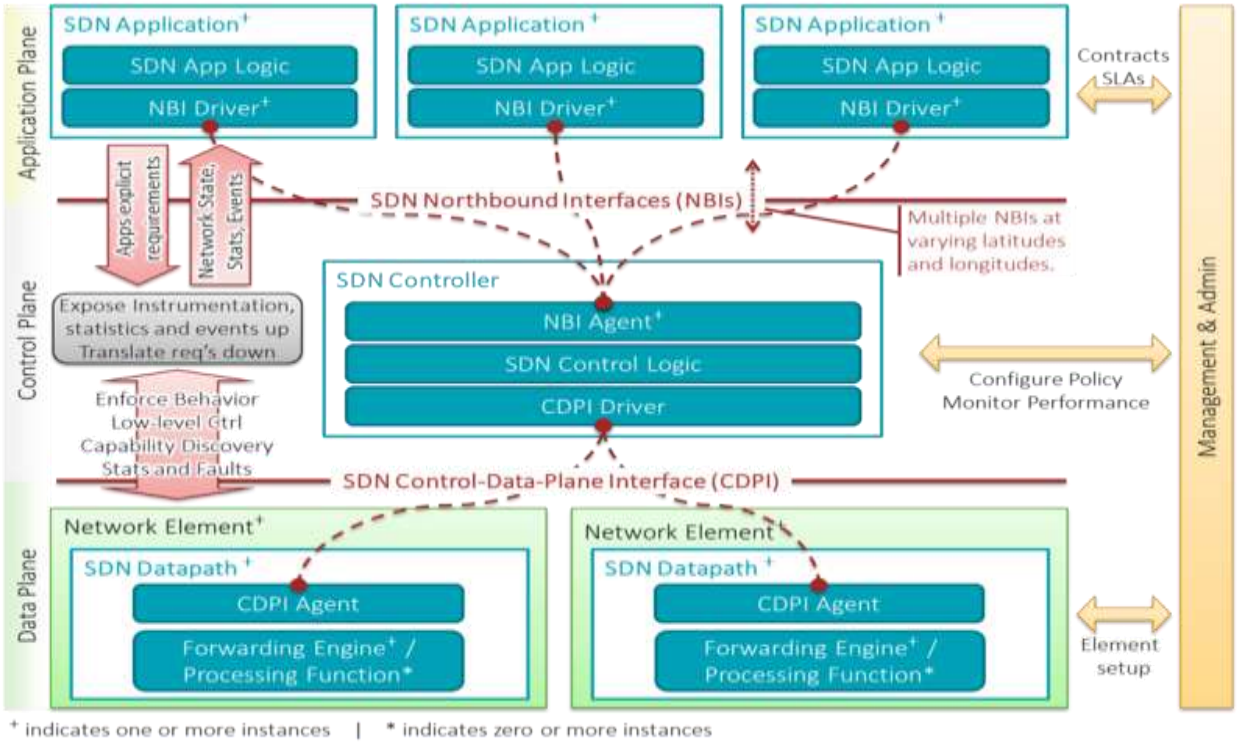
"Big data" means more bandwidth:

Handling today's "big data" or mega datasets requires massive parallel processing on thousands of servers, all of which need direct connections to each other. The rise of mega datasets is fueling a constant demand for additional network capacity in the data center. Operators of hyperscale data center networks face the daunting task of scaling the network to previously unimaginable size, maintaining any-to-any connectivity without going broke.

International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No. (01), January 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

IV. ARCHITECTURAL COMPONENTS

The following figure depicts the architectural components of SDN.



A high-level overview of the software-defined networking architecture

SDN Application:

SDN Applications are programs that explicitly, directly, and programmatically communicate their network requirements and desired network behavior to the SDN Controller via a northbound interface (NBI). An SDN Application consists of one SDN Application Logic and one or more NBI Drivers.

SDN Controller:

The SDN Controller is a logically centralized entity in charge of (i) translating the requirements from the SDN Application layer down to the SDN Datapaths and (ii) providing the SDN Applications with an abstract view of the network (which may include statistics and events). An SDN Controller consists of one or more NBI Agents, the SDN Control Logic, and the Control to Data-Plane Interface (CDPI) driver.

SDN Datapath:

The SDN Datapath is a logical network device that exposes visibility and uncontested control over its advertised forwarding and data processing capabilities. The logical representation may encompass all or a subset of the physical substrate resources. An SDN Datapath comprises a CDPI agent and a set of one or more traffic forwarding engines and zero or more traffic processing functions. These engines and functions may include

simple forwarding between the datapath's external interfaces or internal traffic processing or termination functions.

SDN Control to Data-Plane Interface (CDPI):

The SDN CDPI is the interface defined between an SDN Controller and an SDN Datapath, which provides at least (i) programmatic control of all forwarding operations, (ii) capabilities advertisement, (iii) statistics reporting, and (iv) event notification. One value of SDN lies in the expectation that the CDPI is implemented in an open, vendor-neutral and interoperable way.

SDN Northbound Interfaces (NBI):

SDN NBIs are interfaces between SDN Applications and SDN Controllers and typically provide abstract network views and enable direct expression of network behavior and requirements. This may occur at any level of abstraction (latitude) and across different sets of functionality (longitude).

SDN Control Plane:

Centralized - Hierarchical – Distributed:

The implementation of the SDN control plane can follow a centralized, hierarchical, or decentralized design. Initial SDN control plane proposals focused on a centralized solution, where a single control entity has a global view of the network. While this simplifies the implementation of the control logic, it has scalability limitations as the size and dynamics of the network increase.

Controller Placement:

A key issue when designing a distributed SDN control plane is to decide on the number and placement of control entities. An important parameter to consider while doing so is the propagation delay between the controllers and the network devices, [9] especially in the context of large networks. Other objectives that have been considered involve control path reliability [10] fault tolerance [11] and application requirements [12].

V. ADVANTAGES OF SDN:

Content Delivery:

Controlling data traffic is one of the primary advantages of software defined networking. The ability to direct and automate data traffic makes implementing quality of services (QoS) for voice over IP (VOIP), video and audio transmissions much easier. Software defined networking provides a seamless experience for end-users streaming high quality audio and video.

Lessen Capital Expenditure:

By implementing software defined networking, enterprise businesses can easily optimize existing network devices. Existing hardware can be repurposed to follow the instructions of a SDN controller, and more cost-efficient hardware can be deployed with greater effect. For example, a “white box” switch can be made using commoditized parts already owned by a business, preventing the purchase of proprietary switches.

Centralization:

Software defined networks offer a centralized view of an organizations entire network, making it easier to streamline enterprise management and provisioning. As VLANs become a more prominent part of physical LANs, the number of links and dependencies can easily create confusion. SDNs can speed service delivery and provide more agility for both virtual and physical network provisioning, all from a central location.

Management:

In order to accommodate big data, enterprise businesses are constantly setting up new applications and virtual machines to handle processing requests. By implementing SDN, IT teams are able to change network configurations with no effect to the network. Additionally, SDN supports the management of physical and virtual switches and network equipment from a single centralized controller, something that cannot be done using simple network management protocol (SNMP).

VI. SDN DEVELOPMENT TOOLS

Here, we have listed commercial switches that are currently available, their manufacturer, and the version of OpenFlow they implement

i. Switch Software and Stand-Alone OpenFlow Stacks:

Open vSwitch: Open vSwitch is an OpenFlow stack that is used both as a vswitch in virtualized environments and has been ported to multiple hardware platforms. It is now part of the Linux kernel. It is implemented in C/Python.

OpenFlow Reference: The OpenFlow reference implementation is a minimal OpenFlow stack that tracks the spec. It is implemented in C.

Pica8: An open switch software platform for hardware switching chips that includes an L2/L3 stack and support for OpenFlow. It is implemented in C.

Indigo: Indigo is a for-hardware-switching OpenFlow implementation based on the Stanford reference implementation. It is implemented in C.

Pantou: (C) Pontou is an OpenFlow port to the OpenWRT wireless environment.

OpenFaucet: (Python) OpenFaucet is a pure Python implementation of the OpenFlow 1.0.0 protocol, based on Twisted. OpenFaucet can be used to implement both switches and controllers in Python. It is implemented in Python.

OpenFlowJ: OpenFlow stack written in Java.

Oflib-node: Oflib-node is an OpenFlow protocol library for Node. It converts between OpenFlow wire protocol messages and Javascript objects.

ii. Controller Platforms:

POX: Pox as a general SDN controller that supports OpenFlow. It has a high-level SDN API including a queriable topology graph and support for virtualization. It is implemented in Python.

IRIS: A Resursive SDN Openflow Controller created by IRIS Research Team of ETRI. Our vision was to create an SDN controller platform with the following features: (a) Horizontal Scalability for carrier-grade network (b) High Availability with transparent failover from failure (c) Multi-domain support with recursive network abstraction based on Openflow. It is implemented in Java.

MUL: MūL, is an openflow (SDN) controller. It has a C based multi-threaded infrastructure at its core. It supports a multi-level north bound interface for hooking up applications. It is designed for performance and reliability which is the need of the hour for deployment in mission-critical networks. It is implemented in C.

NOX: NOX was the first OpenFlow controller. It is implemented in C++/Python.

Jaxon: Jaxon is a NOX-dependent Java-based OpenFlow Controller. It is implemented in Java.

Trema: Trema is a full-stack framework for developing OpenFlow controllers in Ruby and C.

Beacon: Beacon is a Java-based controller that supports both event-based and threaded operation.

Floodlight: The Floodlight controller is Java-based OpenFlow Controller. It was forked from the Beacon controller, originally developed by David Erickson at Stanford.

Maestro: Maestro is an OpenFlow "operating system" for orchestrating network control applications.

NDDI: OESS: OESS is an application to configure and control OpenFlow Enabled switches through a very simple and user friendly User Interface.

Ryu: Ryu is an open-sourced Network Operating System (NOS) that supports OpenFlow. It is implemented in Python.

iii. Special Purpose Controllers:

RouteFlow: RouteFlow is an open source project to provide virtualized IP routing services over OpenFlow enabled hardware. RouteFlow is composed by an OpenFlow Controller application, an independent RouteFlow Server, and a virtual network environment that reproduces the connectivity of a physical infrastructure and runs IP routing engines (e.g. Quagga).

Flowvisor: FlowVisor is a special purpose OpenFlow controller that acts as a transparent proxy between OpenFlow switches and multiple OpenFlow controllers.

Resonance: Resonance is a Network Access Control application built using NOX and OpenFlow.

Oflops: OFlops (OpenFLow Operations per Second) is a standalone controller that benchmarks various aspects of an OpenFlow switch.

VII. APPLICATIONS

Software-defined networking has applications in a wide variety of networked environments. By decoupling the control and data planes, programmable networks enable customized control, an opportunity to eliminate middleboxes, as well as simplified development and deployment of new network services and protocols**.**

i. Cellular Networks:

Applying the SDN principles to cellular networks solve some of the deficiencies. First of all, decoupling the control from the data plane and introducing a centralized controller that has a complete view of the whole

network allows network equipment to become simpler and therefore reduces the overall infrastructural cost. Also, operations like routing, real-time monitoring, mobility management, access control and policy enforcement can be assigned to different cooperating controllers making the network more flexible and easier to manage. In addition, using a centralized controller acting as an abstract base station simplifies the operations of load and interference management, no longer requiring the direct communication and coordination of base stations. Instead, the controller makes the decisions for the whole network and simply instructs the data plane (i.e. the base stations) on how to operate. One final advantage is that the use of SDN eases the introduction of virtual operators to the telecommunications market, leading to increased competitiveness.

ii. Data Center Networks:

One of the most important requirements for data center networks is to find ways to scale in order to support hundreds of thousands of servers and millions of virtual machines. However, achieving such scalability can be a challenging task from a network perspective. Initially, the size of forwarding tables increases along with the number of servers, leading to a requirement for more sophisticated and expensive forwarding devices. Moreover, traffic management and policy enforcement can become very important and critical issues, since datacenters are expected to continuously achieve high levels of performance. Additionally, it becomes increasingly difficult to make the data center operate at its full capacity, since it cannot dynamically adapt to the application requirements. The advantages that SDN offers to network management come to fill these gaps. By decoupling the control from the data plane, forwarding devices become much simpler and therefore cheaper. At the same time all control logic is delegated to one logically centralized entity. This allows the dynamic management of flows, the load balancing of traffic and the allocation of resources in a manner that best adjusts the operation of the data center to the needs of running applications, which in turn leads to increased performance [13]. Finally, placing middleboxes in the network is no longer required, since policy enforcement can now be achieved through the controller entity.

iii. Enterprise Networks:

Enterprises often run large networks, while also having strict security and performance requirements. Furthermore, different enterprise environments can have very different requirements, characteristics, and user population. Adequate management is critically important in Enterprise environments, and SDN can be used to programmatically enforce and adjust network policies as well as help monitor network activity and tune network performance.

Additionally, SDN can be used to simplify the network

VIII.CONCLUSION

Recent developments in ICT domain, such as mobile, multimedia, cloud, and big data, are demanding for more convenient Internet access, more bandwidth from users, as well as more dynamic management from service providers. SDN is considered as a promising solution to meet these demands. In this paper, we have presented the concept of SDN and highlighted benefits of SDN and its applications.

REFERENCES

[1] IDC Predictions 2013: Competing on the 3rd Platform, IDC, Framingham, MA, USA, Nov. 2012,WhitePaper.[Online]. Available:http://www.idc.com/research/Predictions13/downloadable/238044.pdf

[2] P.Mell and T. Grance, "The NIST definition of cloud computing (draft),"NIST Special Publication, vol. 800-145, p. 7, 2011.

[3] J. Gantz and D. Reinsel, "Extracting value from chaos," IDC, Framingham, MA, USA, White Paper, Jun. 2011. [Online]. Available:http://www.emc.com/collateral/analyst-reports/idc-extracting-valuefrom-chaos-ar.pdf

[4] J. Manyika et al., "Big data: The next frontier for innovation, competition, and productivity," McKinsey Global Inst., Mumbai, India,pp. 1–137, 2011.

[5] P. Cesar and D. Geerts, "Past, present, and future of social TV: A categorization, "in Proc. IEEE CCNC, 2011, pp. 347–351.

[6] Marc Mendonca and Xuan-Nam Nguyen et al., "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks", Communications Surveys and Tutorials, IEEE Communications Society, Institute of Electrical and Electronics Engineers, 2014, 16 (3), pp.1617-1634.

[7] "Software-Defined Networking[SDN] Definition" Opennetworking.org, Retrieved 26 October 2014.

[8] R. Ahmed, R. Boutaba, "Design considerations for managing wide area software defined networks," Communications Magazine, IEEE, vol. 52, no. 7, pp. 116–123, July 2014.

[9] B. Heller, R. Sherwood, and N. McKeown, "The Controller Placement Problem", proceedings of HotSDN'12, 2012.

[10] Y.N. Hu, W.D. Wang, X.Y. Gong, X.R. Que, S.D. Cheng, "On the placement of controllers in software-defined networks", Journal of China Universities of Posts and Telecommunications, vol. 19, Supplement 2, pp. 92 – 171, 2012.

[11] F.J. Ros, P.M. Ruiz, "Five nines of southbound reliability in software defined networks", proceedings of HotSDN'14, 2014.

[12] D. Tuncer, M. Charalambides, S. Clayman, G. Pavlou, "On the Placement of Management and Control Functionality in Software Defined Networks", proceedings of 2nd IEEE International Workshop on Management of SDN and NFV Systems (ManSDN/NFV), Barcelona, Spain, November 2015.

[13] Al-Fares, Mohammad, et al., "Hedera: Dynamic Flow Scheduling for Data Center Networks," 7th USENIX Symposium on Networked Systems Design & Implementation, 10 (2010): 19-34.