**Software Defined Networks-A working Principles, Operations by using Bottom-up Approach and Alternate methods**

**Dr Ninu SB,Dr V.Devi,Dr Sathyamoorthy ,Mrs C.Vijayalakshmi,**

**Associate Professor,Thiruthangal Nadar College,**

**Principal, Thiruthangal Nadar College**

**Associate Professor,RMKV**

**Assistant professor Thiruthangal Nadar College**

**Abstract:**

**SDN is a developing architecture that is dynamic, manageable, cost effective and adaptable making it ideal for the high bandwidth, nature of today’s applications.SDN that aims to improve the control of network and flexibility. The architecture of SDN decouples the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services. In this chapter we provide an overview of how SDN Works, including the basic components of software defined networking system, their roles and how they interact with each other. The SDN operation involves the SDN device, the controller and the applications by using the bottom-up approach method. An SDN device is composed of an API for communication with the controller, an abstraction layer and a packet processing function. The flow tables are the fundamental data structures in an SDN device, which allow the device to evaluate incoming packets and take the action based on the content of the packet that has been received.**

**The first part of this chapter focus the methods used by the open SDN.We also examine how alternate forms of SDN work. The later half of the chapter focus the alternate SDN implementation in 2 Categories SDN via API’s and SDN via Hypervisor-Based Overlay Networks.**

**This chapter described the basic functionality related to the manner in which an SDN solution actually works. It is important to realize that there is no fundamental incompatibility between the hypervisor-based overlay network approach to SDN and open SDN.It is concluded that these overlay networks as stepping stones toward a more complete SDN solution which includes SDN and Openflow for addressing both the virtual as well as the physical needs of the network.**

**Keywords: SDN, Plane Separation, Flow Tables, SDN Controller, SDN via APIs, Hypervisor Based Overlay Networks**

|  |  |  |
| --- | --- | --- |
| **Software Defined Networks-A working Principles, Operations by using Bottom-up Approach and Alternate methods** | | |
| **Chapter NO** | **Title** | **Page No** |
| **1.1**  **1.1.1**  **1.1.2**  **1.1.3**  **1.1.4** | **Fundamental Characteristics of SDN**  **Plane Separation**  **Simple Device and Centralized Control**  **Network Automation and Virtualization**  **Openess** | **03**  **03**  **03**  **03**  **04** |
| **1.2** | **SDN Operations** | **04** |
| **1.3**  **1.3.1**  **1.3.2**  **1.3.3** | **SDN Devices**  **Flow Tables**  **SDN Software Switches**  **Hardware SDN Devices** | **06**  **06**  **06**  **07** |
| **1.4**  **1.4.1**  **1.4.2** | **SDN Controller**  **SDN Controller Core Modules**  **SDN Controller Interfaces** | **07**  **07**  **08** |
| **1.5**  **1.5.1** | **SDN Applications**  **SDN Application Responsibilities** | **09**  **09** |
| **1.6**  **1.6.1**  **1.6.2**  **1.6.3** | **Alternate SDN Methods**  **SDN via APIs**  **BENEFITS AND Limitations of SDN via APIS**  **SDN via Hypervisor-Based Overlay Networks** | **09**  **09**  **11**  **12** |
| **1.7** | **Conclusion** | **13** |

**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.**

**1.1 Fundamental Characteristics of SDN:**

**Software defined networking, is characterized by five fundamental traits (i) Plane Separation (ii) a simplified device (iii) Centralized control (iv) Network automation & virtualization (v) Openess.**

* + 1. **Plane Separation:**

**The separation of the forwarding and control planes is the first fundamental characteristic of SDN.Based on the properties MAC address, IP Address and VLAN ID the logic and tables are used to deal with incoming packets. The action of forwarding plane describes how it dispenses with arriving packets. The actions may be forward, drop, consume or replicate an incoming packet. To forward the device obtain the correct output port by lookup in the address table of hardware ASIC.Due to buffer overflow conditions a packet may be dropped or due to filtering which result Qos rate –limiting function.**

**The logic and algorithm are applied to program the forwarding plane which reside in the control plane.to require the global knowledge of the network the protocols and algorithms are used. The control plane determines how the tables and logic in the data plane should be programmed. The synchronized factor is applicable for all the distributed forwarding tables on the devices in the network.it may avoid the prevention of loops.**

* + 1. **Simple device and Centralized control:**

**The simplification of devices are controlled by a centralized system which run under management and control software. The software is placed on a centralized controller that manage the network .The primitive operations are provided by the controller to the devices in order to allow them to adopt fast decisions about how to deal with incoming packets.**

* + 1. **Network automation and Virtualization:**

**The SDN can be derived precisely by the abstraction feature of distributed state, forwarding and configuration. The distributed state provides the network programmer with a global network view, each with its own state, collaborating to solve network wide problems. The forwarding abstraction makes the programmer to specify the forwarding behaviors without any knowledge of vendor –specific hardware.Atlast the configuration abstraction, are also known as specification abstraction to express the goals of the entire network without losing the details of how the physical networks will implement those goals. Working with the network by using the configuration is network virtualization as its basic level.**

* + 1. **Openess:**

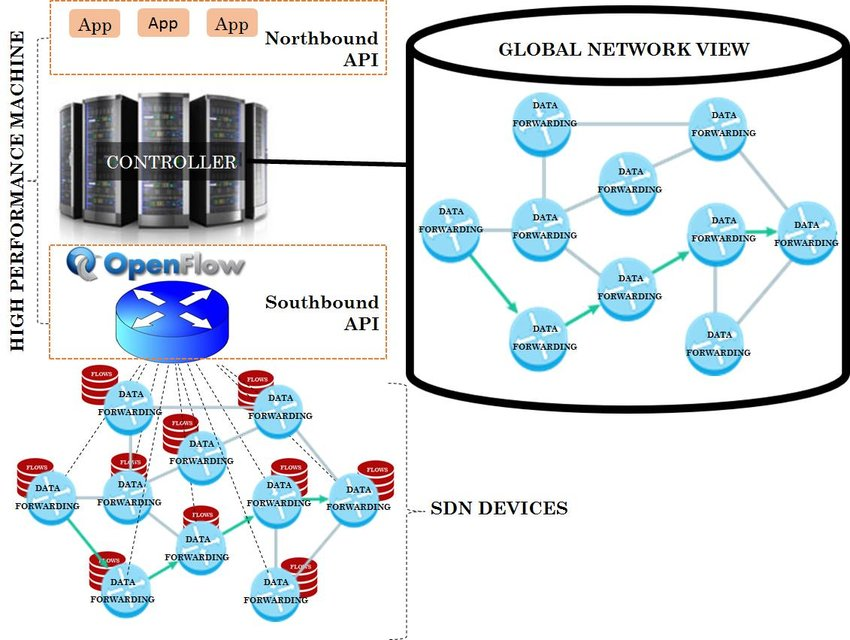
**An open SDN should remain standard, well documented and nonproprietary. The APIs should give software sufficient control to experiment with and control various control plane options. Researchers can take advantage of this capability to test new ideas. Enormous growth of network, the individuals and organizations are applied themselves, resulting better and faster in the structure and functioning of networks.**

* 1. **SDN Operation:(Bottom –up Approach)**

**The behavior and operation of SDN is straightforward in concept level. The basic components of SDN are (i) the SDN devices (ii) the Controller (iii) the applications. The bottom –up strategy is applied to understand the operation of SDN.**

**In fig 1.1, the SDN devices includes the forwarding functionality to decide what to do with the incoming packet. The upper left portion of each device flows defined by the controller actually represent its data. A flow describes a set of packets from one endpoint to other endpoint. A flow is unidirectional, and represented on a device as a flow entry.**

**A flow table placed on the network device and consists of flow entries and actions to perform when a packet match the flow arrives at the device. To find the match the SDN device consult the packet with flow table. If match occurs, it takes the appropriate action which forward the packet. If it fails to find match, the switch can either drop or pass to the controller.**

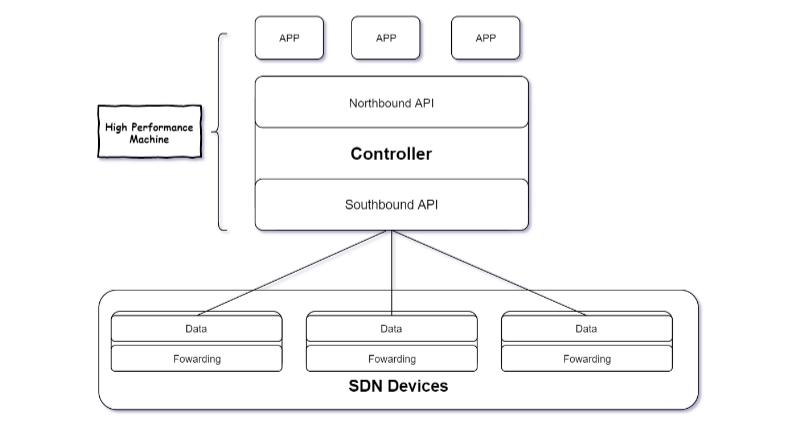


**Fig 1.1 SDN Operation Overview**

**The SDN controller is responsible for abstracting the network of SDN devices it controls and presenting an abstraction of these resources to the SDN applications. The controller allows the SDN application to define flows on devices and help the application to respond the packets which are forwarded to the controller by the SDN devices. It calculate the optimal forwarding solutions for the network in deterministic, predictable manner.**

**On the top of the controller the SDN applications are built. The SDN application interface with the controller by using proactive flows on the devices and to receive packets which has forwarded to controller. Proactive flow is also known as static flow will persist until some changes made in configuration. Reactive flow will establish new flows on the device in order to allow that device to respond to the packet.**

* 1. **SDN Devices:**

**In fig 1.2, for physical switch the packet processing function is embedded in the hardware for packet processing logic. **

**Fig 1.2 SDN Devices**

**The abstraction layer includes one or more flow tables. The mechanism which applied by the packet processing logic is to take actions based on the results of incoming packets and find the highest priority match. If match is found, the incoming packet is processed locally. If else, the packets may be copied to the controller for further processing.**

**1.3.1 Flow Tables:**

**Flow table is a fundamental data structure which allow the device to evaluate incoming packets and placed the proper action. The action may be forward, dropping and flooding. It include a prioritized flow entries, which consist of two components (i) match fields (ii) actions. Based on priority the incoming packet compared with match field and the selection is made with proper action.**

**1.3.2 SDN Software Switches:**

**Software SDN device are found in software based network devices such as the hyper visions of a virtualization system. These hyper visions often incorporation a software switch implementation which connects the various virtual machines to the virtual network. The virtualization system is often controlled a centralized management system.**

**1.3.3 Hardware SDN Devices:**

**To understand how SDN objects can be translated into hardware, we will briefly review the hardware components of today’s networking devices. The specialized hardware includes the layer 2 and layer 3 forwarding tables often implemented using content-Addressable memories (CAMS) and Ternary content-Addressable memories (TCAMs).for making IP-level routing decisions the layer 3 is used.it matches the destination IP address against the entries in the table and take the appropriate action.**

**A series of challenges are focus here,**

**(i) How best to translate from flow entries to hardware entries.**

**(ii) Which of the flow entries to handle in hardware.**

**(iii) How to deal with hardware action limitations which may impact whether to implement the flow in hardware vs. software.**

* 1. **SDN Controller**

**The SDN controller implements policy decisions, controls all the SDN devices that includes the network infrastructure .Controllers often come with their own set of common application modules such as a learning switch, a router, a firewall and a simple load balance.**

**1.4.1 Core Modules:**

1. **End-user Device**

**End users devices are laptops, desktops, printer, Mobile etc.**

1. **Network Device**

**The network devices are switches, routers, wireless access points etc.**

1. **Topology**

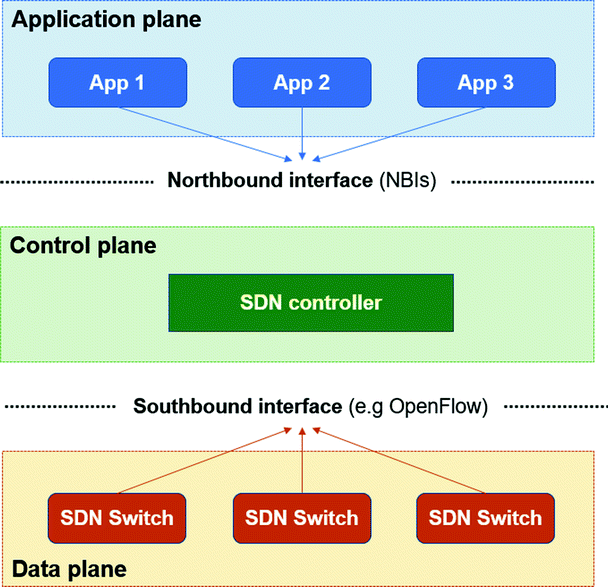
**Concentrates on physically and logically laid out of the network devices.**

1. **Flow management**

**Maintain synchronization of the device flow entries with the database.**

**1.4.2 SDN Controller Interface:**

**For accessing the network the API act as key function which provide by SDN controller.**



**Figure 1.3 SDN Controller**

**In fig 1.3, the controller informs the application of events that occur in the network and events to the application. For the operation of the network the application use different methods.**

**1.4.3 Issues in SDN Controller:**

1. **Suffers towards the adaptation of new technology.**
2. **Demanding network will be high in real-life applications.**
3. **Require more heterogeneous mix of equipment types.**
4. **The challenges of coordinator between application the lack of standard Northbound API and flow prioritization.**

**1.5 SDN Applications:**

**Above of SDN Controller there is an SDN applications which interface to the network through the northbound API.SDN applications are responsible for managing the flow entries. By using this Application able to,**

* **Find the best path and configure the flows to route packets between the endpoints.**
* **For Multiple paths, it balance the traffic.**
* **React to the dynamic topology.**
* **Redirect the traffic.**

**Responsibilities**:

* **To perform whatever function for which it was designed.**
* **Application spend its processing time to respond the events.**
* **Affects the network by responding the events.**
* **The method will be invoked when event occurs.**
* **Incoming packet events are sent to the SDN applications** 
  1. **Alternate SDN methods:**

**The two alternate SDN methods are (i) SDN via APIs (ii) SDN via Hypervisor-Based Overlay Networks.**

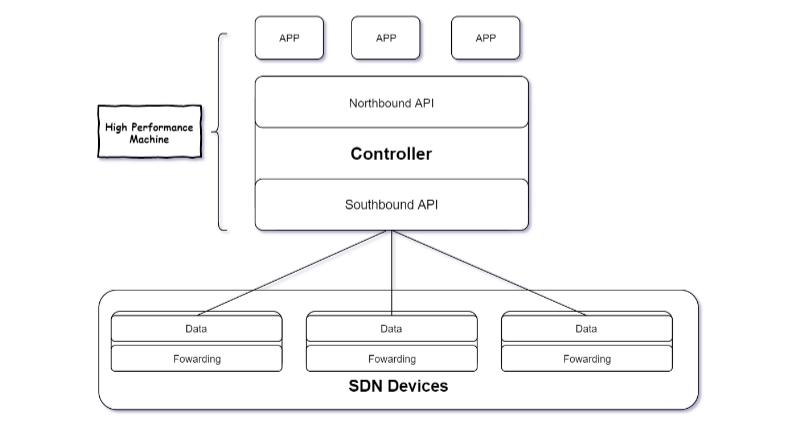
* + 1. **SDN via APIs**

There are number of levels at which the APIs can be provided for SDN applications.

* SDN via Device APIs
* SDN via Controller APIs
* SDN via Policy APIs

**(i) SDN via Device APIs:**

**The strategy is to provide a richer set of control points on the devices, so that centrally located software can manipulate those devices and provide the intelligent and predictable behavior that is expected in an SDN-controlled network.**



**Fig 1.4 Device APIs**

**The Fig 1.4 shows a set of centralized applications communicating with devices with device level APIs.Interface to the routing system (I2RS) provides an interface between routing protocols and the Routing Information Base(RIB).The four key drivers of this protocol are,**

(i)**The need for an interface that is programmatic, asynchronous, and offers fast, interactive access for atomic process.**

**(ii)To provide access to structured routing information and state that is frequently not directly configurable or modeled in existing implementation or configuration protocols.**

**(iii) To provide the ability for network management and other applications to subscribe to structured filterable event notifications from the routing system.**

**(iv)To facilitate extensibility and provide standard data-models to be used by network applications.**

**(ii) SDN via Controller APIs:**

**To build SDN applications SDN via Controller-level APIs provide a platform. These APIs are open and available for application developers.**

**In SDN via Controller APIs,the southbound protocol consists of one or more protocols for functionality .one goal of implementing APIs at the controller level is to provide a level of abstraction between the devices and the application.one example of this case is collecting topology information about devices in the network and presenting it via the controller API.In other cases, the controller performs only light processing of the messages between the application and device.NETCONF is one of the device management and configuration protocols are used by vendors.**

**(iii) SDN via Policy APIs:**

**The API which reside at above layer of controller level. The network configuration form a declarative rather than imperative.**

**(i)Imperative: It require the user to input exactly how to do a particular task.**

**(ii)Declarative: It require the user to input exactly what is to be accomplished**

**1.6.2 Benefits:**

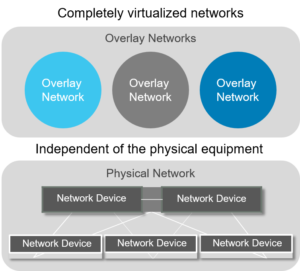
* **Does not require upgrading to Openflow enabled switches.**
* **Easy to write software which respond quickly and automatically changes in network.**
* **Possible to build an SDN solution using the provided APIs on the distributed network devices**
* **Potential for increased Openess in the SDN via APIs approach**

**Limitations:**

* **In some cases, there is no controller, the network programmer needs to interact directly with each switch.**
* **Even controller, it may not provide an abstract, network wide view to the programmer**
* **The programmer developing applications on top of that controller must synchronize with what the distributed control plane is doing.**

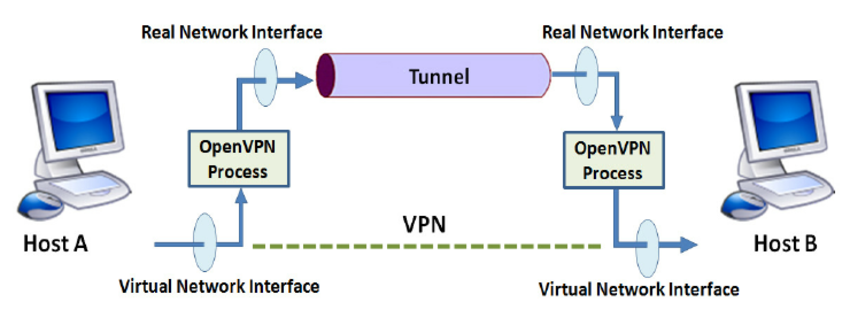
**1.7 SDN via Hypervisor-Based Overlay Networks:**

**The systems at the edges of the network interact with the virtual networks, which obscure the details of the physical network from the devices that connect to the overlays. The virtual network traffic runs above the physical network infrastructure. The hypervisors inject traffic into the virtual network and receive traffic from it.The traffic of the virtual network is passed through those physical devices which can be controlled entirely by the devices at the very edge of the network is called a Tunnel Endpoint.**



**Fig 1.5 Hyper based Relay networks**

**The hypervisor takes the encapsulated packet, based on information programmed by the controller, sends it to the destination’s TP.This TP decapsulates the packets and forward it to the destination host. It sent from the TP source to TP destination. This tunneling mechanism is referred to as MAC-in-IP tunneling.**



**Fig 1.6 Tunneling**

**Fig1.6 shows the roles of these TP as they serve the source and destination host devices. The virtual network capability is typically added to a hypervisor by extending it with a virtual switch. The virtual network has a virtual topology, consisting of the virtual switches interconnected by virtual point-to-point links.**

**In summary, SDN via Hypervisor-Based Overlay Networks is well suited to environments such as data centers already running compute and storage virtualization software for their servers.it does address a number of the needs of an SDN solution. First it address the MAC address explosion in data centers and cloud environments. Second, it addresses VLAN limitations, because all traffic is tunneled and VLANs are not required for supporting the isolation of multiple tenants.Third,it address agility and automation needs, because it is implemented in software and these virtual networks can be constructed and taken down in a fraction of the time that would be required to change the physical network infrastructure.**

**1.8 Conclusion:**

**This chapter described the basic functionality related to the manner in which SDN solution actually works.it is important to realize that there is no fundamental incompatibility between the hypervisor-based overlay network approach to SDN and open SDN.In fact, some implementations use Openflow to create and utilize the tunnels required in this kind of network virtualization. It is not reasonable to think of these overlay networks as stepping stones toward a more complete SDN solution which includes SDN and Openflow for addressing both the virtual as well as the physical needs of the network.**

References:

[1] Shenker S.The future of networking, and the past of protocols. Open networking summit.oct 2019.

[2] Kerner Brocade Doubles Down on Open SDN.Enterprise Networking Planet. May 23 2016.

[3] Open flow management and configuration protocol 2015

[4] Wang Floodlight documentation .project floodlight Dec 2013.

[5] Nox.Retrieved from: http://www.norepo.org

[6] Open daylight technical overview .http://www.op endaylight.org

[7] Ruby: a programmer best friend: http://www.ruby-lang.org

[8] Learn rest:http://www.restapitutorial.com

[9] Trema: Openflow framework in ruby:http://trema.github.io/trema