Unit 1 Introduction to SDN: An Overview

1.1 Traditional Network

A traditional network, or computer network, or data network, or conventional network is a digital telecommunication network that allows various nodes to communicate in order to exchange or share data with each other. This data linking can be established by cable media like wires or optic cables or wireless media. The network nodes are the network devices that generates, route and terminate data. They include hosts such as servers, phones, personal computers, network hardware.

Traditional network consists of three planes (Figure 1):

1. Control Plane
2. Management Plane
3. Data Plane

Refer section 1.4 on more details about these planes.

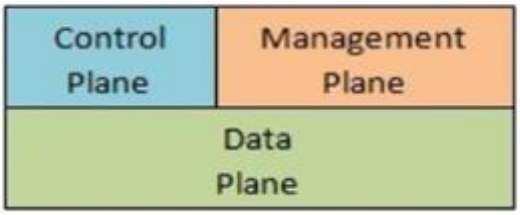

Figure 1 Tradition Network Device

The traditional networks uses special algorithms that are implemented on dedicated devices (hardware components) for controlling and monitoring the data flow in the network, managing routing paths and algorithms. These algorithms and set of rules are implemented in dedicated hardware components, such as Application-Specific Integrated Circuits (ASIC). ASIC are designed for performing specific operations. Packet forwarding is a simple example of this operation. In a traditional network, upon the reception of a packet by a routing device, it uses a set of rules embedded in its firmware to find the destination device as well as the routing path for that packet. Generally, data packets that are supposed to be delivered to the same destination are handled in similar manner and are routed through the same path irrespective of the data types of different packets. This operation takes place in inexpensive routing devices. More expensive routing devices can treat different packet types in different manners based on their nature and contents.

A problem posed by this is the limitation of the current network devices under high network traffic, which poses severe limitations on network performance. Issues, such as the increasing demand for scalability, security, reliability, and network speed, can severely hinder the performance of current network devices because of the ever increasing network traffic. Current network devices lack the flexibility

to deal with different packet types with various contents because of the underlying hardwired implementation of routing rules.

A possible solution to this problem is implementing data handling rules as software modules rather than embedding them in hardware. This enables network administrators to have more control over the network traffic and, therefore, has a great potential to greatly improve the performance of the network in terms of efficient use of resources and speed. Such an approach is defined in software-defined networking (SDN). In SDN, data handling is isolated from the hardware, and its control is implemented in a software module called the controller. **The basic idea behind SDN is to separate the control of data handling in the networking stack from the hardware and implement it in the software. This results in improved network performance in terms of network management, control, and data handling. Thus, for the conventional network, SDN is potential solution which is gaining more acceptance in applications, such as cloud computing. It can be used in data centers and for workload-optimized systems.**

1.1.1 Limitation of Traditional Network

Conventional networking architectures have widespread limitations that have to be overcome to satisfy modern IT necessities. Today's network must scale to deal with increased workloads with extra agility, even as also retaining expenses at a minimal. But the conventional approach has substantial limitations:

1. **Complexity:** The network complexity has greatly increased due to the number of protocols and their features for specific use case. Old technologies were often recycled as quick fixes to deal with new business necessities. Feature of network devices are intended to be vendor specific or are implemented through proprietary commands.
2. **Inconsistent policies:** In the current network there is need to manually configure the security and quality-of-service (QoS) policies or script across hundreds or thousands of network devices. This process makes policy changes extremely complicated for organizations. This manual configuration process in network takes many hours of troubleshooting and are prone to error. When an application is removed, it is impossible to remove all the associated policies from the devices, which increases additional complexity.
3. **Inability to scale:** Day by day application workloads change and demand for high network bandwidth. The IT department either needs to grow with the demands of the organization or needs to be satisfied with an oversubscribed static network. Undesirably, the majority of conventional networks are statically provisioned in any such manner that increasing the variety of endpoints, services, or bandwidth requires substantial planning and redesign of the network.

1.2 Software Defined Network (SDN)

SDN enables network administrators to deal with data in the network in a more efficient and innovative manner by separating control from the forwarding element. By using SDN, administrators have the ability to control data flow as well as to alter the characteristics of the switching devices (routing devices) in the network from a central location, with control application implemented as software module, without the need of dealing with each device individually. This enables network administrators to change routing tables (routing paths) in network routing devices. This also gives an extra layer of control over the network data because the administrator can assign high/low priority to certain data packets or allow/block packets flowing through the network with different levels of control. As a result, the network traffic can be controlled in an efficient manner and, hence, can be used as a mechanism for traffic load management in networks.

Figure 2 depicts the basic architecture of SDN. It consists of two layers, one data layer along with open flow API, and, two controller layer. The controller layer is interfaced with controller housing both control plane and management plane. Above these layer exists application layer.

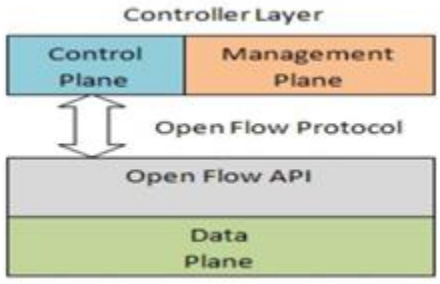

Figure 2 SDN Network

A potential application of SDN is in solutions, such as cloud computing, multitenant architecture, etc., as network administrators have more control over the network traffic and can use network resources in a more efficient manner. SDN is less expensive and provides more control over network traffic flow as compared with conventional network devices.

Several standards are used for the implementation of SDN. One of the most popular and widely accepted standards for SDN implementation is OpenFlow. OpenFlow provides remote control over network routing devices because of its ability to control the routing tables of the traffic in the network.

1.2.1 Difference between Traditional Network and SDN:

Figure 3 depicts the basic architecture difference between two networks for reference. In the network, control plane and data plane plays important role for data transmission for source to destination. As the data is available for routing, the control plan identify optimal path for data packet transmission and the data plane forward the data packet over the network as per the path defined by the control plane.

The traditional network all network devices (such as routers, switches) embedded with control plane and data plane and provide decentralized network control. In case of SDN, control plane and data plane is separated with the centralized controller which control the data plane consists number of forwarding elements (such as switches). SDN provides centralized network approach and enabling network management, configuration from centralized controller through programming.

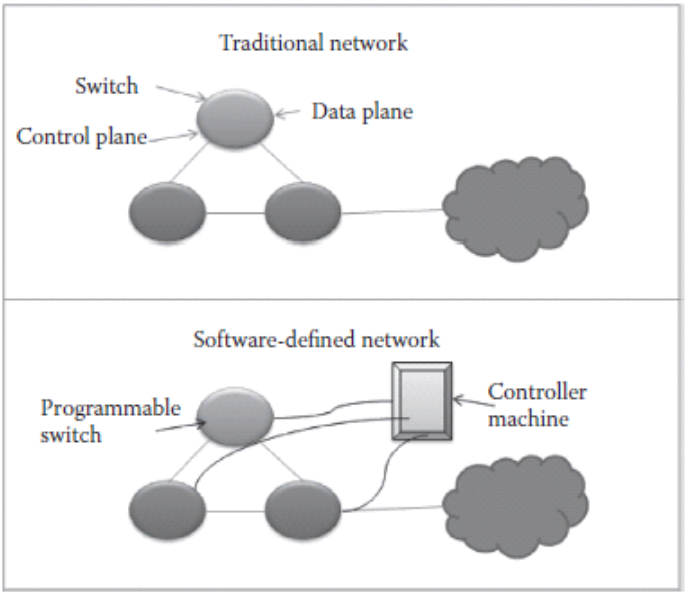

Figure 3 Traditional Networks and SDN Network

Below table describes difference between traditional network and software defining networking type:

Traditional Networking	Software Define Networking
They are Hardware appliances	Configured using open software
They have distributed control plane	They have logically centralized control plane
They are Static and inflexible networks. They are not useful for new business ventures. They possess little agility and flexibility	They are programmable networks during deployment time as well as at later stage based on change in the requirements. They help new business ventures through flexibility, agility and virtualization.
They use custom ASICs and FPGAs.	They use merchant silicon.
They work using protocols.	They use APIs to configure as per need.

1.2.2 Challenges of traditional Network

The traditional networks were built with hardware-based platforms optimized for precise features. These boxes consist of router, ethernet switches, wireless controllers, server load balancers and community safety appliances, such as firewalls and intrusion-detection structures. Network hardware typically runs complicated, distributed control software.

1. Provisioning

Conventional strategies of network provisioning, like manually configuring every device, can't scale to meet the complexity of distributed application. Network automation makes it possible to rapidly provision suitable network resources throughout dynamically moving workloads and lots of devices.

2. Configuration and change management

A lot of time and resources are being spend by network professionals in adapting the physical and virtual network to changes in applications, compute, storage resources and device location.

1.3 The Modern Data Center

A data center is a one type of repository that houses computing facilities like routers, switches, servers and firewalls. It generally include supporting components backup equipment that other organizations and businesses use to organize, process, store and disseminate large amounts of data. A data center may be simple like a room that houses only a few servers, or, complex like dedicated building.

In data center large numbers of servers are physically arranged into highly organized rows of racks of servers. Figure 4 shows today's data center which is depicted in a tree format consisting of three layers. The bottom layer is called access layer which represent top-of-rack (ToR) switch and provides the networking within the rack and the inter rack interface capability for that racks of compute-servers which are hierarchically organized. The middle layer is called Aggregation or Distribution layer consists of aggregation switch which provides connectivity to the adjacent ToR switches and data center rows. The top most layer is called Core layer has core switch which provides routing services to the other part of data center, as well as to services outside data center such as internet, geographically separated data centers and other remote locations.

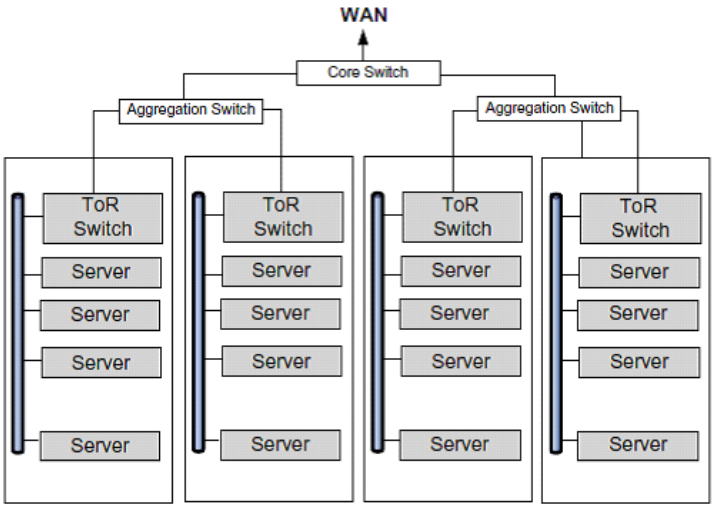

Figure 4 Data Center Network

During the evolution of data center, the number of computers in the data center grew rapidly. The data centers that are being built now will accommodate over 120,000 physical servers which can conceivably host 20 virtual machines (VMs) per physical server. This means that the internal network in such a data center would interconnect 2,400,000 hosts. These growing numbers of host computers within a data center will communicate with each other via a set of communications protocols and networking devices. The East-West traffic of data center is composed of packets sent by one host in a data center to another host in that same data center. Analogously, North-South traffic is traffic entering (leaving) the data center from (to) the outside world.

For example, a query say weather details from a user's web browser might be processed by a web server North-South that retrieves data from a storage server in the same data center (East-West) before responding to the user (North-South).

In data center rediscovering and recalculating routes for a network topology is highly static and only changed under strict centralized control. The increasing preponderance of East-West traffic does not benefit from the overhead and complexities that has evolved in traditional network switches to provide the decentralized survivability.

1.3.1 Data Center Need

The outburst of the speed and size of data centers has strained the capabilities of traditional networking technologies. Let's discuss the need of data center briefly

1. **Automation**

The ability to dynamically instantiate networks and to disable them when they are no longer needed is called automation. It is sometimes referred as agility. Thus, automation allows networks to come and go

at will. This must happen fast, efficiently, and with a minimum of human intervention. Not only do networks come and go, they also tend to expand and contract. It may require a dynamic, agile, and automated network to keep pace with changes.

2. Scalability

The connection of end stations to a single network has grown exponentially with data centers and cloud environments. The limitations of MAC address table sizes and number of VLANs has become barrier to network installations and deployments. The presence of large number of physical devices in data centers also pose broadcast control problem. The use of virtual networks and tunnels can contain the number of devices in a broadcast domain to a reasonable number.

3. Multipathing

Accompanying the large demands placed on the network by the scalability there is need for efficient and reliable network. That is, the network must make optimal use of its resources, and resists failures of any kind, and if failure occurs the network must be able to recover immediately.

4. Multitenancy

The requirement of hosting dozens or even hundreds or thousands of customers, or *tenants*, in the same physical data center is been created due to advances in data center technology, as well, the subsequent advent of cloud computing. A set of physical hardware hosting multiple tenants has been possible for some time in the server and storage area. Multitenancy implies that the data center has to provide each of its multiple tenants with its own (virtual) network that it can manage in a manner similar to the way it would manage a physical network.

5. Network Virtualization

The scale and fluidity introduced by server and storage virtualization has increased the urgency for automation, multitenancy and multipathing. The general idea of virtualization is that you create a higher-level abstraction that runs on top of the actual physical entity you are abstracting. The growth of compute and storage server virtualization has created demand for network virtualization. This means having a virtual abstraction of a network running on top of the actual physical network. Virtualization allows network administrator to create a network anytime and anywhere he chooses, as well as expanding and contracting networks that already exist. Intelligent virtualization software should be capable of this task without requiring the upper virtualized layer to be aware of what is occurring at the physical layer.

1.4 Planes of networking and Roles

The traditional Network Switch consists of Data, Control and Management Planes, Figure 5.

The Data Plane also called as forwarding Plane, comprises of the forwarding tables with its associated logic and ports for the reception and transmission of packets. The data plane also performs packet buffering, packet scheduling, and, header modifications and forwarding. The large amount of packets handled by switch is moved through the data plane.

After arriving of data packet at data plane (switch), header information checks with forwarding tables. If it's found then packets header gets modified and it will be forwarded. Sometime packets are handled in different way simply because of header information is not found in forwarding tables or because they belong to a control protocol that must be processed by the control plane.

The control plane is involved in many activities and responsible for processing a number of different control protocols which managing the active topology of the network. The main role of control plane to keep forwarding information in the table, so that data plane can handle huge amount of traffic independently as possible.

Management Plane helps to configure and monitor the switch which gives opportunities to extracts and modifies data in the data plane and control plane. The network administrator use some form of management system to communicate to the management plane in the switch.

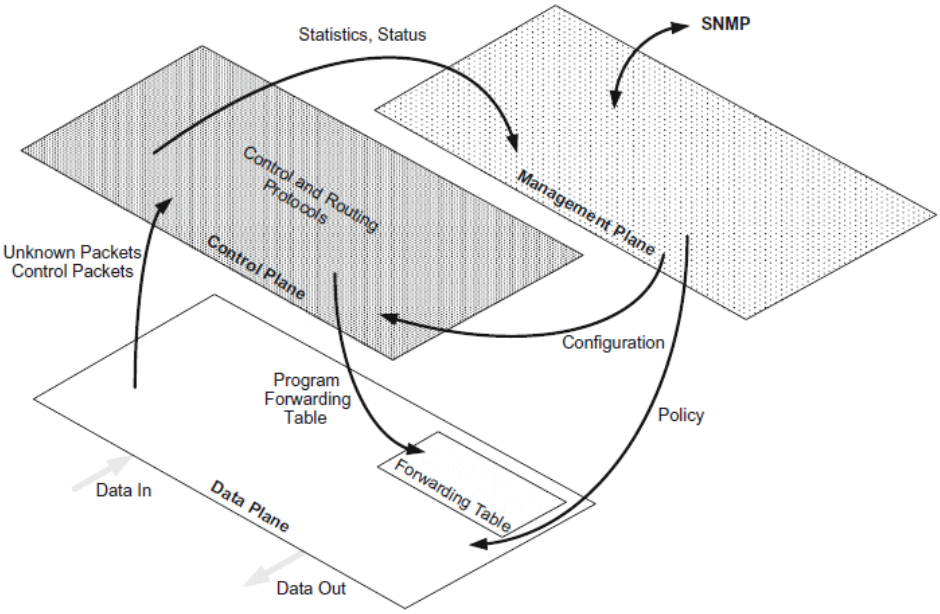

Figure 5 Role of Control, Management and data Plane

1.5 SDN Fundamental characteristics

1. **Plane Separation**

The separation of the forwarding (data plane) and control planes is the first fundamental characteristic of SDN. The forwarding Plane represent the forwarding functionality which includes the logic and tables for deciding how to deal with incoming packets based on characteristics such as MAC address, IP address, and VLAN ID, existing in the forwarding plane. The fundamental actions for incoming packet like forward, drop, consume, or replicate are performed by the forwarding plan and can be described by the way it dispenses with arriving packets. The control plane consists of all the protocols, logics and algorithms that are used to program the forwarding plane. Many of these algorithms and protocols require global knowledge of network. The main function of control plane is to determine how logic and forwarding tables in the data should be programmed or configured.

In the traditional network each device has its own control plane and data plane. The control plane is to run routing or switching protocols so that all the distributed forwarding tables on the devices throughout the network stay synchronized. The prevention of loops is the outcome of this synchronization. Traditionally the control and data plane have been considered logically separate, they co-reside in legacy Internet switches. In SDN, the control plane is separate from switching device and it act as centralized controller.

2. A Simple Device and Centralized Control

Structuring the idea of separation of forwarding and control planes, the next characteristics is the simplification of the devices. These devices are then controlled by the centralized system which run control plane and management software. In traditional network, network devices run hundreds of thousands of lines of complicated control plane software which allows device to behave autonomously. However, in SDN the software is removed from the device and is placed in a centralized controller which manages the network using higher level policies. Thus the devices when appropriate on order get the primitive instructions from controller allowing them to make fast decisions on dealing with incoming packets.

3. Network Automation and Virtualization

SDN can be derived precisely from the abstractions of distributed state, forwarding, and configuration. The distributed state abstraction provides the network programmer with a global network view that shields the programmer from the realities of a network that is actually comprised of many machines, each with its own state, collaborating to solve network wide problems. The programmer can specify the forwarding behaviors without any knowledge of vendor-specific hardware through forwarding abstraction. Finally, the configuration abstraction, which is sometimes called the specification abstraction, must be able to express he desired goals of the overall network without getting the details of how the physical network will implement those goals. Working with the network through this configuration abstraction is really network virtualization at its most basic level.

In SDN, the centralized software-based controller provides an open interface on the controller to allow for automated control of the network. In Open SDN, the terms northbound and southbound are often used to distinguish whether the interface is to the applications or to the devices. Generally, In most of the figures the applications are depicted above (i.e. to the north of) the controller, whereas devices are depicted below (i.e., to the south of) the controller. In OpenFlow southbound API controller programs the network device, and, in northbound API controller allows software application to be plugged into the controller to provide the algorithms and protocols that can run the network efficiently. These applications can quickly and dynamically make changes in the network as need arises. The northbound API of the controller is intended to provide an abstraction of the network devices and topology allowing the software to operate without knowledge of the individual characteristics and idiosyncrasies of the network devices themselves.

4. Openness

One of the characteristic of Open SDN is that its interfaces should remain standard, well documented, and not proprietary. The premise is that keeping open both the northbound and southbound interfaces to the SDN controller will allow for research into new and innovative methods of network operation. This capability used by research institutions as well as entrepreneurs in order to easily experiment with and test new ideas. Hence the speed at which network technology is developed and deployed is greatly increased as much larger numbers of individuals and organizations are able to apply themselves to today's network problems, resulting in better and faster technological advancement in the structure and functioning of networks. The presence of these open interfaces also encourages SDN-related open source projects. Similarly to facilitating research and experimentation, open interfaces permit equipment from different vendors to interoperate and this normally produces a competitive environment which lowers costs to consumers of network equipment.

1.6 Need of SDN

1. Increased Cost of Development

Present autonomous networking device must store, manage and run complicated control plane software. The result of these demands on the device is manifested in increased cost per device due to the processing power required to run that advanced software as well as the storage capacity to hold it.

2. Closed Environments Encourage Vendor Lock-in

Throughout the years, standards have been developed in the networking space for most relevant protocols and data that are used by switches and routers. Vendors do their best to implement these standards in a manner that allows heterogeneous networks of devices from multiple vendors to coexist with one another. In spite of good intension of vendors, enhancements are often made to these standard

implementation, which allow a vender's product to outperform its competitor. The end result of this will be smooth interoperability difficulty between products of different venders. Adherence to standards helps alleviate the issues associated to support multiple vendor types in a network. But problems with interoperability and management often far outweigh the advantages that might be gained by choosing another vendor. As a result customer sticks to vender they chose years or even decades before. This sort of vendor lock-in alleviates downward pressure on cost because the vendor is largely safe from competition and can thus preserve high profit margins.

3. Complexity and Resistance to Change

It is quite often that in networking a point comes where a network is made operational and from that point it better to leave thing without disturbing unless system breaks and requires restart. In spite of efforts of standardization, there is an argument to stay with single-vendor solution as there is only one vendor involved and vendor's accountability is not diluted. Hence, resistance to change results in long term technological stagnation and sluggishness. The ideal would be a simpler, more progressive world of networking, with open, efficient, and less expensive networking devices. This is a goal of SDN.

4. Increased Cost of Operating the Network

With further enhancements in networks, the operation expenses (OPEX) will increase and can be seen more significant than the corresponding capital expenses (CAPEX). SDN has the capacity to accelerate the automation of network management tasks in a multivendor environment. Provisioning of new services and agility to switch equipment between different services should lead to lower OPEX with SDN.

1.7 SDN Architecture

The nature of SDN architecture is dynamic, commercial, adaptable, and controllable; it is a designing model for the high bandwidth and dynamic nature of today's applications. This architecture decouples the network control (network intelligence) and data plane (perform forwarding functions) which enables the network control to be directly programmable, and the underlying infrastructure layer abstracted from applications. The SDN architecture shown in figure 6 consists three layers, Application Layer (Application Plane), Control Layer(Control Plane) and Infrastructure Layer(Data Plane) shown in Figure. The applications (Security, Bandwidth Management, and Load Management) exist in the application layer and send their network requirements send to control plan through a Northbound Interface (NBI). SDN uses NBI to communicate with the applications and business logic present in the application layer to help network executives to programmatically manage traffic and deploy the services.

In SDN, control plan is detached from the data plane. This control plane is a centralized software-based controller that maintains the centralized view of the overall network and enables the network administrator to direct the underlying system about how to forward the traffic.

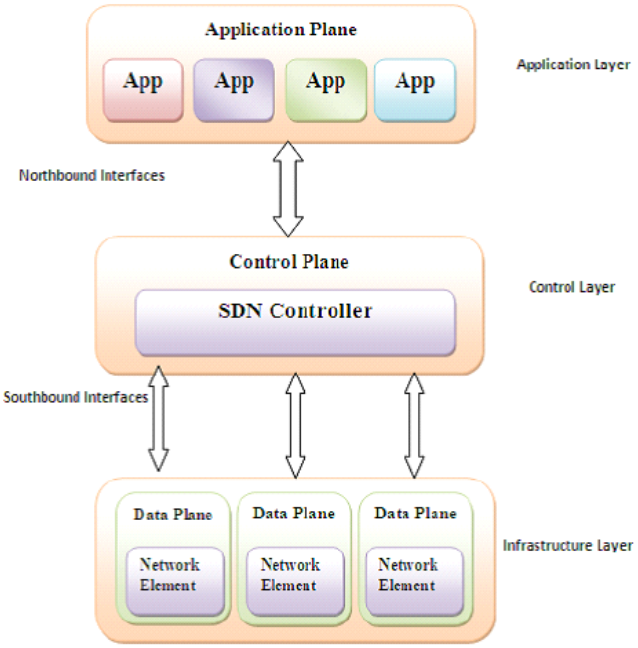

Figure 6 Layer Wise SDN Architecture

SDN Controller is the brain of the network that interacts with the data plane via a Southbound Interface (SBI), also called Control Data Plane Interface (i.e. OpenFlow). The SBI of SDN relays the information to the switches and routers deployed in a data plane. Data Plane (Forwarding Plane) consists of some network elements (e.g. Switches) responsible for forwarding the traffic as per the rules set by the controller. SDN architecture will provide a set of application programming interfaces that simplifies the implementation of network services like routing, multicasting, access control, etc. Such a SDN network structure allows a network administrator to implement highly scalable, adaptable, the manageable network that will fulfill the business dynamic needs.

1.8 ONF

OpenFlow commenced with the publication of the original proposal in 2008. In 2011, a nonprofit consortium/ association called the Open Networking Foundation (ONF) was formed by a group of service providers to commercialize, standardize, and promote the use of OpenFlow in production networks. The ONF is a new type of Standards Development Organization that has a very active marketing department that is used to promote the OpenFlow protocol and other SDN-related efforts. The organization hosts an annual conference called the Open Networking Summit as part of these efforts.

One novel aspect of the ONF is that its board of directors is entirely made up of major network operators and not the networking vendors themselves. As of this writing, the ONF board is composed of chief technology officers (CTOs), technical directors, and fellows from companies such as Google, Facebook, Deutsche Telekom, Verizon, Microsoft, and NTT, among others. This helps prevent ONF from supporting

the interests of one major networking vendor over another. It also helps provide a real-world perspective on what should be investigated and standardized. The following Table 1 shows ONF working groups

Open Networking Foundation Working Groups	
Workgroup	**Description**
Extensibility	Ongoing creation of new OpenFlow versions to support more features and implementations.
Architecture and Framework	Defining the scope of SDN that the ONF will attempt to standardize.
Forwarding Abstractions	Looking at how the standard interacts with actual hardware implementation components.
Testing and Interoperability	Defining test and interoperability criteria, certification, etc.
Configuration and Management	Defining a protocol for configuring non-flow-related OpenFlow parameters (e.g., setup, controller, etc.)
Migration	Study migration of existing networks towards an SDN network based on OpenFlow.
Optical Transport	Responsible for identifying areas in which the OpenFlow Standard can be applied to optical networks.
Market Education	Responsible for educating the SDN community on the value of OpenFlow and for channeling market feedback within the ONF.
Wireless & Mobile	Study methods by which OpenFlow can be used to control wireless radio area networks (RANs) and core networks.
Northbound Interface	Developing concrete requirements, architecture, and working code for northbound interfaces.

Table 1 ONF working groups

1.9 SDN Interfaces

Following interfaces are the part of SDN structure.

1. Southbound Interface

Southbound Interface (SBI) refers to the Application programming Interface (API) exposed to lower layers. It allow controller plane to communicate with data plane. The defined protocols permit the controller to push policies to the forwarding plane. Majority of the SDN implementation uses OpenFlow and the network configuration southbound APIs.

2. East-West Interface

East-west interface allows communication between the multiple controllers. This interface is not yet standardized. It uses distributed routing protocol like BGP and OSPF.

3. Northbound Interface

The northbound interface (NBI) enables communication between the SDN Controller and the applications running over the network. NBI allows network operators to easily customize the network controls and perform this task through APIs. The well known Northbound API is REST API.

1.10 SDN History

The innovation in network management has enabled making of programmable computer networks. SDN history is divided into three stages:

1. **Active networks** (ranging from mid-1990s to early 2000s), which introduced programmable functions in the network to enable greater to innovation;
2. **Control and data plane separation** (ranging from 2001 to 2007), which developed open interfaces between the control and data planes; and
3. **OpenFlow API and network operating systems** (ranging from 2007 to around 2010), which represented the first instance of widespread adoption of an open interface and developed ways to make control-data plane separation scalable and practical.

Network virtualization has played an important role throughout the historical evolution of SDN.

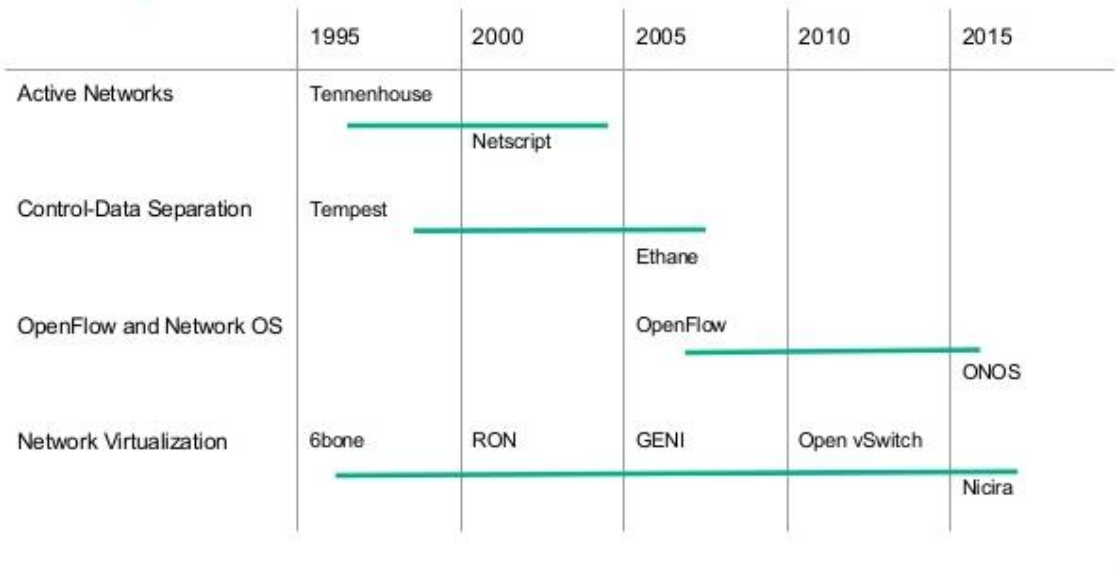

Figure 7: The origin of SDN

Active networks

As the internet traffic increased in the mid 1990s, network operators faced with numerous issues including route caching, preferential treating/filtering and traffic engineering. Networking gear at the time did not offer capabilities to effective manage network traffic. To provide some control over traffic passing through otherwise passive network, researchers undertook the work of creating network level API to bring some level of programmability to otherwise static network core. Tennenhouse & Wetherhall (1996) was the first to introduce “Active Network” idea, in which the user has injected customized programs to network nodes. These programs will be interpreted by network nodes to perform desired operation on the data flowing through the network. The idea of active network was radical at the time but did not gain much traction. However, the work later found validity in programs such as GENI (Global Environment for

Network Innovations), NSF's FIND (Future Internet Design) and EU's FIRE (Future Internet Research and Experimentation Initiative).

Separating Control and Data Planes

In the early 2000s, network operator introduced traffic engineering (a commonly known practice to control the path of traffic forwarding) to manage increased traffic, improve performance and reliability of the network. But the approach was primitive at best and increase frustration to improve the traffic engineering led to researchers looking for alternative to decouple control plane from data plane. Among several proposals, such as open control interface between control and data plane, "ForCES" (Forwarding and Control Element Separation), centralized control of the network, "RCP" (Routing Control Platform), "SoftRouter" and Ethane are important to note.

Ethane is a flow based policy enforcement mechanism that protects enterprise networks from threats (including namespace concerns, e.g. hostname bindings). It is a research project at the Stanford University that aim to create policy enforcement at enterprise networks through flow based network and managed by central domain controller that secures bindings and enforces policy, flow and access control.

OpenFlow and Network OS

These prior works on network programmability and separation of control and data plane gain further momentum during the mid 2000s as some of the chip vendors like Broadcom offered open API for programmers to control certain forwarding behavior (Feamster, Rexford & Zegura, 2014). The genesis of Openflow can be attributed to the tireless works of network operators, equipment vendors and networking researchers that created technology push and pull towards network programmability and disaggregation. It is more of an industry adoption when compared to intellectual predecessors (Feamster, Rexford & Zegura, 2014). Parallel to this development academia also felt the need to allow students to experiment new ideas in real network at scale. This aspiration guided a group of researchers at Stanford University to begun working on the initial concept of OpenFlow in 2008. By December, 2009 Openflow version 1.0 specification was released. Since its inception the development work and standardization efforts of Openflow is managed by **Open Networking Foundation (ONF)**, a user-led organization. The Openflow group at Stanford deployed testbeds to demonstrate capabilities of the protocol in single campus and over the WAN to multiple campuses. A real SDN use case is thus materialized in those campuses and by 2012 experimental use of Openflow deployment began in other realms including Data Center Networking.

1.11 SDN advantages and Disadvantages

The Major advantages of SDN are listed below-

1. **Intelligence and speed**

SDNs have the ability to optimize the distribution of the workload and make end devices more intelligent. This results in high-speed networks and gives more efficient use of the resources.

2. **Patterns in the networks**

In SDNs, the administrators have remote control over the network and can change the network characteristics, such as services and connectivity, based on the workload patterns. This enables administrators to have more efficient and instant access to the configuration modifications based on the network status.

3. **Multitenancy**

The concept of SDN can be expanded across multiple partitions of the networks, such as the data centers and the data clouds; this enables different players of the network to interact with each other in a more secure and efficient manner and provides the administrator with better tools for monitoring.

4. **Virtual application networks**

The network administrators mostly implemented virtual application networks using the Distributed Overlay Virtual Network (DOVE), which helps in the transparency, automation, and better mobility of the network loads that have been virtualized.

1.12 SDN Devices and Device Functions

An SDN device consists of an API for communication with the controller, an abstraction layer, and a packet-processing function. In the case of a virtual switch, this packet-processing function is packet processing software (Figure 7) but for physical switch the packet-processing function is embodied in the hardware for packet-processing logic as shown in figure 8.

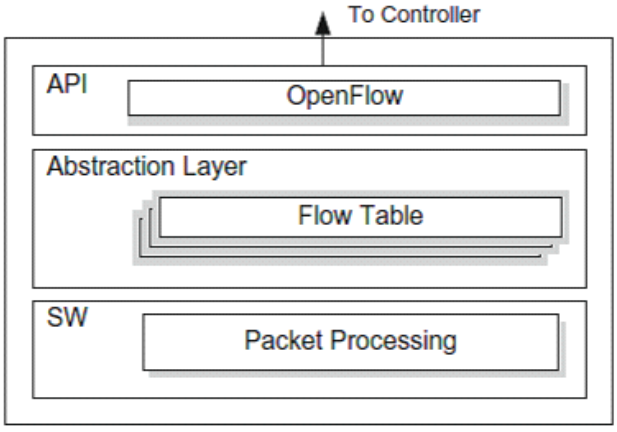

Figure 8 SDN Software Switch Structure

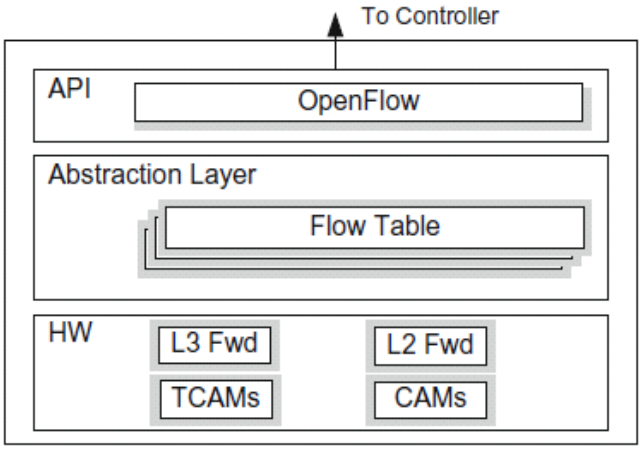

Figure 9 SDN Hardware Switch Structure

Abstraction Layer

The abstraction layer embodies one or more flow tables. Flow tables are the fundamental data structures in an SDN device. These flow tables allow the device to evaluate incoming packets and take the appropriate action based on the contents of the packet that has just been received. In network traditionally packets have been received by networking devices and evaluated based on certain fields. Depending on that evaluation, actions are taken. These actions may include forwarding the packet to a specific port, dropping the packet, or flooding the packet on all ports, among others. Switch flow table consist of number of prioritized flow entries, each of which commonly consists of two components like match field and actions.

Match fields are used to compare against incoming packets. An incoming packet is compared against the match fields in priority order, and the first complete match is selected. Actions specify for each flow entry are instructions that the network device should perform if an incoming packet matches the match fields specified for that flow entry. Match fields can have wildcards for fields that are not relevant to a particular match.

Packet Processing

The packet processing mechanism consists of logic to take actions based on the results of evaluating incoming packets and finding the highest-priority match. If the match is found for incoming packet it is processed locally until it is explicitly forwarded to the controller. If the match is not found for packet it may be copied to the controller for further processing. This process is also called controller consuming the packet. In case of the hardware switch these mechanisms are implemented by the specialized hardware. For the case of software switch, these same functions are mirrored by software. For the reason that case of the software switch is incredibly simpler than the hardware switch.

Software Switches

The simplest means of creating an SDN device is implementing SDN devices in software, because the flow tables, flow entries, and match fields involved are easily mapped to general software data structures, such as sorted arrays and hash tables. Implementations in software are likely to be slower and less efficient than those implemented in hardware, since they do not benefit from any hardware acceleration. The packet processing function uses sophisticated software logic to implement efficient match field lookups. The resource constraints are less in implementing software SDN device, since considerations such as processing electricity and memory size are not an issue in typical implementations, whereas, a hardware SDN tool implementation supports only a comparatively limited number of flow entries.

As software device implementations have more flexibility to implement more complex actions. The software-based network devices, such as the hypervisors of a virtualization system are set up with mostly Software SDN device implementations. As well as such hypervisors often incorporate a software switch implementation that connects the various virtual machines to the virtual network. The virtual switch working with a hypervisor is a natural fit for SDN. The centralized management system is controlled the whole virtualization system.

SDN Hardware Switches

Hardware implemented SDN devices keeps the promise of operating faster than software implemented SDN devices. Thus, they are more applicable to performance in sensitive environment, such as in data centers and network cores. Presently, the network devices use specialized hardware which is designed to inspect incoming packet and make subsequent decisions based on packet matching operation. It can be seen in figure 7, that packet processing logic (software based) in figure 8 is been replaced by specialized hardware. This hardware consists of layer two and layer three forwarding tables, typically implemented using content-addressable memories (CAMs) and ternary content material-addressable reminiscences (TCAMs). The layer three forwarding table is used for making IP-level routing decisions which matches the destination IP address against entries in the table. This is fundamental operation of a router based on the match the layer 3 forwarding table takes appropriate routing action. The layer two forwarding table is used for making MAC-level forwarding decisions which matches destination MAC address against entries in the table. This is the fundamental operation of a switch based on the match the layer 2 forwarding table takes appropriate forwarding action.

1.13 SDN Based Network Applications

With new protocol, such as OpenFlow which is becoming more standard in the industry, software-defined network (SDN) is becoming easier to implement. SDN decouples control plane from the data plane, thus allowing switches to compartmentalize their two main functions. The control plane generates the routing table and determines where the packets should be sent. The separation allows the network to be abstracted further, which simplifies networking analysis. Many companies use OpenFlow protocols within

their data center networks to simplify operations. OpenFlow and SDN allow data centers and researches to innovate their network with new ways because it is easier to abstract network.

1. Internet Research

The internet was originally built for research. As such, during its origin, its architecture never conceived of the vast network that it has now become. As technology evolves, internet is not able to meet the new requirements. Thus, updating the Internet brings many challenges because it is constantly being used. It is difficult to test new ideas and strategies to solve the problems found in an existing network. SDN technologies allows to test ideas for a future Internet without changing the current network.

SDN permits new networking protocols to be defined and readily experimented in real conditions over production networks. SND allows separation of control and data traffic with an OpenFlow switch that make easy to separate hardware from the software. Experimentation with new addressing schemes to test the new internet architecture schemes is possible due to this separation of control and data traffic. Generally, it is difficult to experiment with new types of networks because these networks use different addressing schemes and include other nonstandard aspects, it is difficult to integrate these changes into existing network.

The separation of the control and data plane is possible with routers, switches, and access points from many different companies using OpenFlow protocol. The device send data packets based on defined rules from the controller. If the device do not have any rule for data packet, the device forwards packet to the controller for review. The controller decides on packet, and, if necessary, sends a new rule to the device so that it can handle future data packets in the same manner.

Currently, there are several projects in Europe and Brazil that use OpenFlow. In Europe, eight islands are currently interconnected using OpenFlow, whereas in Brazil, there are plans to create a network that will work with that in Europe. Several other projects are currently being developed, including MobilityFirst, eXpressive Internet Architecture etc. Each of these projects is important in creating a better Internet for current and future needs.

2. Ruler Connectivity

Apart from internet research, SDN also has application in making the Internet more widespread than it already is. If a person lives away from vastly populated area, it is almost impossible to use the internet. Rural areas are often forgotten or ignored by the large companies because the profit margins are small, and it is difficult to update the required hardware.

SDN simplifies complex data center and enterprise network; it can further be used to simplify rural Wi-Fi network. The main issues with rural environments include sparse population, small profit margins and resources constraints, and others. Recent innovation in technologies like long-distance Wi-Fi, larger unlicensed frequency band and SDN help alleviate these issues.

SDN is useful because it separates network construction and network configuration by placing the control and management functionality into the central controller which allows companies to decrease startup

costs in rural environments and gain more profit. As rural networks become more profitable, more companies will be willing to give access to more and more rural areas.

Also SDN give an advantage that all of the rural networks would then be viewing the network from a global perspective, and the companies could then share software solutions with others, rather than rely on ad-hoc methods that are only applicable to specific situation or environment. SDN gives opportunities to rural networking companies to work with established companies rather than compete.

However, there are several difficulties involved in setting up the rural network solution. These difficulties include a network such as SDN that has never been set up in practice anywhere and it is more difficult to set up controls in rural environments than in data centers, where it is easier to separate control traffic from data traffic. A true test environment in a rural area would need to be conducted to find the feasible solution.

3. Updating Data Center Networks

Currently the main problem faced with network is to connect different data center networks. Often, data center networks use proprietary architectures and topologies, which create issues when merging different networks; however, there is often a need to merge two divergent networks. For example, when a new company is acquired, the original network must be expanded to incorporate the new network, but with the aforementioned problems, this merge is often time consuming. SDN brings a solution to this networking issue. OpenFlow-based network infrastructure service software can be used to connect data center networks.

The problem of latency of these interconnected data center networks could solved by moving the workload to underused networks. If a network is busy at a certain time of the day, the workload might be completed sooner in a network of a different time zone or in a network that is more energy efficient.

4. Cloud Data Scalability

The data centers are an integral part of many companies. For example, Google has a large number of data centers, so they can quickly and accurately provide data when requested. Similarly, many other companies use data centers to provide data to clients in a quick and efficient manner, but data centers are expensive to maintain. OpenFlow allows switches to be managed from a central location which help companies to save money on setting up and configuring networks.

OpenFlow allows testing of new routing techniques and changes in protocols without affecting anything in a production environment. This feature of OpenFlow allows researchers to invent new networking protocols that can work more efficiently for many applications.

“Baker C., A. Anjum, R. Hill, N. Bessis and S.L. Kiani have conducted research on scalability of cloud data center using OpenFlow protocol. In their research, they have created a data center model with a large number of nodes to test performance, throughput, and bandwidth. Such a model consists of 192 nodes with four regular switches and two core switches with an OpenFlow controller. A firewall was

present between the core switches, the OpenFlow controller, and the router. Mininet was also used to prototype their network and test the performance. Mininet is an application that allows researchers to customize an SDN using OpenFlow protocols. Additionally, they have used several other tools like Iperf, Ping, Ping All, Ping Pair and CBench to analyze their network set up. After testing their network using all the tools, they found that OpenFlow would be good to use in data centers because of its ability to reduce switch configuration time as well as to be controlled from a centralized controller. They recommend that data centers wait on OpenFlow to be tested further before changing all their switches to OpenFlow switches. Because their testing was done on virtual machine using Mininet, it would further need to be tested in real environment to ensure that all results are realistic.

5. Mobile Application offloading

Mobile devices have become increasingly popular both for general consumers and for businesses in recent years. To make mobile devices truly useful in a business environment, they need to securely send data to other servers and other work machines. Any additional security software needs to be lightweight so as not to impede the functioning of the device as battery life and performance are most critical on mobile platforms.

Two important considerations to make when creating offloading software for mobile devices are privacy and resource sharing. In business applications privacy is important because people often work on data that need to be kept secure. Some data can be sent among only a few people, whereas other data do not require the same level of security. For mobile offloading applications the ability to determine which data require additional security is important. In addition, resource sharing is important for mobile offloading because it allows mobile devices to exploit the machines that are idle or not fully using their capabilities. Enterprise-centric offloading system (ECOS) is used to address concerns of security and resource sharing. The controller of the system needed to do most of the computational work to relieve mobile devices of this necessity. Applications with additional security requirements are only offloaded on approved machines as ECOS was designed to offload data to the idle computers. Performance was also considered for different users and applications. ECOS reviews the available idle computing resources to find an appropriate resource to offload data. The machine chosen is necessary to provide a benefit in the form of latency or energy savings or both. Furthermore, the machine chosen has to meet the minimum security requirements for the particular application. Three security categories for the applications are user-private data (accessibility by certain users), enterprise-private data (accessibility within the business environment), and no-private data (accessibility by any one, no security required). SDN is used to control the network and to select resources. It is a good choice because it is less resource intensive than the other choices.

The selected resources must be able to meet the security requirements listed above. The controller will decide if such type of device is available for offloading that meets the security requirements while maintaining energy savings. If such device is not there, data is not allowed to be offloaded from the

mobile device. If energy savings are not necessary, then any available resource with enough capacity is used. OpenFlow switches are used so that the controller can regulate the flows.

6. Mobile Virtual machines

In business, applications running on virtual machine are becoming increasingly common. The application on virtual machines allows companies to keep lower operation cost and make it very flexible. To extract the full potential from a virtual machine, there need to be a means of making them more portable. The main problem associated while making virtual machines portable is to maintain the IP address of the virtual machine in the process. To maintain the same IP address, the virtual machine would have to remain in the same subnet or a new scheme would need to be created.

It was found that the current methods of handing virtual machines were not efficient. The solutions found by “Mann,V., A. Vishnoi, K. Kannan, and S. Kalyanaraman” in their studies includes use of mobile IP or a dynamic DNS. The main issue with both these solutions is that someone has to manually configure the network setting after removing the virtual machine. This restricts data centers and business from easily porting their virtual machines to new locations. The application like CrossRoads was developed by Mann et al. to solve virtual machines mobility issue. It was designed to allow the mobility of both live and offline virtual machines. It has three main purposes.

- To take care of traffic from data centers as well as external users.
- To make use of OpenFlow with the assumption that each center an OpenFlow controller
- To make use of pseudo address for IP and MAC address so that they remain constant when porting while allowing the real IP to change according.

The software’s basic implementation was to create rules for finding the virtual machines in different networks. The CrossRoads controller would keep track of the real IP and MAC address for the controller in each data center as well as the virtual machines in its own network. A request is broadcasted to the controllers when it is intended for application running on a particular virtual machine. If the controller receives a request for a virtual machine that is not in table, it broadcast the request to other controllers. The controller who has the real IP address of virtual machine then sends the pseudo MAC address to the original controller, and the original controller can update its table in case it gets another request in the near future. Mann et al. set up the virtual testing environment to understand CrossRoads performance in a realistic environment. The address resolution protocol (ARP) latency was decreased by 30% while maintaining similar ping times, available bandwidth, and HTTP response time with respect to default testbed.

1.14 SDN Challenges

1. Latency

A centralized controller means certain decisions will suffer non trivial round-trip latency as the networking element requests the policy direction from controller. The way this delay impacts the operation of the network, and to what extent, remains to be determined. Moreover, with the central controller providing policy advice for a number of network devices, it is unknown whether the conventional servers on which the controller runs will be able to service these requests at sufficient speed to have minimal or no impact on network operation.

2. Scale

The SDN network having centralized controller must handle network topology organization, optimal path determination and responses to changes. This is the appropriate location for this functionality, however, with increase of network devices in network the scale and ability of a single controller to handle these devices is questionable. It is difficult to know how well a centralized system can handle hundreds, thousands, or tens of thousands of network devices and to know what the solution is when the number of network devices outgrow the capacity the controller to handle them. If we try to scale by adding controllers, how do they communicate, and who orchestrates coordination among the controllers?

3. High availability (HA)

The centralized controller of network must not be the single point of failure. That means there is a need for redundancy schemes.

1) Availability of redundant controller so that processing power is available in the event of failure of a single controller.
2) The actual data used by the set of controllers needs to be mirrored such that the controllers can program the network devices in a consistent fashion.
3) The communications paths to the various controllers need to be redundant to ensure that there is always a functioning communications path between a switch and at least one controller

4. Security

With centralized controller security attacks are able to focus on that one point of failure. There is a possibility that centralized controller system is more vulnerable to attack than distributed network. Thus it

is important protect both centralized controller and the communication channels between controller and network devices.

Traditional Switches

The core of a traditional network switch is based on custom silicon, either an ASIC, FPGA or NPU. Even though FPGAs and NPU's can be modified in the field, they are still limited in functionality and relatively expensive. These hardware devices can forward packets based on Layer 2 and Layer 3 input at the requisite "wire speed." The trade off worth noting is performance over flexibility. The control plane handles the routing functions and calculates the packet forwarding rules.

The figure below illustrates a traditional non-SDN switch, with the following components:

• **The transceivers (TRX)**, which are the ports transmitting or receiving communications on the medium (copper, optical, radio frequency, etc.).

• **An Application Specific Integrated Circuit (ASIC)** is used to handle the incoming and outgoing data packets. An ASIC is a specialized purpose built silicon device that executes just a limited set of tasks, and, in this particular instance, do it very fast (up to 40Gbit/sec per port).

• **The Layer 2 and Layer 3 tables** are the central building blocks the ASIC acts on.

To fulfill the demands of virtualized servers, Virtual Local Area Networks (VLANs) or Virtual Routing Functions (VRFs), new features have been added for better isolation of the hosts or virtualized machines: Access Control, Quality of Service (QoS), port groups, etc. Most of these features must be configured manually and in different ways for various vendors.

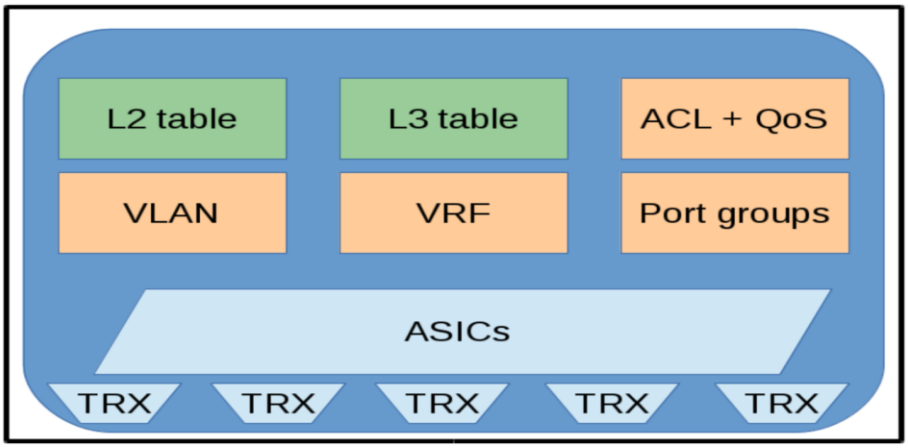

Figure 10 : A traditional non-SDN switch.

Traditional Switches and the Three Planes

The next figure illustrates what a switch looks like when combining the components of a non-SDN switch with the data, control, and management planes:

• The transceivers (TRX) and the ASIC form the data plane.

• A general purpose CPU hosts both the control and management planes.

• The control plane handles the routing functions and is also responsible for calculating the forwarding rules.

• The management plane is used to set up and change the network switch configuration.

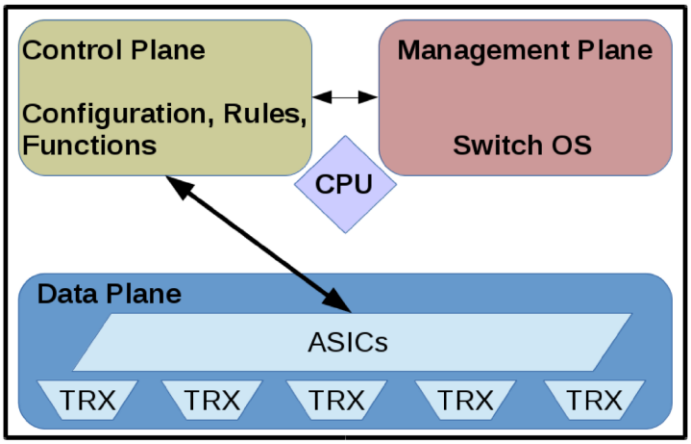

Figure 11 : Data, Control, and Management Planes in a Switch.

Text Books

1. Thomas D. Nadeau, Ken Gray, SDN: Software Defined Networks, An Authoritative Review of Network Programmability Technologies, O'Reilly Media,ISBN:10:1-4493-4230-2, 978-1-4493-4230-2
2. Paul Goransson and Chuck Black, Software Defined Networks: A Comprehensive Approach, Morgan Kaufmann, ISBN:9780124166752, 9780124166844.

Reference Books

1. Vivek Tiwari, SDN and OpenFlow for Beginners ,Digital Services,10: 1-940686-00-8 13: 978-1-940686-00-4
2. Fei Hu, Network Innovation through OpenFlow and SDN: Principles and Design,CRC Press,ISBN:10: 1466572094
3. Open Networking Foundation (ONF)Documents, https://www.opennetworking.org
4. OpenFlow standards, http://www.openflow.or
5. Online Reading, http://www.nec-labs.com/~lume/sdn-reading-list.html,
6.https://www.linux.com/topic/networking/software-defined-networking-fundamentals-part-2-switches-and-network-architecture/