TELCOMA
AN ISO CERTIFIED COMPANY

SDN & NFV

Full course at https://telcomaglobal.com

Module 1 : understanding SDN & NFV

SDN

Introduction :

- It is a new approach to designing , building and managing networks.
- SDN allows network administrators to quickly and easily manage network services from a centralize location without having to manually configure each individual network element (switch/router).

Introduction :

- The basic concept of SDN is to make network control decisions from centralized location.
- It is done by decoupling and optimizing the system that makes switching/routing decisions and other control functions such as signalling from the underlying systems that forwards traffic to the selected destinations.

Introduction :

- SDN requires some mechanisms for the centralized controller (such as opendaylight or open network operating system (ONOS) to communicate with the distributed data plane.
- This interface is known as southbound interface.
- Southbound interface is open flow.

Introduction :

- The open networking foundation was founded to promote SDN and openflow by specifying the openflow application programming interfaces (API) for connecting between software defined controllers and switches/routers.

SDN planes :

- Control plane : it is where the administration of network takes place.
- Data plane : it encompases the application of rules which are defined on control planes, this is the place where actual packet processing takes place.

Generic principle :

- The data plane must be very fast to handle a very high number of packets with simple rules so as to obtain a good bit rate in the network.
- For legacy switches, the whole control and data planes would be tightly linked into the same hardware box.
- Virtual switches make it possible to centralize the management of the switches and to remotely program them.

SDN Architecture

Architecture :

- SDN architecture defines how a networking and computing system can be built using a combination of open, software based technologies and commodity networking hardware that separate the SDN control plane and SDN data plane of the networking stalk.
- Control and data plane elements can communicate using OpenFlow protocol.

SDN stalks :

- In SDN architecture , the spitting of the control and data forwarding functions is referred to as “disaggregation “ .
- This architecture gives the applications more information about the state of the entire network from the controller.

SDN architecture components :

- SDN applications
- SDN controllers
- SDN networking devices

SDN overview

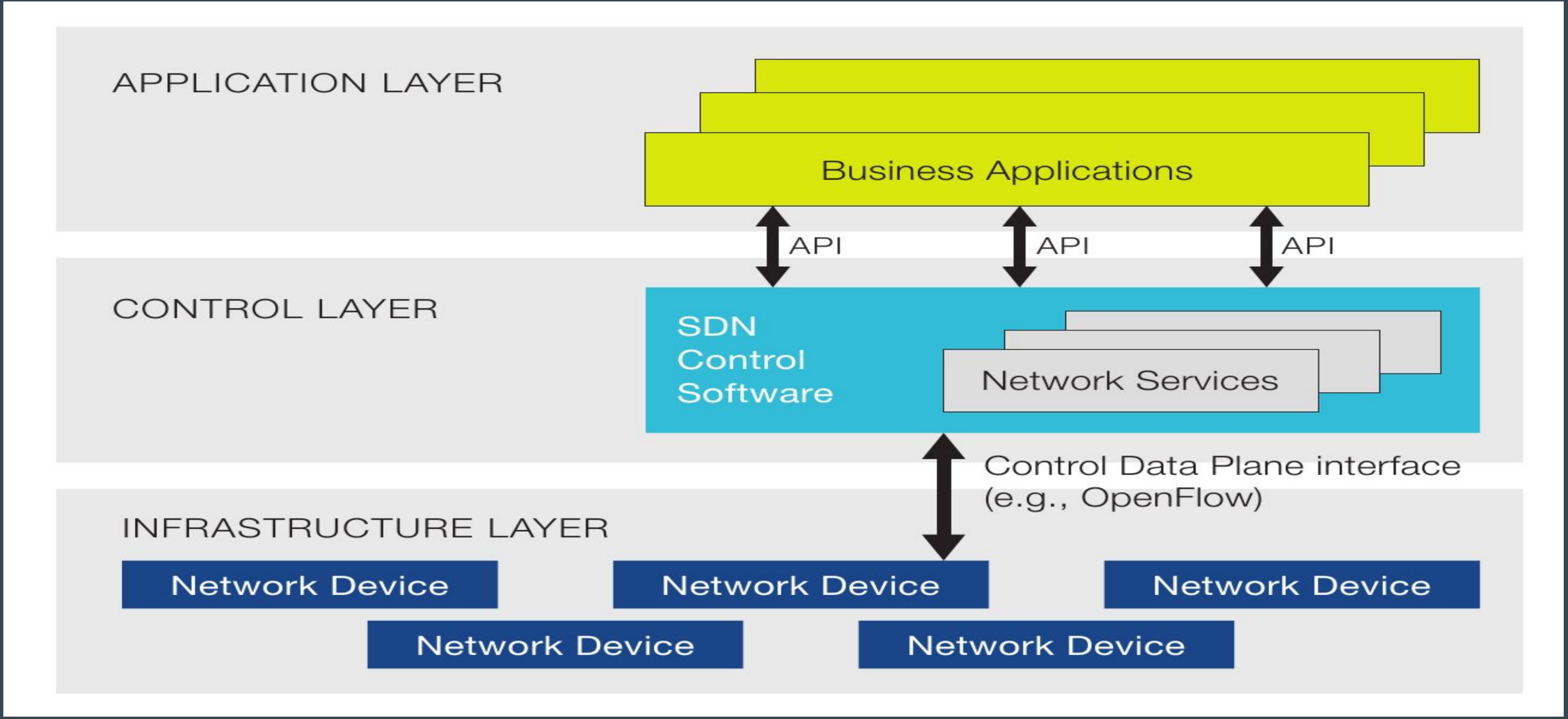
SDN overview :

- The aim of SDN is to provide open interfaces.
- It enable the development of software that can control the connectivity provided by a set of network resources and the flow of network traffic through them.
- The initial view comprised of infrastructure , control and application layers

SDN overview :

- The infrastructure layer comprises network elements, which expose their capabilities toward the control layer via interfaces south bound from the controller.
- The SDN applications exist in the application layer and communicate their network requirements towards the controller plane via northbound interfaces.
- In the middle, the SDN controller translates the applications, requirements and exerts low level control over network elements.

SDN overview :



SDN overview :

- With the addition of management plane ,
- In the data plane, management is required for initially setting up the network elements , assigning the SDN controlled parts and configuring their SDN controller.
- In the controller plane, management needs to configure the policies defining the scope of control given to the SDN application and to monitor the performance of the system.

SDN overview :

- In the management plane, management typically configures the contracts and service level agreements.
- In all planes, management configures the security associations that allow distributed functions to safety intercommunicate.

SDN Overview :

- In the SDN controller, different agents may expose control over the network at different levels of abstraction or function sets.

SDN planes

Data plane :

- It comprises of a set of one or more network elements , each of which contains a set of traffic forwarding or traffic processing resources.
- Resources are always abstractions of underlying physical capabilities or entities.

Controller plane :

- It comprises a set of SDN controllers , each of which has exclusive control over a set of resources exposed by one or more network elements in the data plane.
- The functionality of the SDN controller is to faithfully execute the requests of the applications it supports ,while isolating each application from all others.

Application plane :

- It comprises of one or more applications, each of which has exclusive control of a set of resources exposed by one or more SDN controllers.
- An application may invoke or collaborate with other apps.

Management :

- Each application, SDN controller and network element has a functional interface with management.
- The minimum functionality of the manager is to allocate resources from a resource pool in the lower plane to a particular client entity in the higher plane.

Principles and Architectural components

Principles :

- Decoupling of controller and data planes
- Logically centralized control
- Exposure of abstract network resources and state to external applications

Decoupling of controller & data planes :

- This principle calls for separable controller and data planes.
- It is understood that control must necessarily be exercised within data plane systems.

Logically centralized control :

- In comparison to local control, a centralized controller has a broader perspective of the resources under its control , and can potentially make better decisions about how to deploy them.

Exposure of abstract network resources :

- Applications may exist at any level of abstraction or granularity , attributes often described as different latitudes, with the idea that further suggests a degree of abstraction.

Principle :

- The principle of abstracting network resources and state to applications via A-CPI allows for programmability of the network .
- The concept of hierarchically recursive application/ controller layers and trust domains also allows application programs to be created that combine a number of component applications into a more comprehensive service.

Principle :

- This architecture incorporates the concept of trust domain boundaries, which is vital to widespread commercialization.
- It defines components entirely within particular trust domains, with well defined reference points to other trust domains.

Purpose of hierarchical levels :

- Scaling and modularity
- security

Data plane

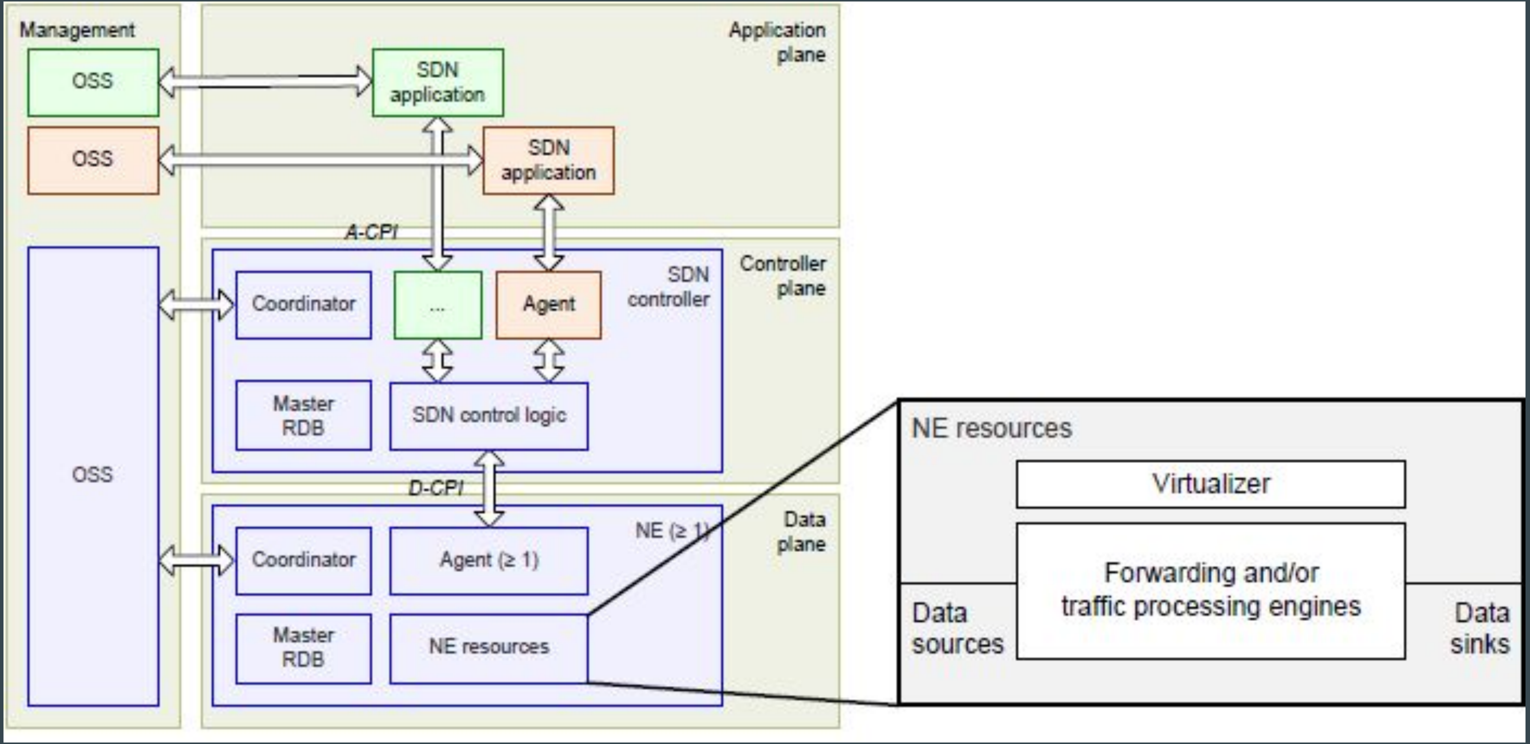
Data plane :

- It incorporates the resources that deal directly with the customer traffic.
- The NE resource block comprises data sources, data sinks and forwarding and /or traffic processing engines as well as a virtualizer whose function is to abstract the resources to the SDN controller and enforce policy.

Data plane :

- This expansion of detail also introduces a master RDB , the conceptual repository of all resource information known to the network element.

Data plane :


Management
OSS
OSS
OSS
Application plane
SDN application
SDN application
A-CPI
Controller plane
SDN controller
Coordinator
...
Agent
Master RDB
SDN control logic
D-CPI
Data plane
NE (≥ 1)
Coordinator
Agent (≥ 1)
Master RDB
NE resources
NE resources
Virtualizer
Forwarding and/or traffic processing engines
Data sources
Data sinks

Data plane :

- The data plane implements forwarding decisions made in the controller plane.
- In principle, it does not make autonomous forwarding decisions.
- The data plane coordinator is the entity by which management allocates data plane resources to various agents.

Controller plane

Controller plane :

- The SDN controller plane is modelled as the home of one or more SDN controllers.
- SDN control is logically centralized.
  A controller typically has subnetwork scope, spanning more than a single physical NE.
  There is no resource contention with other entities, the SDN controller regards itself as the owner of the virtual resources allocated to it by management.

Controller plane :

- An SDN controller is expected to coordinated a number of interrelated resources, often distributed across a number of subordinate platforms, and sometimes to assure transactional integrity as part of the process. This is called orchestration.

Controller plane :

- SDN Controller is expected to coordinate a number of interrelated resources, often distributed across a number of subordinate platforms and sometimes to assure transactional integrity as part of the process. This is called orchestration.

SDN Controller

SDN controller :

- It could be a confederation of identical processes arranged to share load or protect one another from failures.
- Controller components are free to execute on arbitrary compute platforms, including compute resources local to a physical NE.

SDN controller :

- SDN controller is understood to have global scope , for some value of globe, and that its components are understood to share information and state, such that no external block need concern itself with conflicting or contradicting commands from the controller.

SDN controller :

- Multiple manager or controller components may have joint write access to network resources, but to comply with SDN principles , they must either :
  Be configured to control disjoint set of resources or actions

Functional components :

- Data plane control function
- Coordinator
- Virtualizer
- Agent

Data plane control function :

- This component effectively owns the subordinate resources available to it and uses them as instructed by the OSS/ coordinator or virtualizer (s) that controls them.
- Scope of an SDN is expected to span multiple (virtual) NE's or even multiple virtual networks , the DPCF must include a function that operates on the aggregate. This function is commonly called orchestration.

Coordinator :

- The coordinator is the functional component of the SDN controller that acts on behalf of the manager.
- Clients and servers require management, throughout all perspective on data, control and application plane models, so coordination functional blocks on ubiquitous.

virtualizer :

- It differs from the virtualization concept as used in SDN architecture.
- In the SDN architecture, virtualization is the allocation of abstract resources to particular clients or applications.

SDN controller :

- An SDN controller offers services to applications by way of an information model instance that is derived from the underlying resources, management installed policy and local or externally available support functions.
- The functional entity that supports the information model instance and policy at an A-CPI is called virtualizer.

SDN controller :

- Virtualizer and DPCF and possibly other SDN controller functions must collaborate to provide features such as notification interpretation, resource sharing, implicit provider services and transactional integrity.

Agent :

- The controlled entity is designated the agent, a functional component that represents the client's resources and capabilities in the server's environment.
- An agent in a given SDN controller at level N represents the resources and actions available to a client or application of the controller at level N+1.

Other controller components :

- These may take the form of applications or features supported by the controller.
- As components of the SDN controller, such applications or features are subject to the same synchronization expectation as other controller components.
- To facilitate integration with third party software, the interfaces to such applications or features may be same as that of A-CPI.

Shared resources :

- An extension of shared resource feature is the possibility that the provider could maintain a pool of uncommitted resources for an on demand allocation.
- Maximizing resource usage implies that resources may be oversubscribed.

Application plane

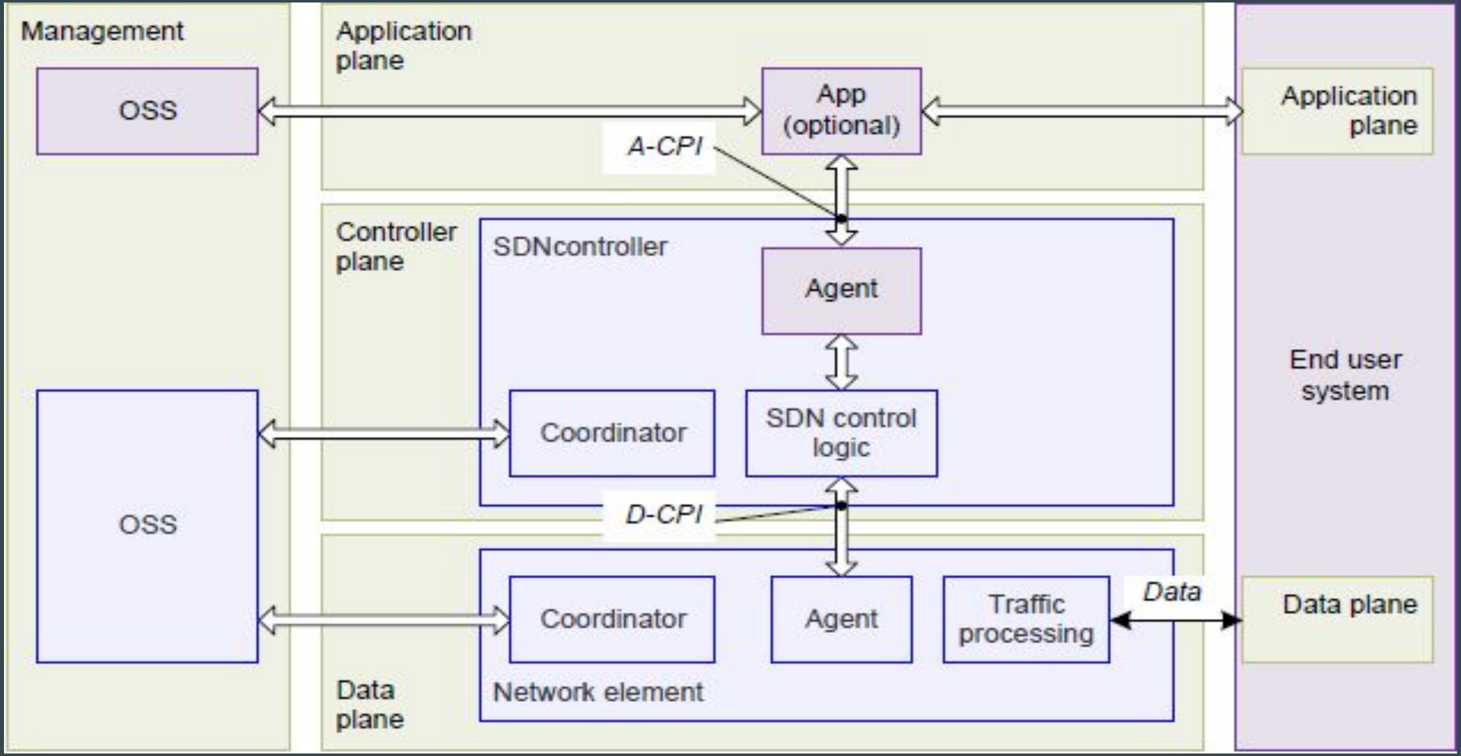
Application plane :

- SDN principles permits applications to specify the resources and behavior they require from the network, with the context of business and policy agreement.
- The interface from the SDN controller to the application plane is called the Application controller plane interface.

Application plane :

- An application plane entity may act as an information model server.
- An application plan entity may act as an information model client.
- An application plane entity may act in both roles simultaneously.

Multi plane end user system :



Virtualization :

- SDN control and management must be designed to operate on abstracted and virtualized resources, which ultimately emcompases underlying physical resources, possibly through several successive levels of virtualization.
- Data plane of an SDN is a network, a set of nodes that forwards traffic and may also produce, consume , store or process traffic.

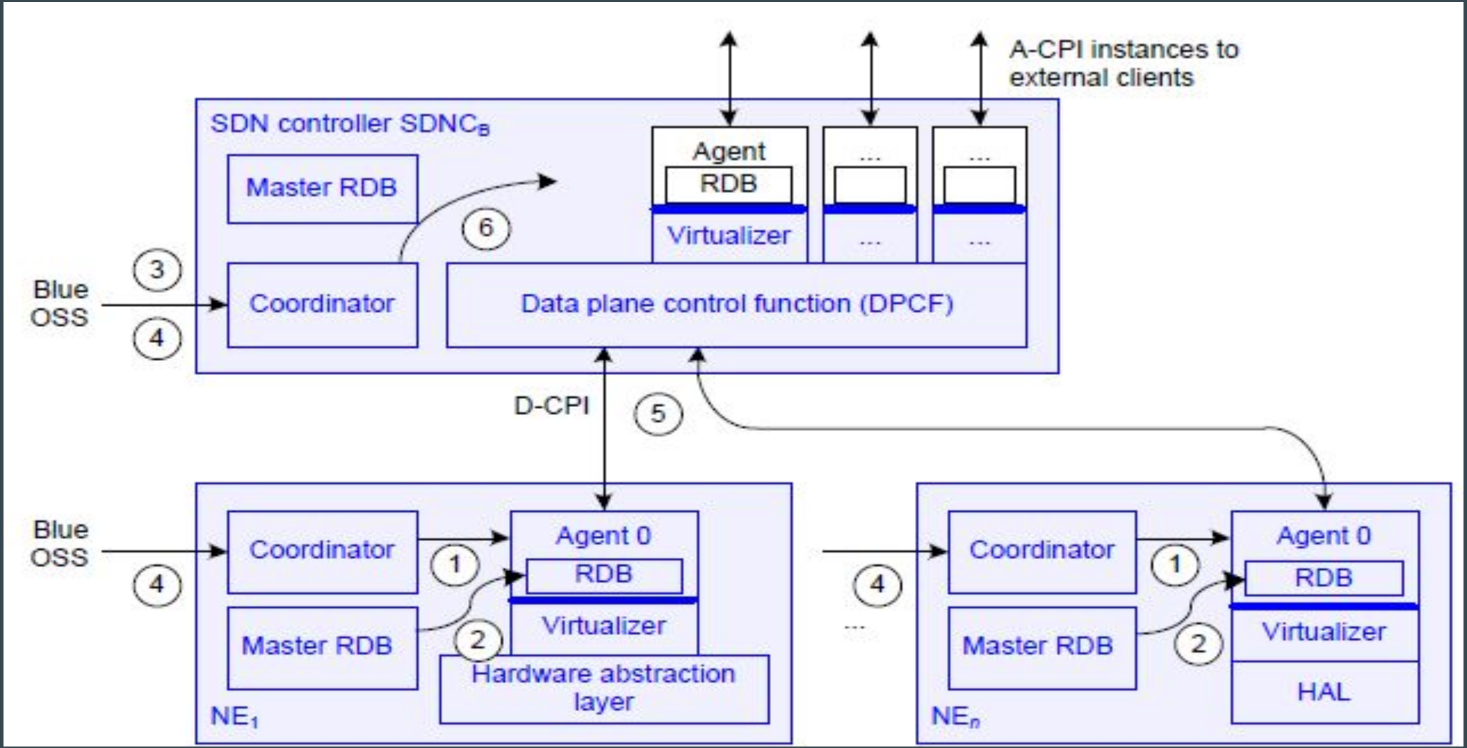
Control functions & interactions :

- Control is at the heart of SDN.
- Four scenarios are :
  Single player SDN provider
  SDN provider with SDN clients, with underlying network exposed.
  SDN provider with virtualized network , non-recursive.
  SDN provider with recursive virtualized network.

Single player SDN provider :

- All networking is based on a set of physical NE’s.
- Set of NE’s that forms one or more sub networks within the control domain of a single SDN controller.

SDN control of physical switches :



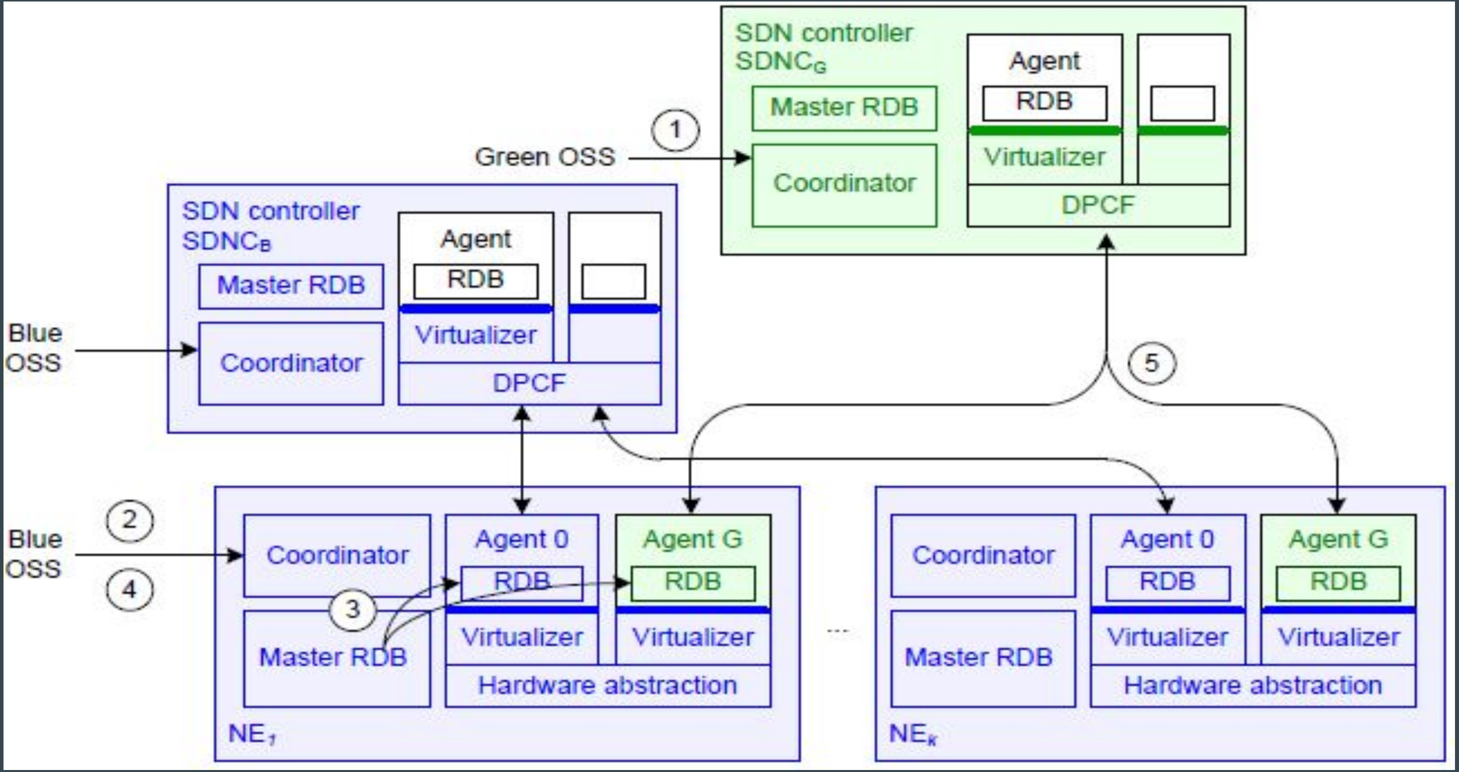
Single player SDN provider :

- Each of n NE’s contain a coordinator function.
- Each of n NE’s contain its own master RDB, which models all the resources in a NE.

SDN provider with SDN clients :

- The virtual network is exposed to include only selected ports and their supporting resources, but it rests directly on blue’s physical NE’s.

SDN provider with SDN clients :



SDN provider with virtualized network :

- N

Fundamental characteristics of SDN

Characteristics :

- Plane separation
- A simplified device
- Centralized control
- Network automation and virtualization
- openness

Plane separation :

- Separation of the forwarding and control planes.
- Characteristics such as MAC address, IP address and VLAN ID resides in the forwarding plane.
- It may forward, drop, consume or replicate an incoming packet.
- Protocols, logic and algorithms that are used to program the forwarding plane reside in the control plane.

A simple device and centralized control :

- Simplification of devices which are then controlled by a centralized system running management and control software.
- This software based controller manages the network using higher level policies.

Network automation & virtualization :

- Basic abstractions are : distributed state, forwarding and configuration.
- The distributed state abstraction provides the network programmer with 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.

Network automation & virtualization :

- The forwarding abstraction allows the programmer to specify the necessary forwarding behaviors without any knowledge of vendor specific hardware.

Openness :

- The characteristics of open SDN is that its interfaces should remain standard, well documented.
- The API's that are defined should give software sufficient control to experiment with and control various control plane options.

Openness :

- The speed at which network technology is developed and deployed is greatly increased as much larger number of individuals and org are able to apply themselves to today’s network problems, resulting in better and faster technological advancement in the structure and functioning of networks.

SDN operation

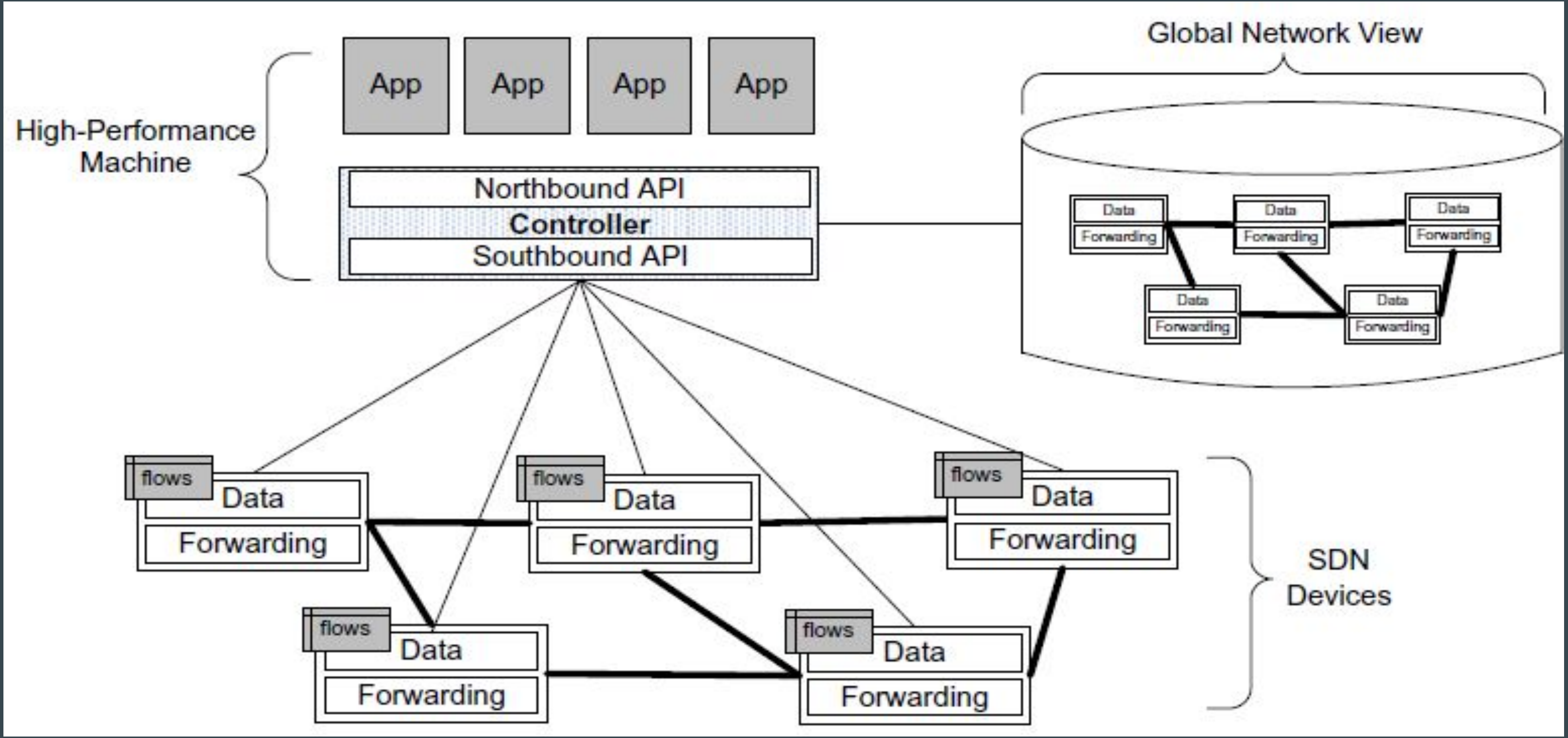
SDN operation :

- The SDN devices contain forwarding functionality for deciding what to do with each incoming packet.
- The devices also contain the data that derives those forwarding decisions.
- The data itself is actually represented by the flows defined by the controller.

SDN operation :

- A flow describes a set of packets transferred from one network endpoint to another network endpoint.
- A flow is unidirectional in that packets flowing between the same two endpoints in the opposite direction could each constitute a separate flow.

SDN operation view :



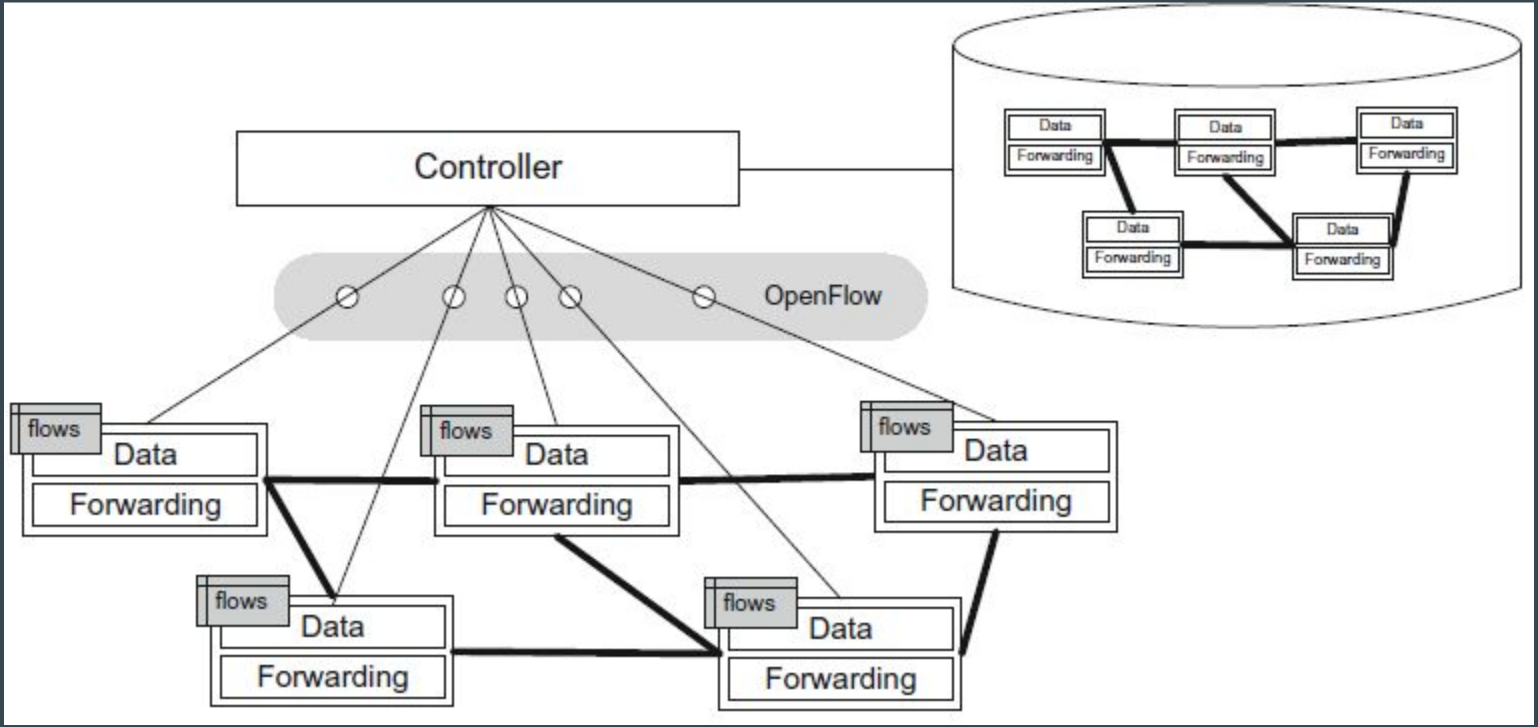
SDN operation :

- When the SDN device receives a packet, it consults its fow tables in search of a match.
- If the SDN device finds a match, it takes the appropriate configured action, which usually entails forwarding the packet.

SDN operation :

- SDN applications are built on top of the controller.
- The SDN application interfaces with the controller, using it to set proactive flows on the devices and to receive packets that have been forwarded to the controller.
- There are also reactive flows that are defined or modified as a result of stimuli from sources other than packets from the controller.

Controller to device communication :



SDN Devices

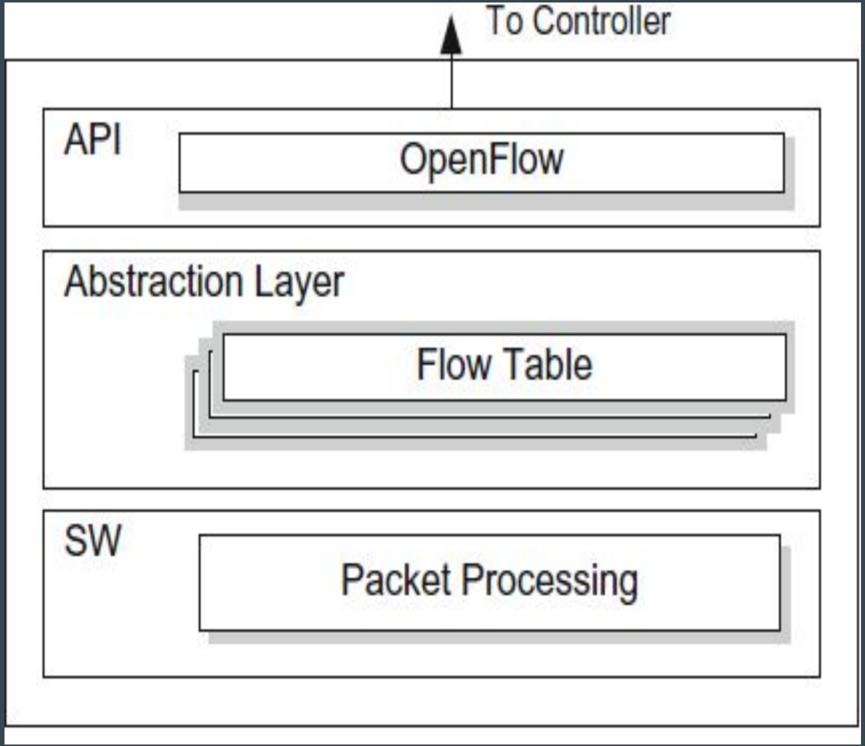
SDN devices :

- An SDN device is composed 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.
- The abstraction layer embodies one or more flow tables.

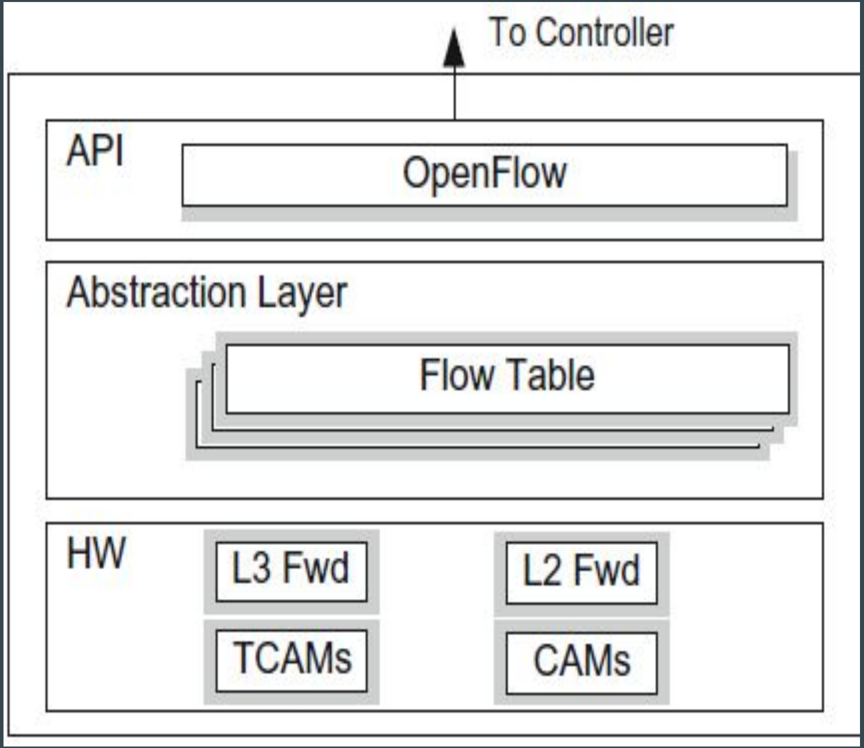
SDN devices :

- The packet processing logic consists of mechanisms to take actions based on the results of evaluating incoming packets and finding the highest priority match.
- In case of hardware switch, these mechanisms are implemented by specialized hardware.
- In case of software switch, these same functions are mirrored by software.

SDN software switch anatomy :



SDN hardware switch anatomy :



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.

Flow tables :

- Flow tables consists of number of prioritized flow entries, each of which typically consists of two components : match fields 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.

SDN Software switches

SDN Software switches :

- Implementation of SDN devices in software is the simplest means of creating an SDN device, because of flow tables, flow entries and match fields mapped to general software data structures , such as sorted arrays and hash tables.
- For network devices that must run at high speeds, such as 10 gbps , 40 gbps, and 100 gbps only hardware implementations are feasible.

Hardware SDN devices :

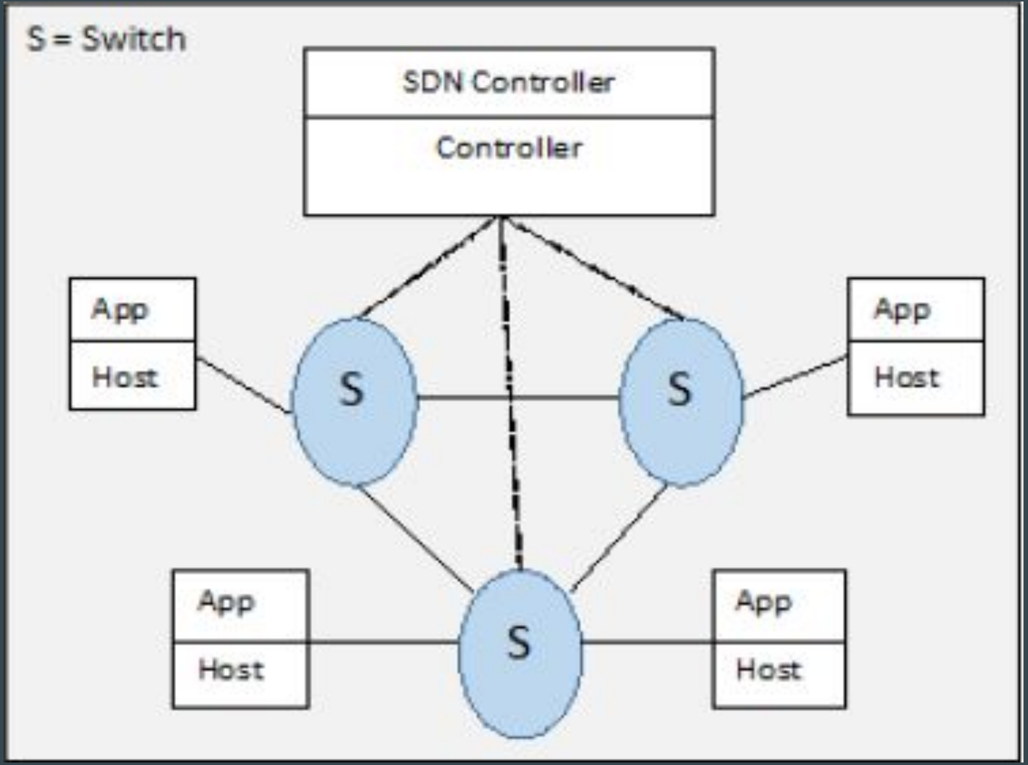
- Hardware implementations of SDN devices hold the promise of operating much faster than their software counterparts and thus more applicable to performance sensitive environments such as data centres and network cores.
- The hardware includes the layer-2 and layer-3 forwarding tables.

Mininet

Mininet Introduction :

- It is an approach of using operating system virtualization features , and and processes to allow it to scale up to hundreds of nodes.
- Users can implement a new network feature or a new architecture, test it on large topologies with application traffic and then deploy the same code and test scripts into real production network.

Emulating hardware n/w using Mininet :



Mininet :

- It can simulate SDN networks , can run controller for experiments.
- Mininet by default includes OVCS controller and open VSwitch.

Commands :

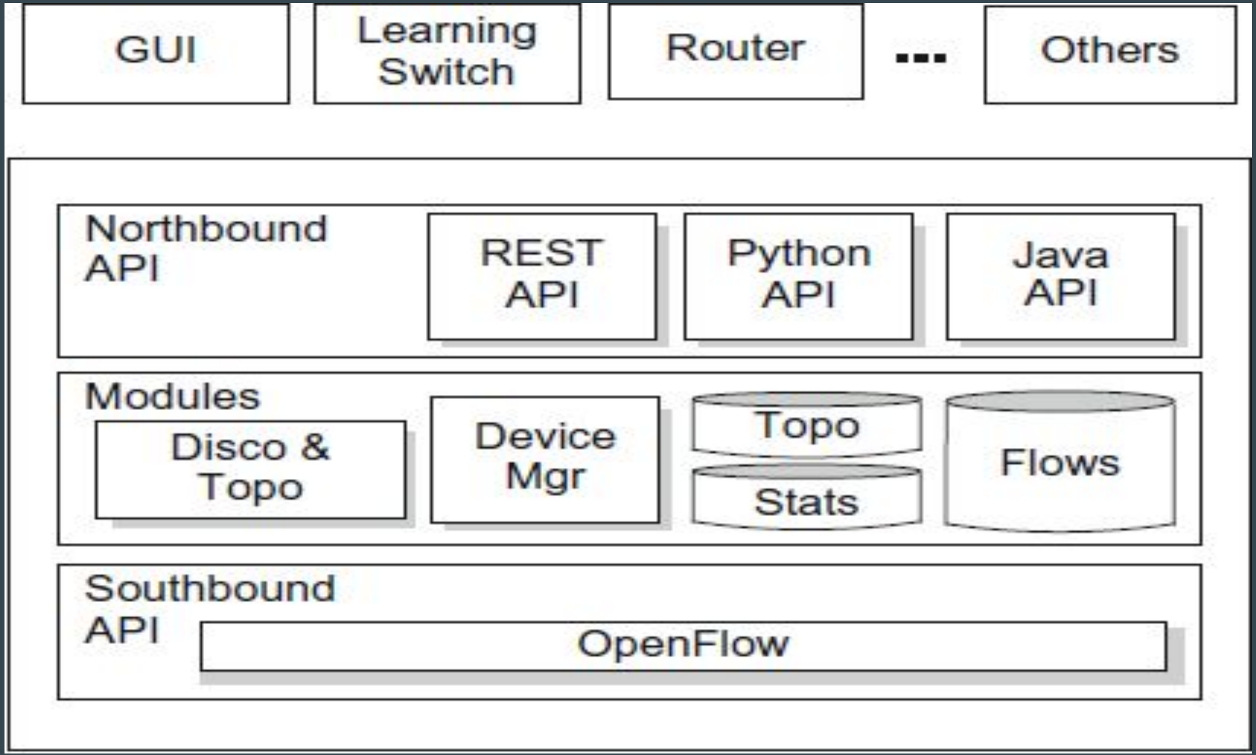
- Sudo apt-get install POX/Floodlight
- $ sudo mn
- Sudo mn-h
- mininet>nodes
- mininet>dump
- mininet > h1 ping h2
- mininet>h1 ping-cl h2
- Sudo mn-c

SDN Controller

SDN controller :

- A controller maintains a view of the entire network, implements policy decisions, controls all the SDN devices that comprise the network infrastructure and provides northbound API for applications.

SDN controller anatomy :



SDN controller :

- Floodlight controller includes a java API and a RESTful API.
- The opendaylight controller provides a RESTful API for applications running on separate machines.
- The northbound API represents an outstanding opportunity for innovation and collaboration among vendors and open source community.

SDN Controller core modules

SDN controller core modules :

- End user device discovery
- Network device discovery
- Network device topology management
- Flow management

SDN controller core modules :

- The core functions of the controller are device and topology discovery and tracking , flow management, device-management and statistics tracking.
- It maintains a flow cache that mirrors the flow tables on the various switches it controls.

SDN controller core modules :

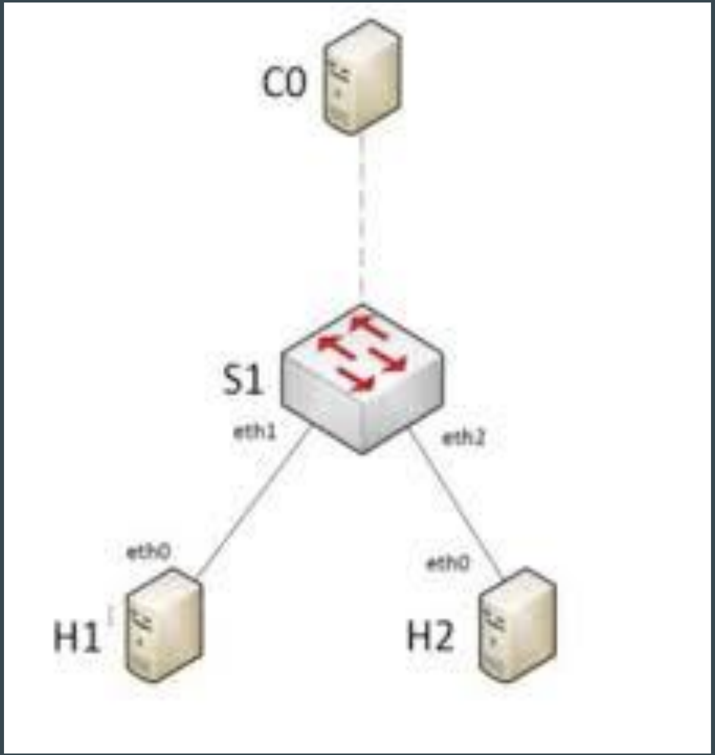
- The controller may provide high level APIs that give an abstraction of the network itself, so that the application developer need not be concerned with individual devices but rather with the network as a whole.
- Events are communicated from the controller to the application.

Mininet topologies

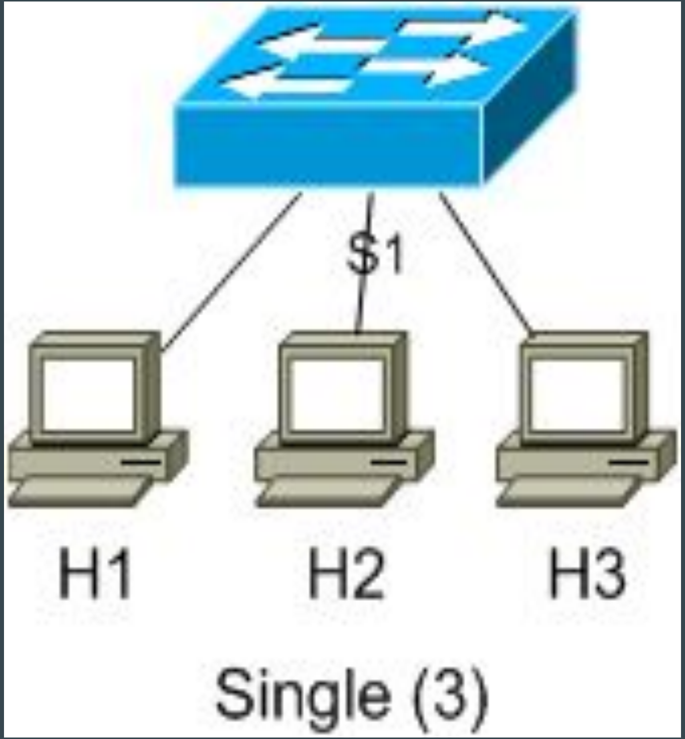
Mininet topologies :

- Minimal
- Single
- Reversed
- Linear
- tree

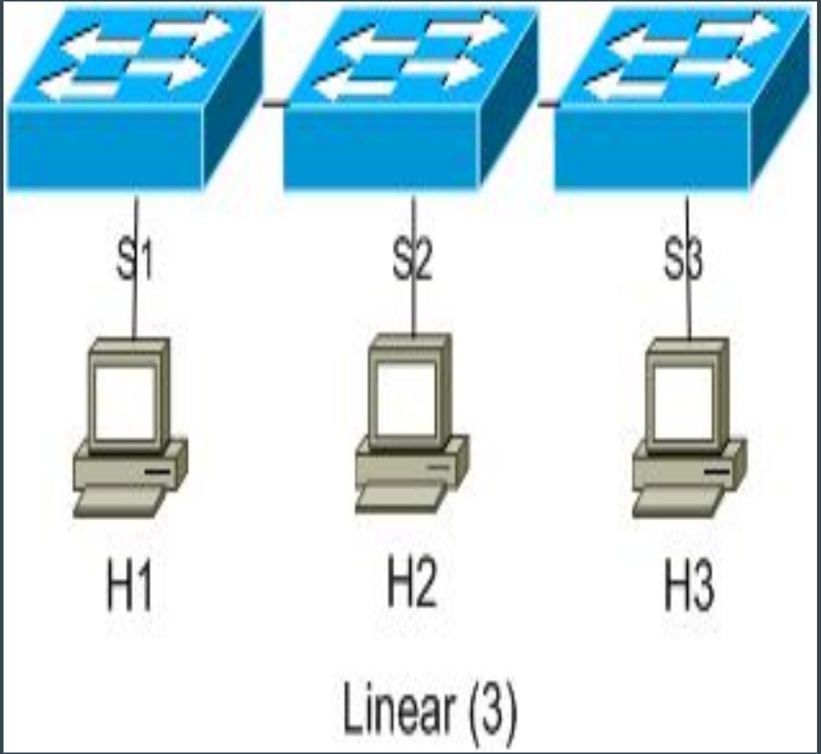
Minimal topology :



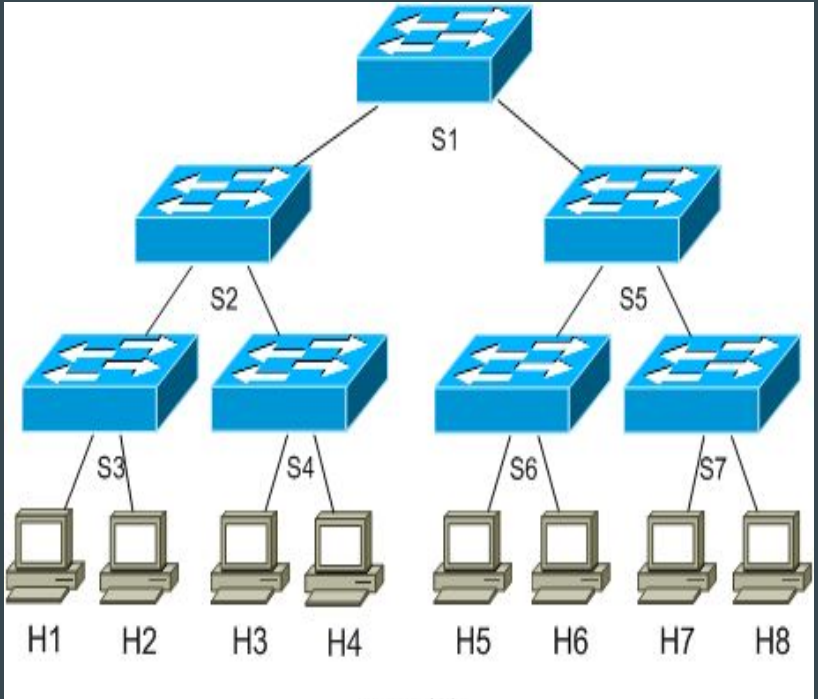
Single topology :



Linear topology :



Tree topology :



Tree (3)

Mininet topologies :

- To monitor the interactions between the controller and any switches in the network, we start wireshark and capture traffic on the mininet.
- We set wireshark to capture packet on the Mininet VMs loopback interface.

Mininet Installation

Mininet VM Installation :

- Download the Mininet pre-installed VM image.
- Download and install one of the hypervisors.
- Import VM image into selected hypervisor.

Native installation from source :

- Download source from github
- Full installation : Mininet + openVswitch + wireshark + etc.
- Minimum installation : Mininet + openVswitch

Mininet CLI usage :

Interact with hosts and switches

- $ sudo mn
- $ sudo mn -- controller = remote, ip= [IP_ADDDR] , port = [listening port]
- $ sudo mn -- custom [topo_script_path]--topo=[topo_name]
- mininet>nodes
- mininet>net
- mininet>dump

Mininet CLI usage :

Test connectivity between hosts

- mininet> h1 ping-c1 h2
- mininet> pingall

Mininet CLI usage :

Run a regression test

- mininet> iperf -s -u -p [port_num] &
- mininet> iperf -c [IP] -u -t [duration] -b [bandwidth] -p [port_num}

Link variations

- $ sudo mn -link tc, bw = [bandwidth] , delay = [delay_in_millisecond]

Mininet CLI usage :

Python interpreter

- $ py locals()
- $ py [mininet_name_space].[method]

Mininet API usage :

Low level API : nodes and links

- mininet.node.Node
- mininet.link.Link

Mininet API usage :

- MAC/set MAC : return/assign MAC address of a node or specific interface.
- IP/set IP : return/assign IP address of a node or specific interface.
- Cmd : send a command , wait for o/p, and return it.
- Terminate : send kill signal to node and clean up after it.
- Link : create a link to another node, make two new interfaces.

Mininet API usage :

Middle level API : network object
 mininet.net.Mininet

Add Host : add a host to network
Add Switch : add a switch to network
Add link : link two nodes into together
Add controller : add a controller to network
Get node by name : return node (s) with given name (s)
Start : start controller and switches
Stop : stop the controller, switches and hosts
Ping : ping between all specified hosts and return all data

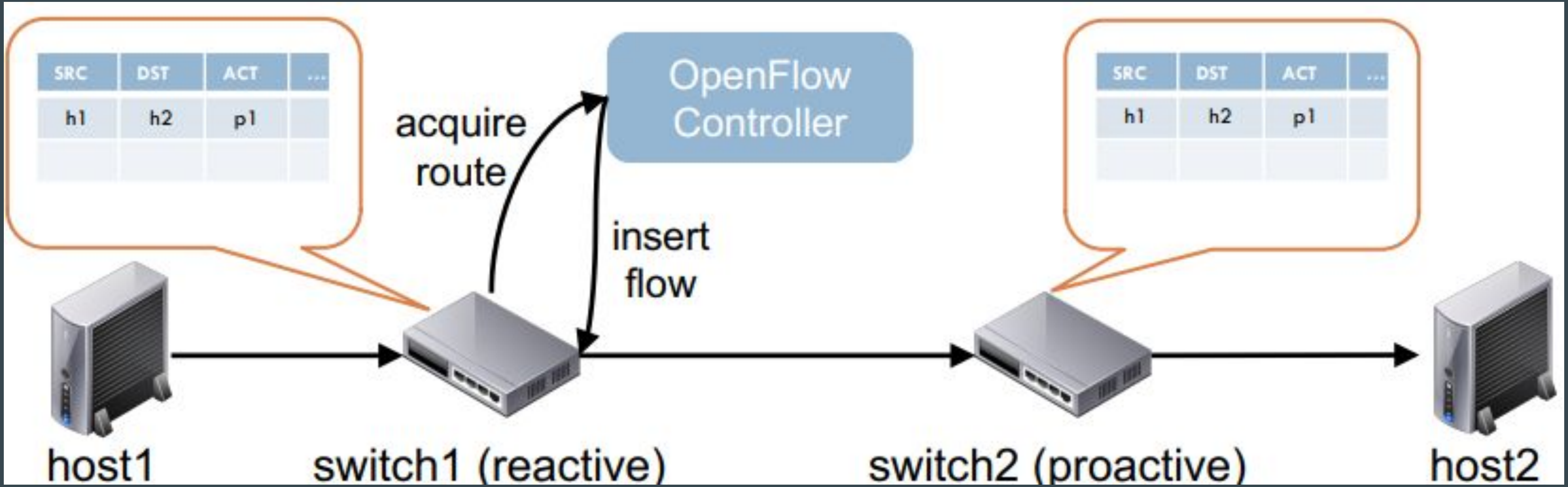
Mininet API usage :

High level API : topology template
mininet.topo.Topo

Methods similar to net : e.g add Host, add switch , add link
Add Node : add node to graph
Add port : generate port mapping for new edge
Switches : return all switches
Hosts/nodes/switches/links : return all hosts
Isswitch : return true if node is a switch, return false otherwise

Two flow insertion methods :

- Re - active flow insertion
- Pro - active flow insertion



Development tools

Development tools :

- Openflow controller
- Openflow switch
- Ovs-ofctl
- Wireshark
- Iperf
- Mininet
- cbench

Mininet commands :

- mininet>nodes
- mininet>help
- mininet>h1 if config
- mininet> xtern h1 h2
- $ sudo mn - c

Mininet VM setup

Mininet VM setup :

- Once you have downloaded the .ovf image, start up virtual box, then
  Select file > import appliance and select the .ovf image that you downloaded.
- Able to simply double- click the .ovf file to open it up in your installed virtualization program. Press the “import” button.
- Unpacked image is about 3GB.

Setting up the VM for ssh access :

- If running a virtualbox, you should make sure your VM has two network interfaces.
- One should be a NAT interface that it can use to access the internet, and other should be a host only interface to enable it to communicate with the host machine.
- Both interfaces should be configured using DHCP.

Setting up the VM for ssh access :

- From the virtual machine console, login to the VM, then enter
  $ ifconfig -a
- You should see three interfaces (eth0, eth1, lo) , both eth0 and eth1 should have IP address assigned. If this is not the case, type
  $ sudo dhclient ethX
  For the access to the VM:
  $ ssh - X [user] @ [guest IP here]

Using the VM GUI :

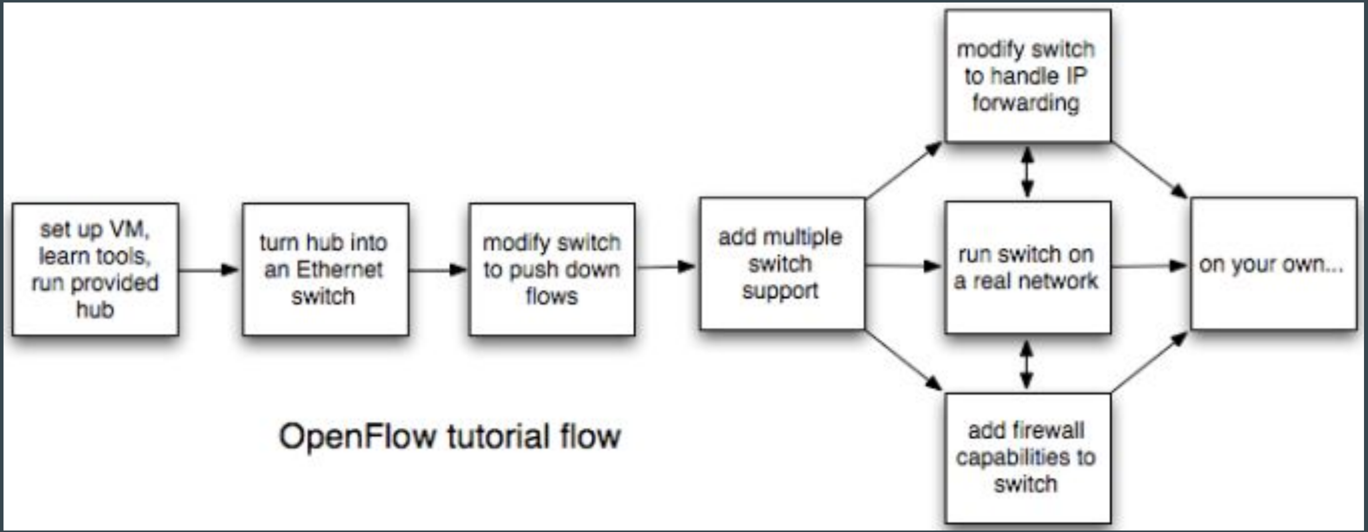
- Log in to the VM console window and type :
  $ sudo apt-get update && sudo apt-get install xinit lxde virtualbox - guest-dkms

- At the point, you should be able to start an X11 session in the VM console window by typing :
  $ startX

Open Flow

Open Flow :

- It is an open interface for remotely controlling the forwarding tables in network switches , routers and access points.


set up VM, learn tools, run provided hub
turn hub into an Ethernet switch
modify switch to push down flows
add multiple switch support
modify switch to handle IP forwarding
run switch on a real network
add firewall capabilities to switch
on your own...
OpenFlow tutorial flow

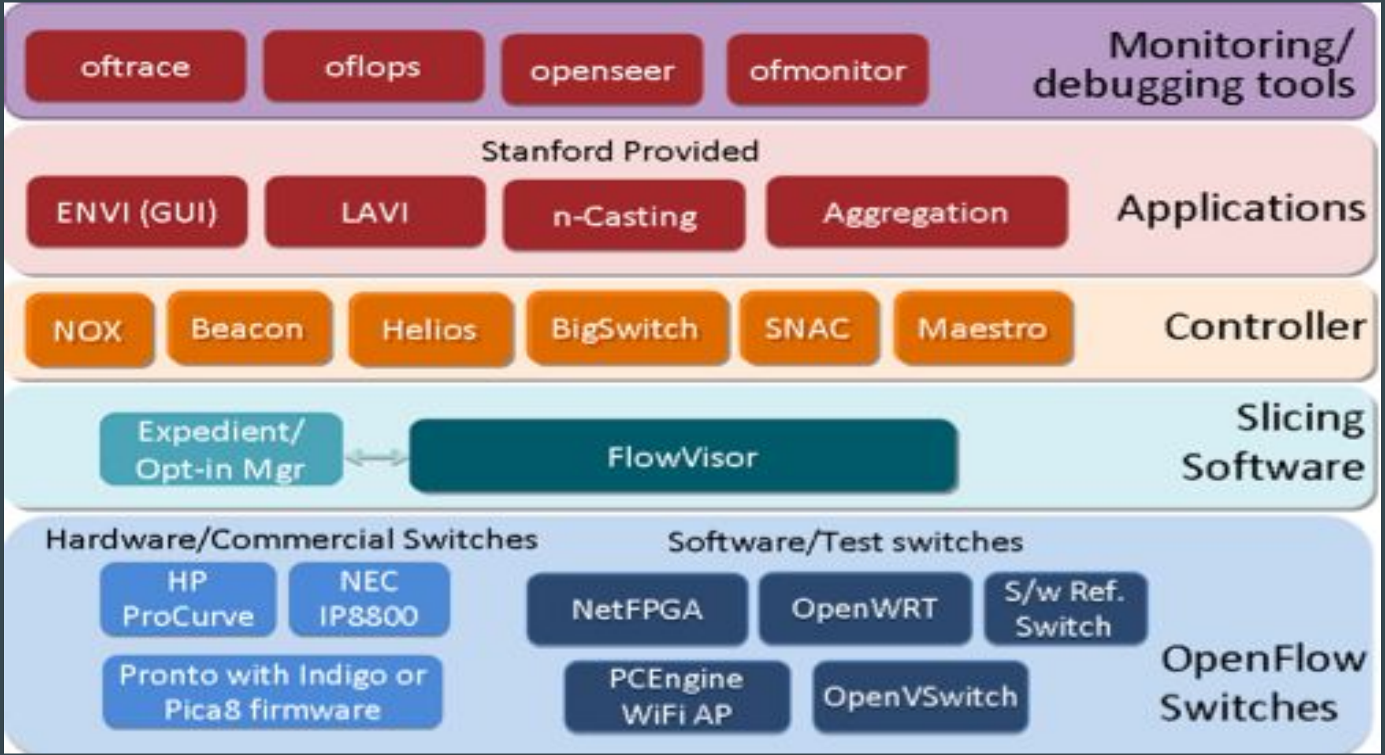
Open flow :

- It is the communication protocol in SDN environments that enables the SDN controller to directly interact with the forwarding plane of network devices such as switches and routers.
- Programmability
- Centralized intelligence
- Abstraction

Openflow controller :

- It is a type of SDN controller that uses the openflow protocol.
- An openflow controller uses the openflow protocol to connect and configure the network devices to determine the best path for application traffic.
- An SDN controller relays information to the switches / routers and the applications and business logic.

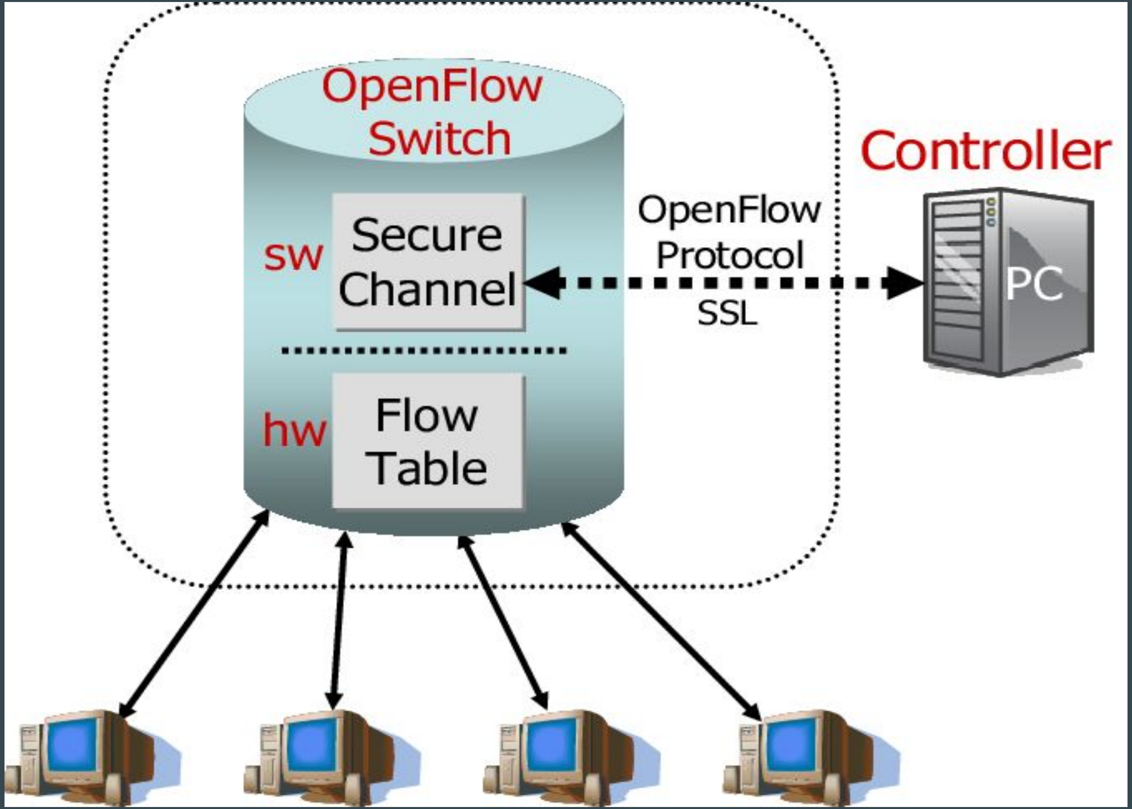
Openflow controller :



Open Flow components

Openflow components :

- Flow table
- Controller
- Secure channel



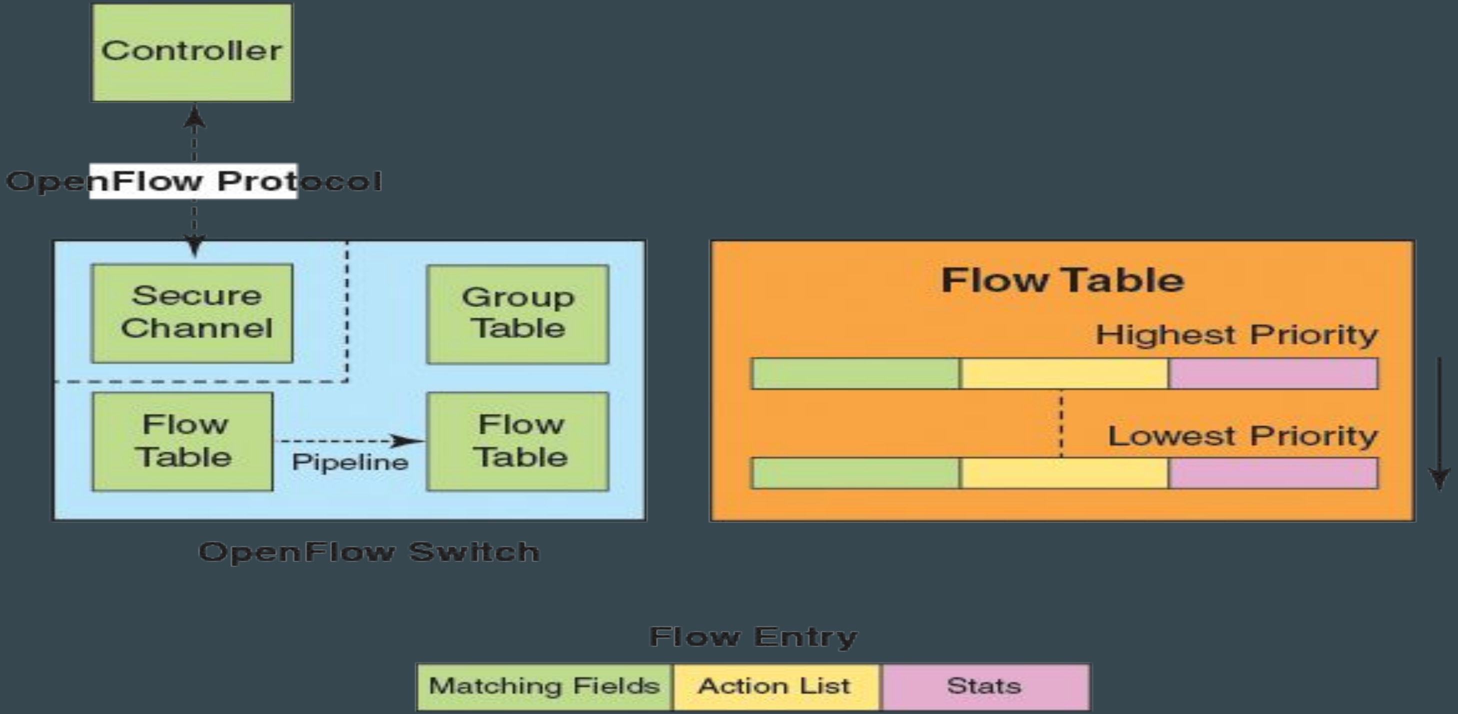
Flow entries :

- Ingrees port
- VLAN ID
- Ethernet : source, destination, type
- IP : source, destination, protocol
- TCP : source, destination

Actions :

- Forward traffic to a given port or ports.
- Encapsulate and forward traffic to the controller for inspection.
- Drop traffic
- Forward traffic through switch’s normal processing pipeline.

Flow entries and Actions :



Switches :

- Dedicated openflow switches : dumb datapath element that forwards packets ports.
- Openflow enabled switches : re-use existing hardware and software
  TCAMs in hardware
  Secure channel and openflow protocol in software
- Perform openflow experiments while not interfering with normal production traffic.

Scope of openflow :

- It defines the exact format of messages that switch and controller used to communicate with each other.
- It defines what exactly a controller can configure in a switch , list of items and tasks.
- It defines all communication components between the switch and a controller.

Scope of openflow :

- A table that implements classification and forwarding logic is popularly known as flow table.
- The packet classification process is a basic rule of type LHS -> RHS.
- RHS is basically an action to be performed.

Scope of openflow :

- Openflow allows a TCP connection from the controller to the switch for sending commands.
- It is recommended to use TCP instead to provide a secured option.

Matching :

- Openflow processing is the matching of an incoming packet with some rule.
- The fields of the incoming packets are used for matching.
- The openflow 1.4 specification allows 41 fields of a packet for classification.
-

Pipelining :

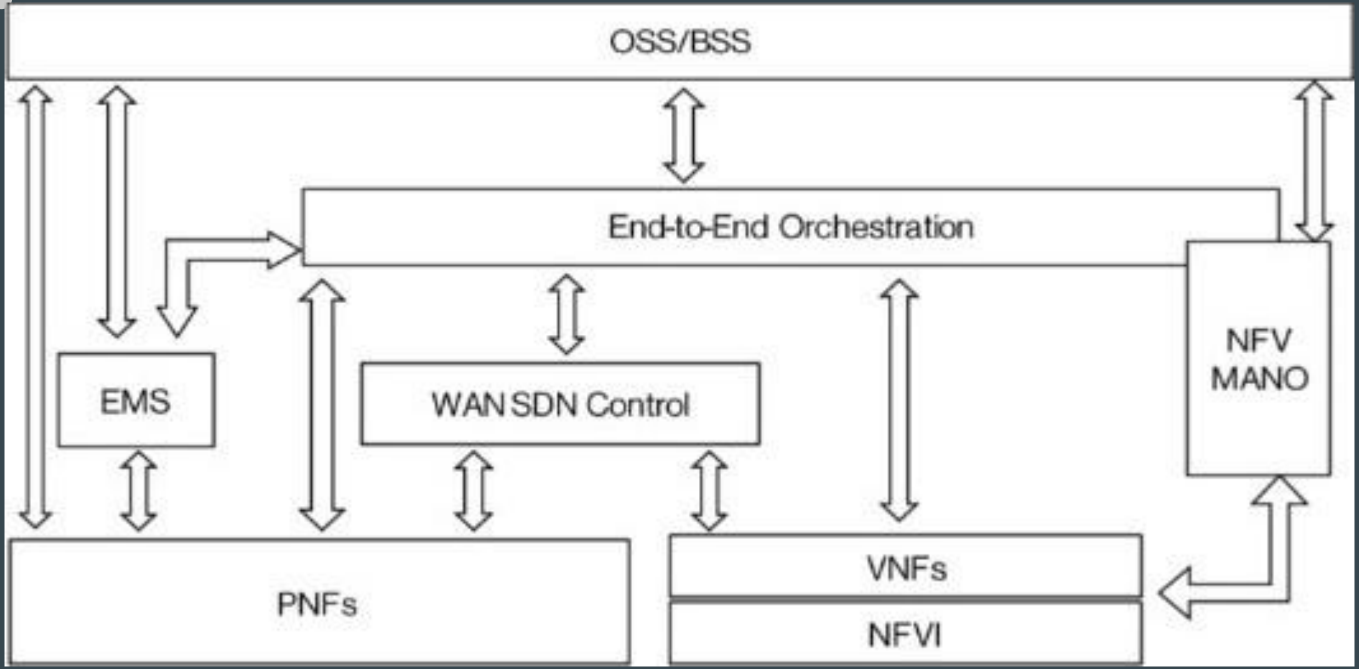
- The process which links flow tables is known as pipelining.
- Flow tables can be synchronized.
- Synchronization is either single or bi-directional.
- Additional field, metadata is also used in pipelining process.

NFV

NFV :

- This concept is based on the use of COTS hardware for general purpose compute, storage and network.
- Software functions necessary for running the network are now decoupled from the hardware.
- NFV enables deployment of virtual network functions as well as network services described as NF forwarding graphs of interconnected NFs and end points within a single operator network or between different operator networks.

End to end orchestration :



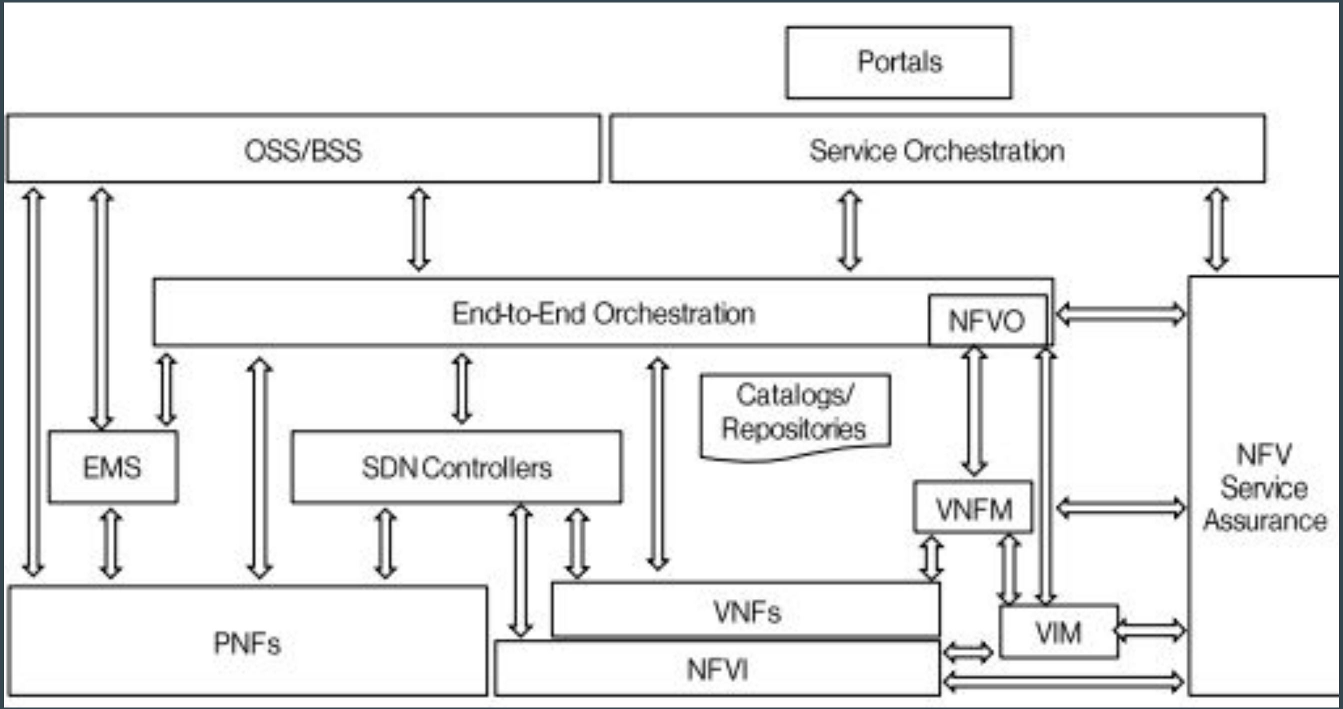
E2E orchestration :

The management and control complex has three main building blocks :

- NFV MANO
- WAN SDN control
- EEO
-

NFV Architecture

Architecture Framework :



Architecture Framework :

- NFVI - network function virtualization infrastructure
- VNF - virtualized network function
- PNF - physical network function
- VIM - virtualized infrastructure manager
- VNFM - VNF manager
- EEO - end to end orchestration
- catalogs/repositories
- Service assurance

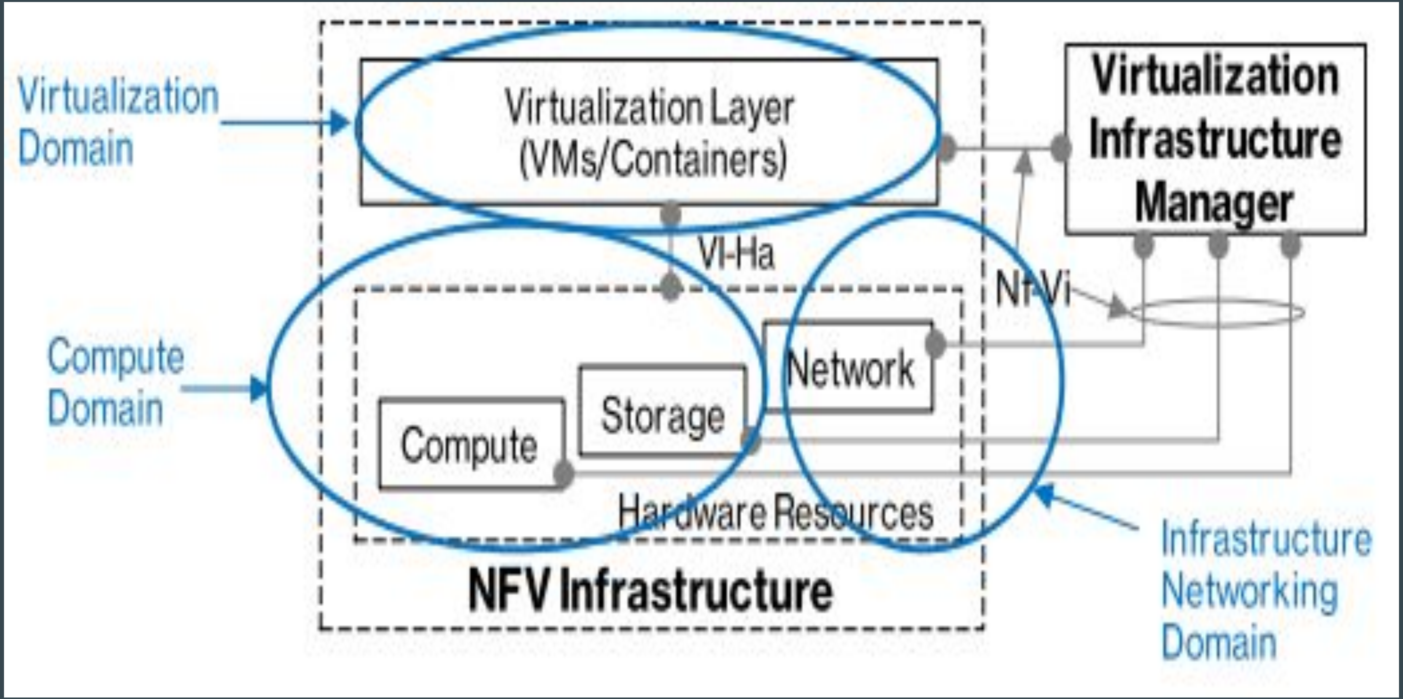
Architecture Framework :

- Data centre SDN controller
- WAN SDN controller
- Access SDN controller
- domain/vendor-specific controller
- Service orchestration
- Portals
- Element management system
- Operations support systems and business support system

NFV Infrastructure :

- It comprises all hardware and software components building up the environment in which VNFs are deployed, managed and executed.
- It includes hardware resources, virtualization layer and virtualized resources , CPU/chipset , forwarding box and their associated interfaces.

NFV Infrastructure :


Virtualization Domain
Virtualization Layer (VMs/Containers)
Virtualization Infrastructure Manager
Vl-Ha
Nf-Vi
Compute Domain
Network
Storage
Compute
Hardware Resources
NFV Infrastructure
Infrastructure Networking Domain

Platform aspects of NFVI

NFV domains :

- Compute domain : includes compute and storage resources that are commonly pooled.
- Virtualization domain : includes the virtualization layer for VM’s and containers that abstracts the hardware resources and decouples the VNF from the underlying hardware.
- Infrastructure networking domain : includes both physical and virtual networking resources that interconnect the computing and storage resources.

NFV domains :

- The infrastructure networking domain is characterised by mainstream physical and virtual switches.
- User space DPDK enabled vSwitch
- Single root I/O virtualization
- PCI-passthrough

VNF Manager

Role & overview of VNF Manager :

- VNF manager is an entity that is part of the NFV architecture framework defined by ETSI NFV that interacts with the VIM , the NFVO , the VNF’s and the service assurance.
- The VNF manager is responsible for the lifecycle manager of VNF instances.
- Each VNF instance is associated with the VNF manager.

Role & overview of VNF Manager :

- The deployment and operational behavior of each VNF is captured in a template called virtualized network function descriptor (VNFD) that is stored in the VNF catalog.
- NFV-MANO uses a VNFD to create instance of the VNF it represents and to manage lifecycle of those instances.
- A VNFD has a one to one correspondence with a VNF package and it fully describes the attributes and requirements necessary to realize such a VNF.

Role & overview of VNF Manager :

- Different versions of a VNF package may correspond to different implementations of the same function, different versions to run in different execution environments or different release versions of the same software.
- The repository may be maintained by the NFVO or another external entity.

VNF descriptors

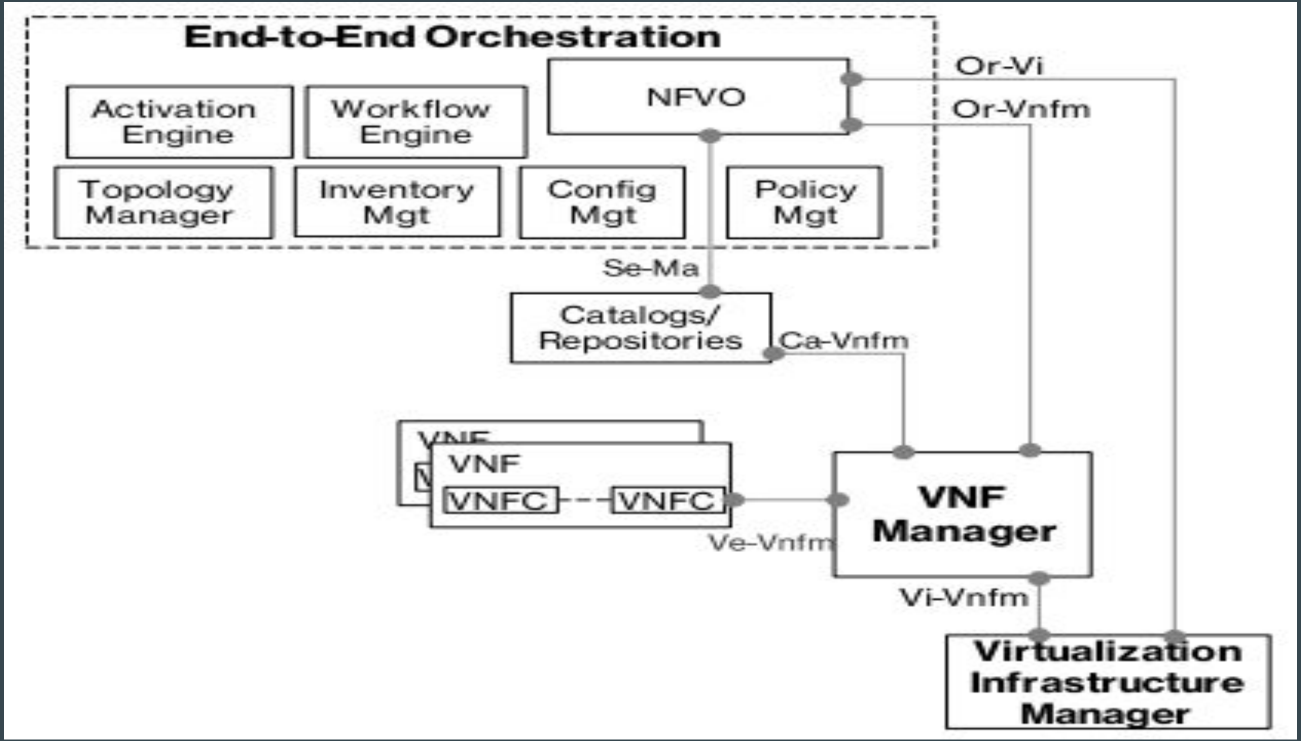
VNF Descriptors :

- In the NFVI, inter-component communication is exposed and supported by NFVI ; therefore the intra-VNF virtual links must be defined as part of the VNFD to ensure VNF’s operate properly.
- A VNF descriptor is a deployment template which describes a VNF in terms of deployment and operational behavior requirements.

VNF Descriptors :

- The VNFD also contains connectivity, interface and KPI requirements that can be used by NFV-MANO functional blocks to establish appropriate virtual links within NFVI between VNFC instances, or between a VNF instance and the endpoint instance to other NF’s.

VNF Descriptors :



VNF Descriptors :

- The Vi-Vnfm interface is used to define the required platform and network characteristics required for each VNFC. the VNFM lifecycle management functions generate specific API calls to the VIM using information elements present in VNFD .
- The Ve-Vnfm interface is used to perform additional configuration and post-processing tasks on the VNFC’s.

End- to- End orchestration

Introduction :

- EEO is responsible for lifecycle management of end-to-end network services.
- EEO manages end-to-end network services, including connectivity through the network infrastructure outside the data centre, which typically consists of PNF’s.
- EEO allocates resources, triggers instantiation of VNF’s and configures network functions in order to activate the end-to-end service.

Key functions :

- Resource allocation and orchestration
- Workflow execution
- Topology and inventory management
- Configuration and activation

SDN Controller Framework

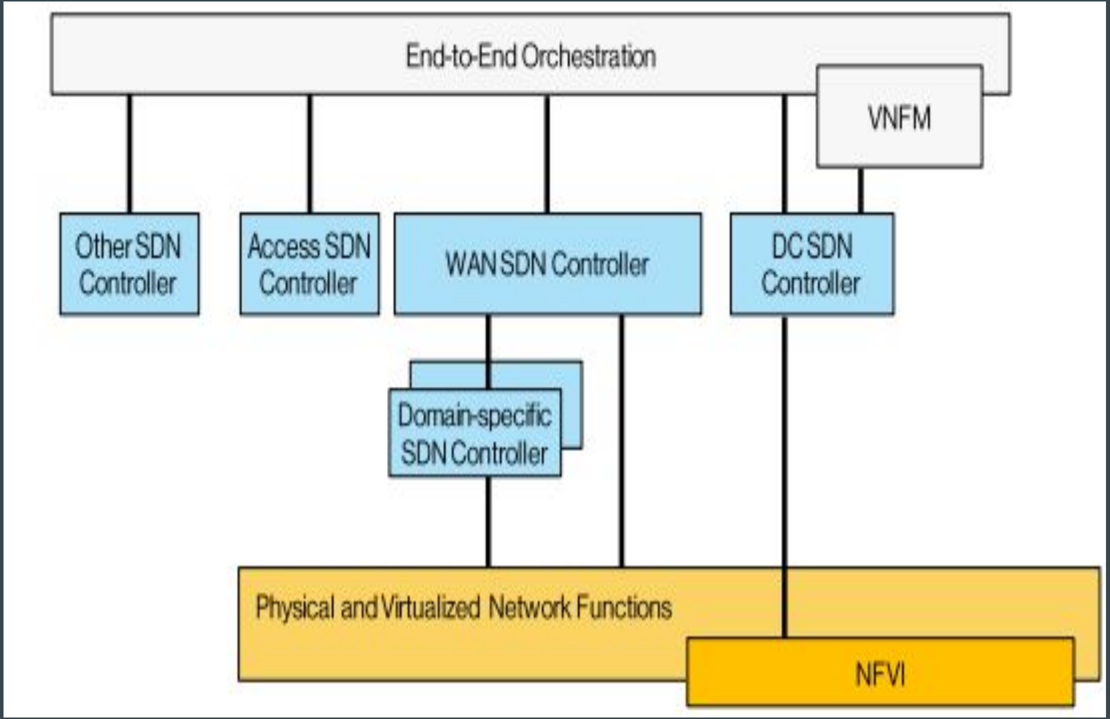
Management & control architecture :

- OSS/BSS systems leverage APIs and interfaces provided by these EMS systems to drive business operations in to the network.
- The goal of SDN is to separate the management and control planes from the data plane networking functions themselves.

SDN control architecture :

- The SDN network architecture proposed decoupling of network control from packet forwarding and a direct programmability of the network control function.
- Network intelligence is centralized in software based SDN controllers, which maintain a global view of the network.

SDN - NFV Reference architecture :



SDN control architecture :

Control across domains occurs in two forms :

- Through the end to end orchestrator
- Through an SDN controller

SDN Controller Framework

SDN control architecture :

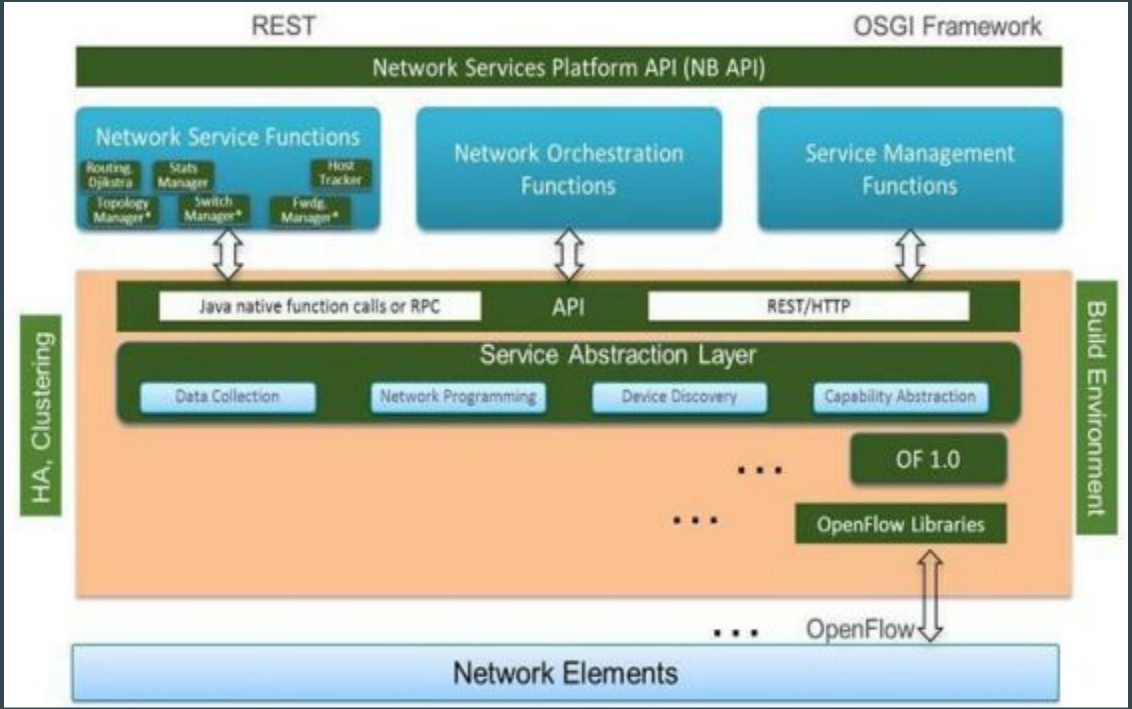
Control and

Opendaylight controller

Introduction :

- Goal : to promote SDN and NFV.
- It is a modular open platform for customizing and automating networks of any size and scale.
- It is driven by a global , collaborative community of vendor and user org. That continuously adapts to support the industry broadset of SDN & NFV use cases.

Introduction :



Introduction :

This architecture enables the controller to be :

- Flexible
- Scalable in the development process
- Run time extensible

Opendaylight architecture :

Model driven

- The core of the opendaylight platform is the model-driven service abstraction layer.
- In opendaylight , underlying network devices and network applications are all represented as objects as models, whose interactions are processed within the SAL.

Opendaylight architecture :

- The opendaylight controller will start with an openflow 1.0 southbound plugin.
- These modules are linked dynamically into a Service Abstraction Layer (SAL).
- The SAL figures out how to fulfill the requested service irrespective of the underlying protocol used between the controller and the network devices.

Opendaylight architecture :

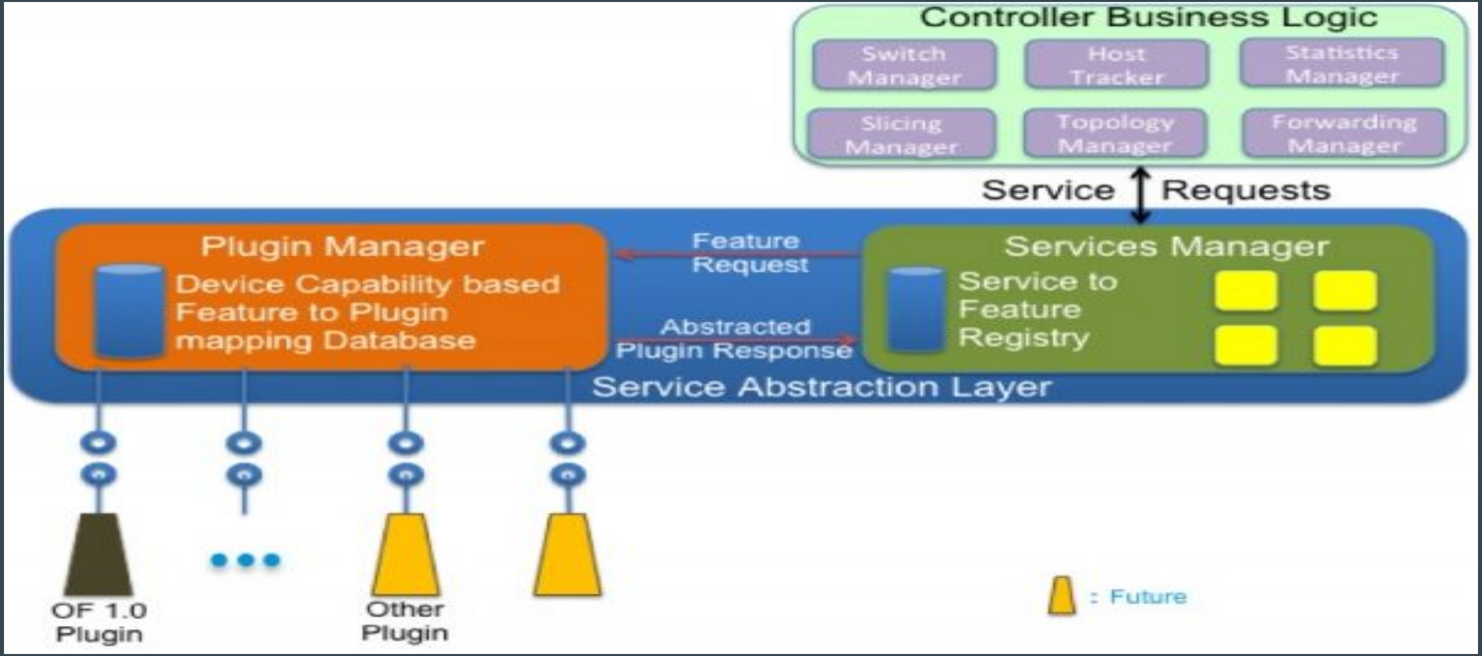
- The controller exposes open Northbound API's which are used by applications.
- Support for OSGi framework and bidirectional REST for the northbound API.
- OSGi framework is used for apps that will run in the same address space as controller.
- REST API is used for apps that do not run in the same address space as controller.

Service Abstraction Layer

Service Abstraction layer :

- It is at the heart of the modular design of the controller and allows it to support multiple protocols on the southbound and providing consistent services for modules and apps.
- The SAL provides basic services like device discovery which are used by modules like topology manager to build the topology and device capabilities.

Service Abstraction layer :

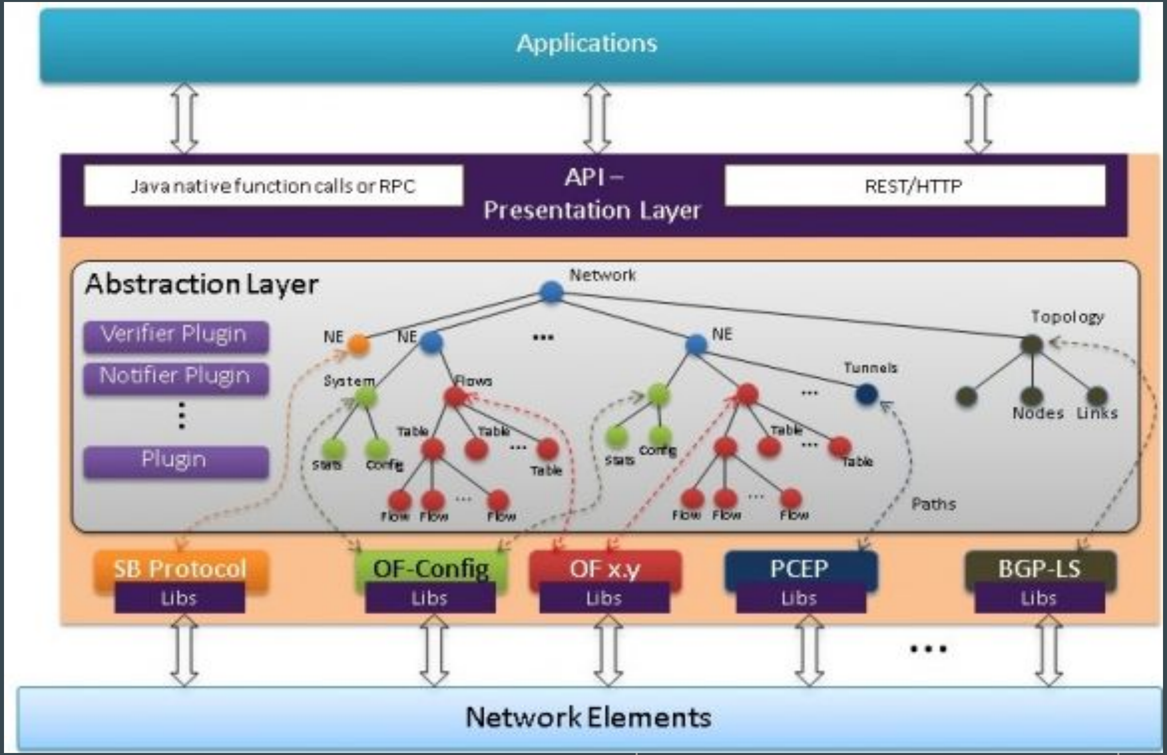


SAL services :

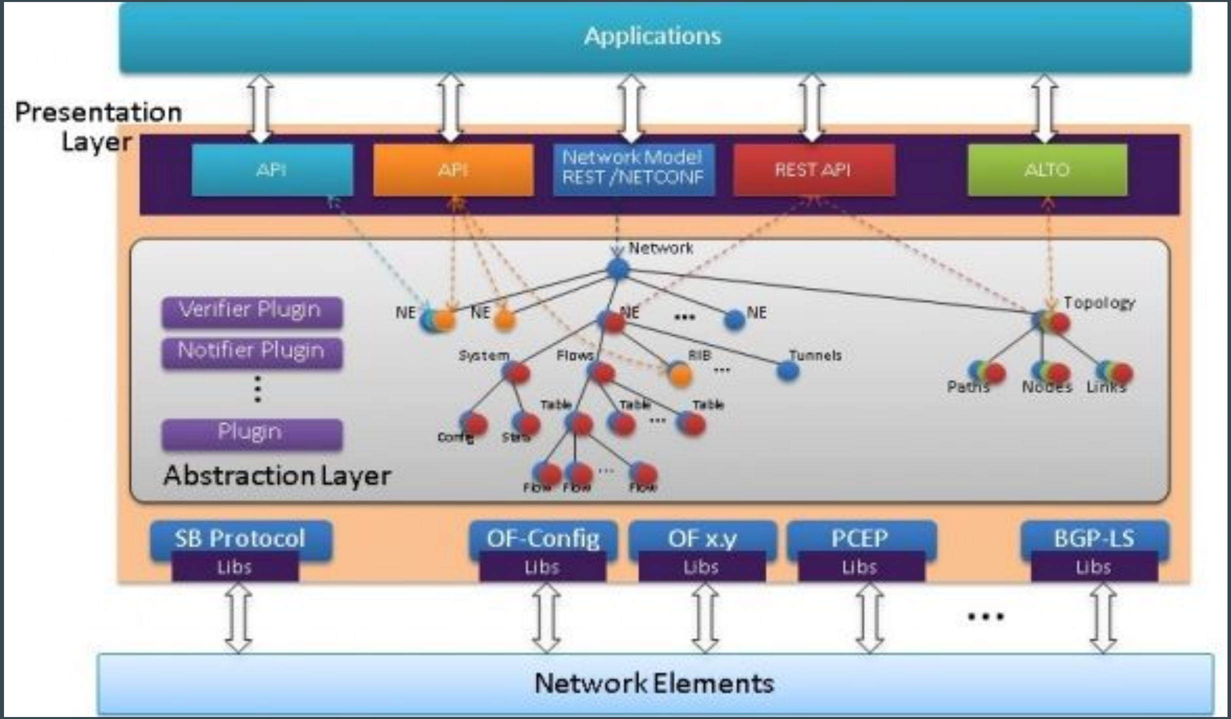
- iListenDatapacket
- IDatapacketservice
- IPluginOutDataPacketService
- IPluginInDataPacketService

Evolution of the controller SAL

Evolution :


Applications
Java native function calls or RPC
API – Presentation Layer
REST/HTTP
Abstraction Layer
Verifier Plugin
Notifier Plugin
Plugin
Network
NE
NE
NE
Topology
System
Flows
Tunnels
Stats
Config
Table
Table
Table
Flow
Flow
Flow
Nodes
Links
Paths
SB Protocol
OF-Config
OF x.y
PCEP
BGP-LS
Libs
Network Elements

SAL services :



Switch Manager :

- The switch manager API holds the details of the network element.
- As a NE is discovered, its attributes are stored in the database by the switch manager.

High availability :

- The opendaylight controller supports a cluster based high availability model.
- To make controller highly available, we need to add resilience at: controller level , make sure that the OF enabled switches are multi-homed to multiple instances of the controller.

Hypervisor

Hypervisor :

- It is a function which abstracts , isolates , operating systems and applications from the underlying computing hardware.
- It is sometimes known as virtual machine monitor.
- Hypervisors provide several benefits to the enterprise data center.

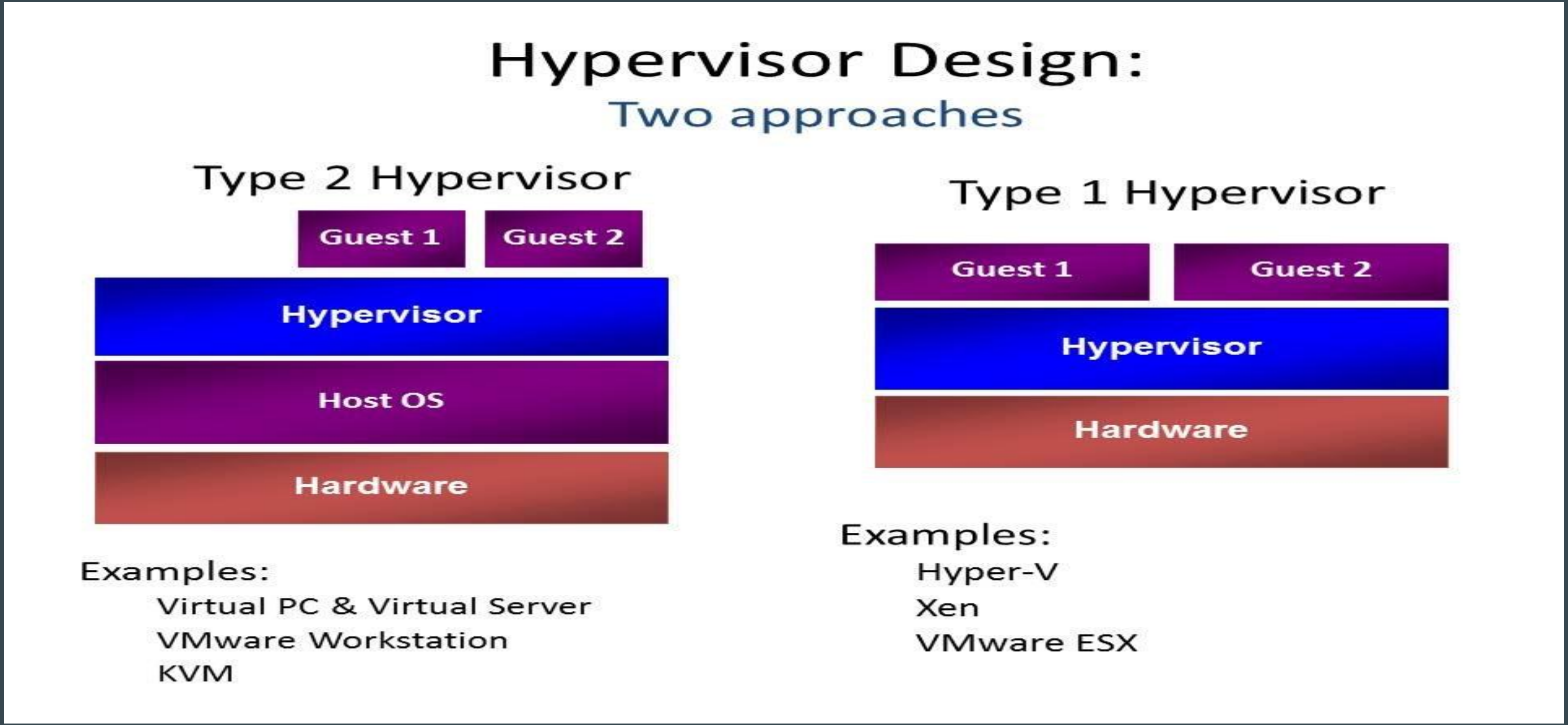
Hypervisor :

- The ability of a physical host system to run multiple guest VM’s can vastly improve the utilization of the underlying hardware.
- The abstraction that takes place in a hypervisor also makes the VM independent of the underlying hardware.

Hypervisor types :

- Hypervisors are implemented as a software layer - such as VMware vSphere or Microsoft Hyper-V.
- Hypervisor can also be implemented as code embedded in a system’s firmware.
- There are two principle types of hypervisor.
- Type 1 : Vsphere or hyper-v
- Type 2 : VMware player or parallel desktops.

Hypervisor types :



Examples:
- Virtual PC & Virtual Server
- VMware Workstation
- KVM

Examples:
- Hyper-V
- Xen
- VMware ESX

Hypervisor

Hypervisor uses :

- Hypervisor are important to any system administrator or system operator because virtualization adds a crucial layer of management and control over the data center and enterprise environment.
- Storage hypervisors are key element of software defined storage.

Hypervisor uses :

- Networks are also being virtualized with hypervisors, allowing networks and network devices to be created, changed, managed and destroyed entirely through software without ever touching physical network devices.

OpenFlow

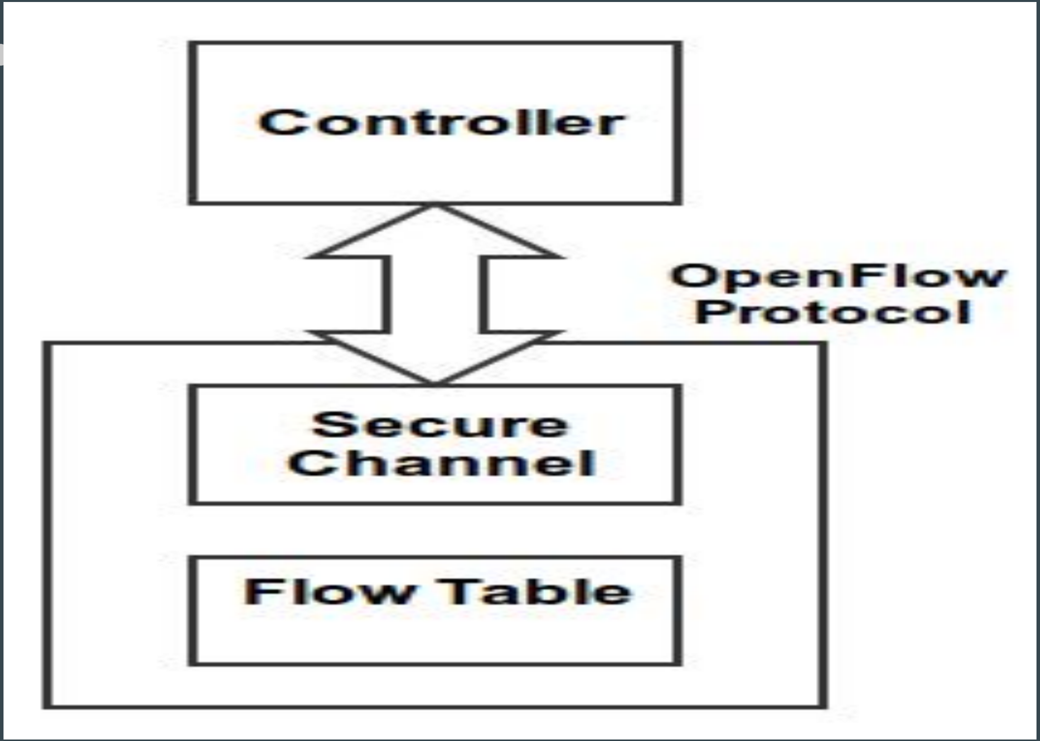
OpenFlow :

- OpenFlow protocol addresses the two key requirements needed to turn the concept of SDN in to practical implementation.
- There must be a common logical architecture in all network devices to be managed by an SDN controller.
- A standard , secure protocol is needed between SDN controller and the network device.

OpenFlow components :

- An SDN controller communicates with openflow-compatible devices using the openflow protocol running over the secure sockets layer (SSL).
- Two major components of openflow are flow table and secure channel protocol.

OpenFlow components :



Flow table :

- The basic building block of openflow is flow table.
- Each packet that enters a device passes through one or more flow tables.

Header	Action	Counters

Flow table :

Header field : each flow table header entry is made of 6 components , which defined the matching rules and basic rules for the corresponding flow.

- Match fields
- Priority
- Counters
- Instructions
- Timeouts
- Cookie

Flow table :

Action : each flow entry is associated with zero or more actions that dictate how the device handles matching packets.

- Forward
- Drop
- Modify field

Secure channel :

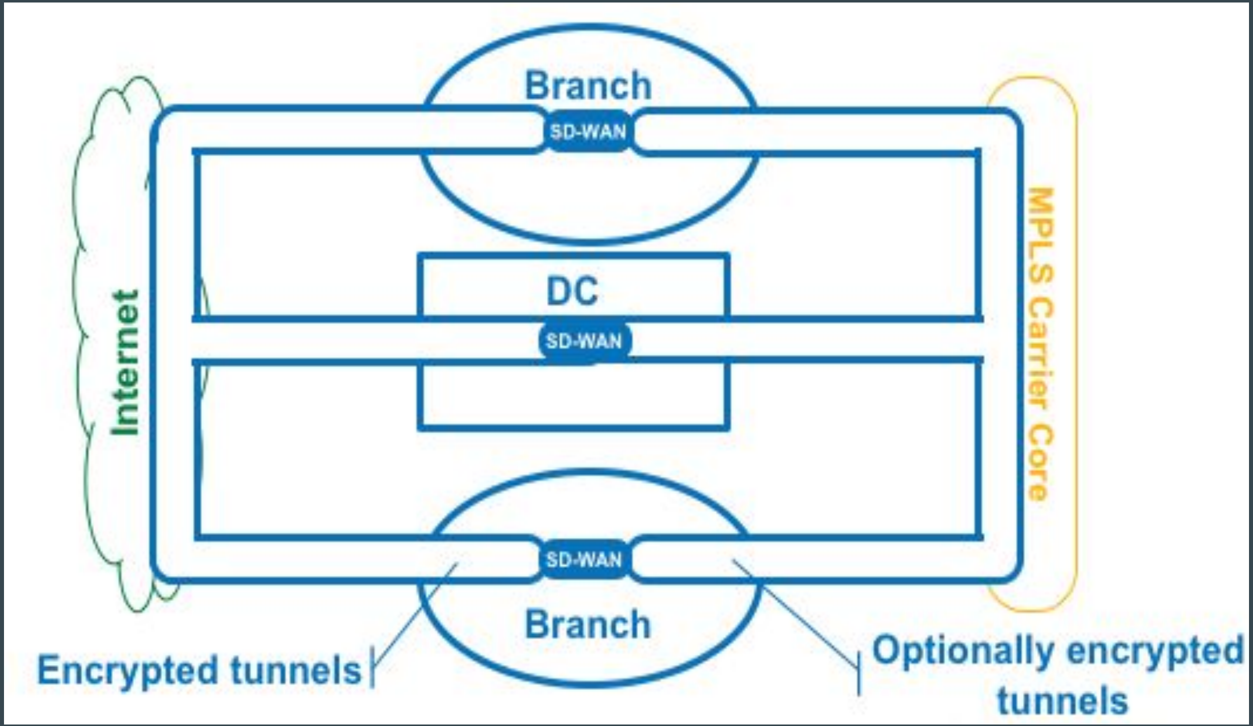
- The OF protocol describes message exchanges that take place between an OF controller and an OF device.
- It supports three types of messages :
  Controller to device
  Asynchronous
  symmetric

SD-WAN

SD-WAN :

- It is a specific application of SDN technology applied to WAN connections such as broadband internet, 4G, LTE or MPLS.
- It relies on four central components :
  Edge connectivity abstraction
  WAN virtualization
  Policy- driven , centralized management
  Elastic traffic management

SD-WAN :


Branch
SD-WAN
Internet
DC
SD-WAN
MPLS Carrier Core
SD-WAN
Branch
Encrypted tunnels
Optionally encrypted tunnels

Why SD-WAN..? :

- Security
- Centralized orchestration
- Network visibility
- Provider reliability
- Performance
- Scalability
- Connection
- Flexibility

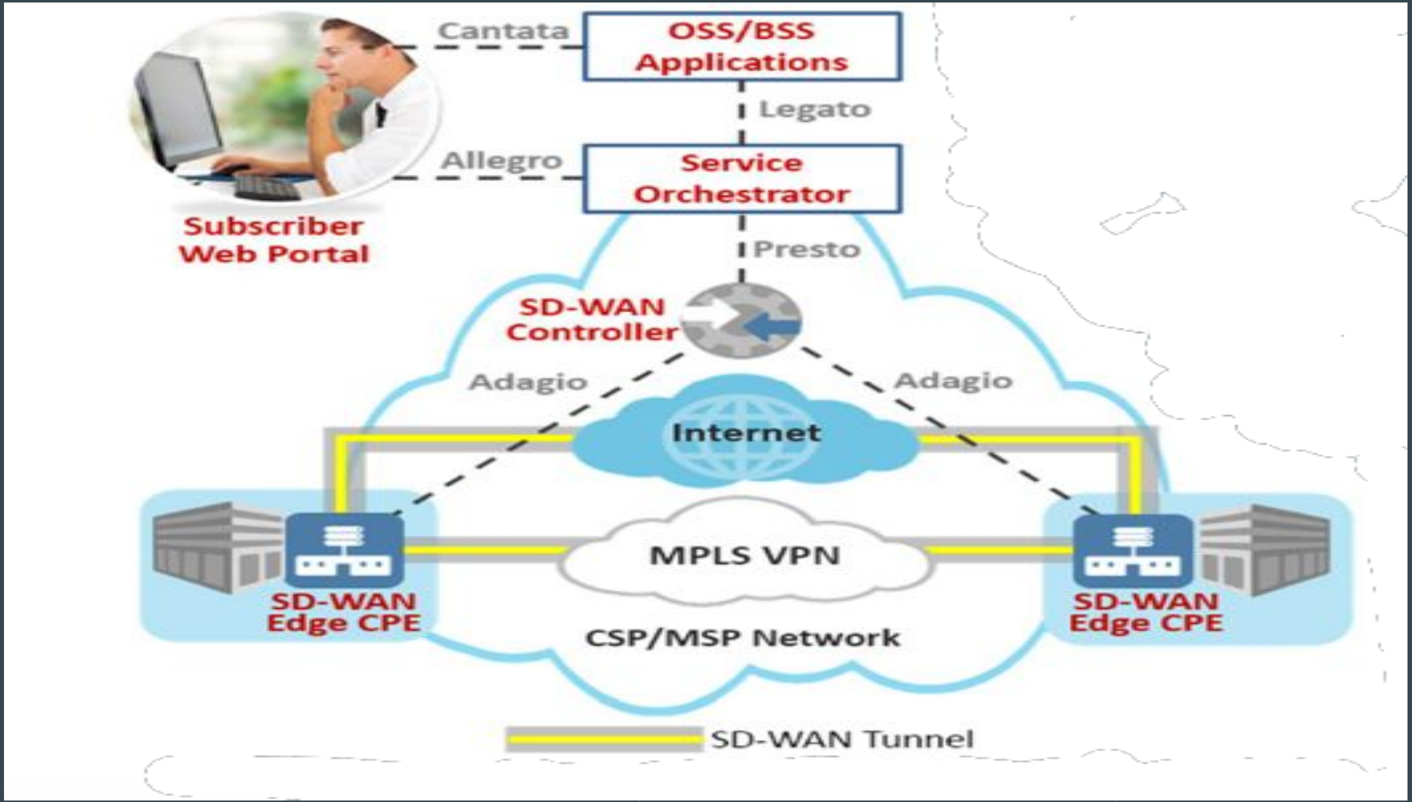
SD-WAN characteristics :

- SD-WAN implementation is with incorporating newer SDN, NFV and service orchestration technologies.
- These technologies provides the integration and service deployment automation that has made SD-WAN managed services so compelling.

SD-WAN service components :

- SD-WAN Edge
- SD-WAN Controller
- Service Orchestrator
- SD-WAN gateway
- Subscriber web portal

SD-WAN service components :



vCPE

vCPE :

- It is a way to deliver network services such as routing , firewall security and VPN connectivity to enterprises by using software rather than dedicated hardware devices.
- vCPE also known as cloud CPE, abstracts the intelligence of such devices into software based functionality that resides in a remote data center.

vCPE :

- It is a leading application to move hardware appliance functionality to software on commercial platforms that way data centers have.
- The customer premises are natural fit for vCPE deployments.
- Multiple functions can be loaded onto a single server, simplifying management and facilitating more rapid upgrades.

vCPE benefits :

- For service providers
- For enterprises
- Lower equipment costs
- Elimination of truck rolls
- The ability to shop for best of breed solutions
- Freedom to customize services to specific customer needs.

SDN & NFV for mobile EPC

SDN & NFV for mobile EPC :

- For service providers
- For enterprises
- Lower equipment costs
- Elimination of truck rolls
- The ability to shop for best of breed solutions
- Freedom to customize services to specific customer needs.

SDN for mobile EPC :

- SDN is essentially applied to decouple the control and data planes of the EPC gateways i.e SGW and PGW.

SDN for mobile EPC :

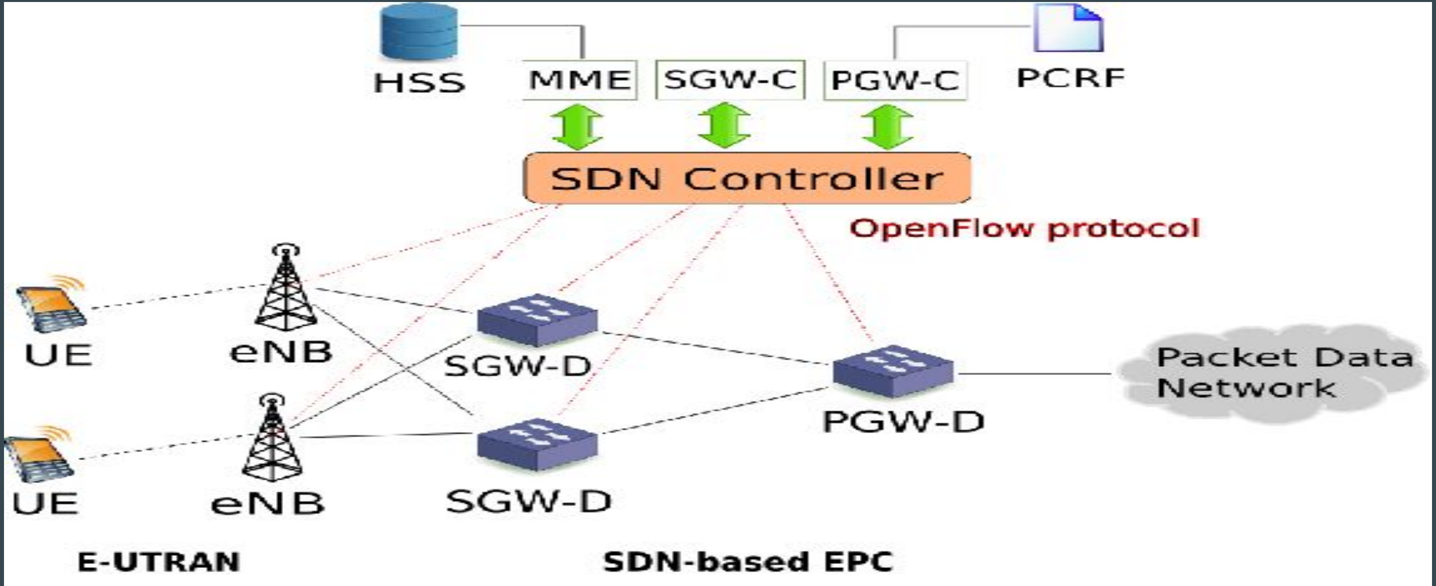
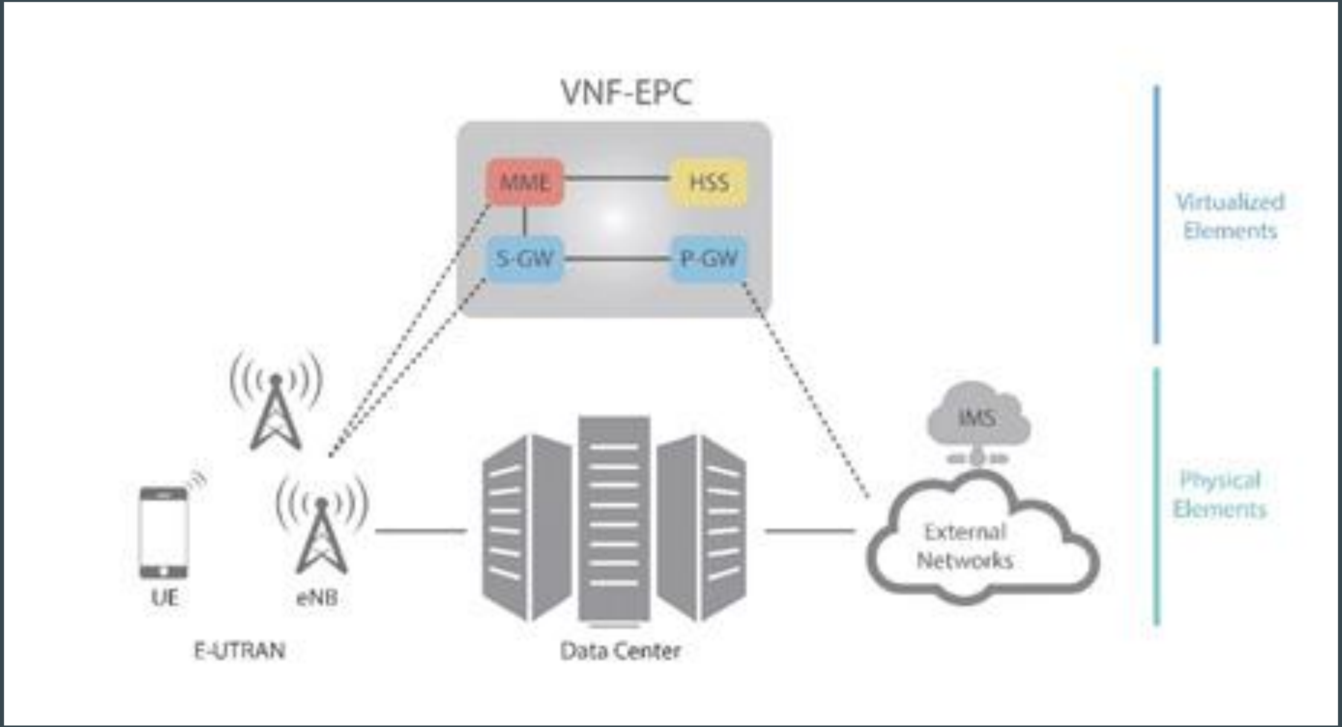


Fig. 2: A typical SDN-based LTE EPC.

NFV for mobile EPC :

- NFV MANO also allows the integration with external OSS/BSS .
- NFV promises significant cost saving either in deployment or in operation for mobile operators.

NFV for mobile EPC :



NFV MANO

NFV MANO :

- With NFV management and organization (MANO) , management of NFV is now addressed by the MANO stream.
- It is the ETSI-defined framework for the management and orchestration of all resources in the cloud data center.
- This includes computing, networking,storage and virtual machine resources.

NFV MANO :

NFV MANO is broken up into three functional blocks :

- NFV orchestrator
- VNF manager
- Virtualized infrastructure manager

NFV orchestrator :

- NFV orchestration is used to coordinate the resources and networks needed to setup cloud-based services and applications.
- This process uses a variety of virtualization software and industry standard hardware.

SDN computing platforms

SDN Computing platforms :

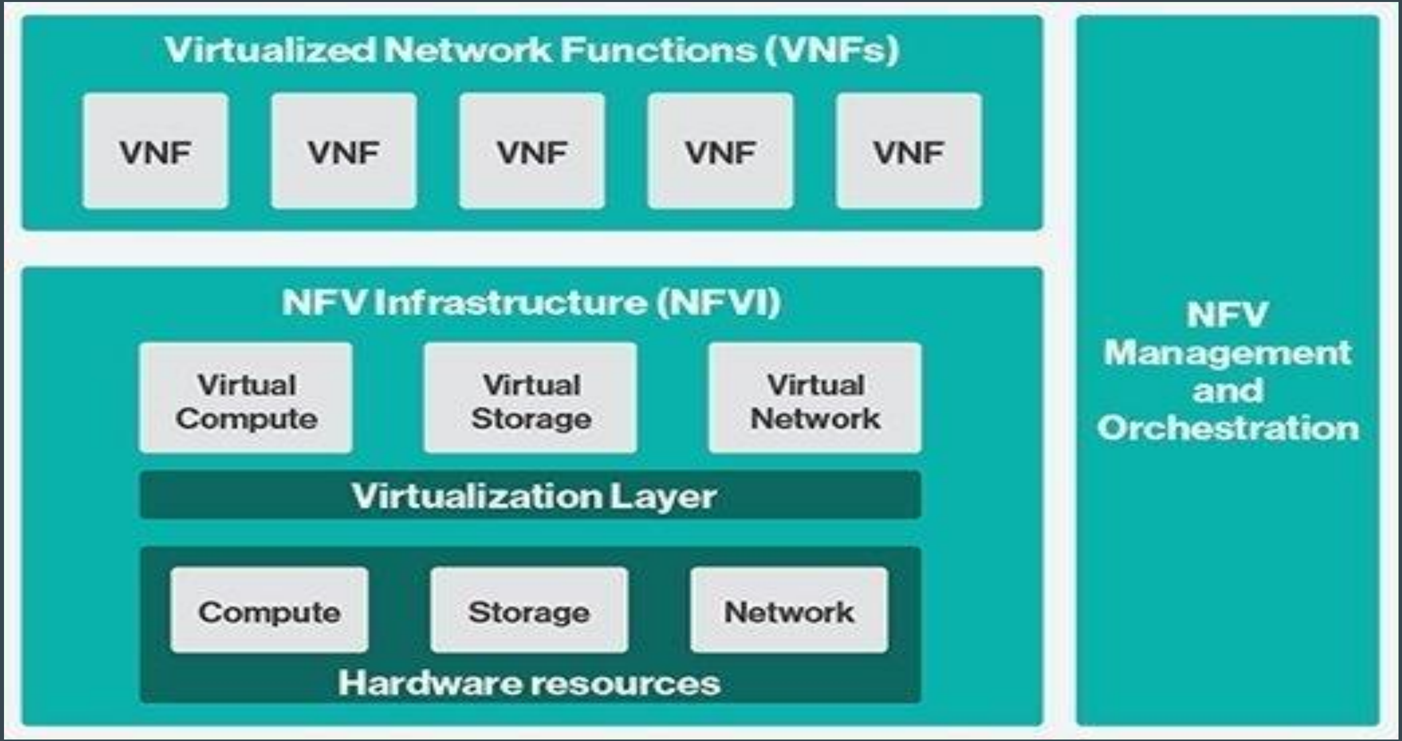
- Opendaylight
- Cisco ACI
- ONOS
- Project Floodlight
- VMWARE
- Beacon
- Juniper contrail
- vortiQa
- POX
- Nuage

VNF

Virtualized network functions :

- A VNF takes on the responsibility of handling specific network functions that run on one or more VM's on top of hardware networking infrastructure.
- VNFs can help increase network scalability and agility , while also enabling better use of network resources.

Virtualized network functions :

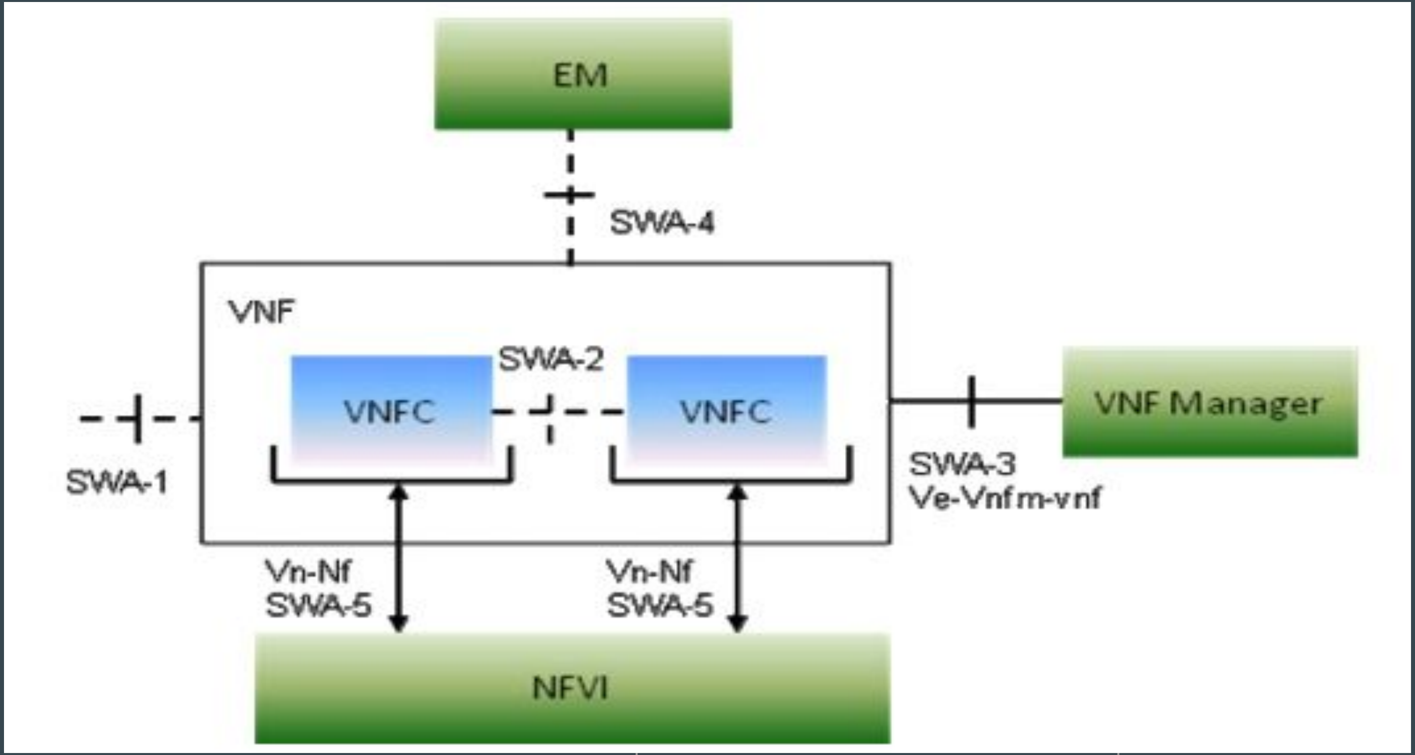


Virtualized network functions :

- Software implementation of the legacy network functions .
- Network function capable of running over NFVI.
- Network function orchestrator by NFVO and VNF manager.

VNF Architecture

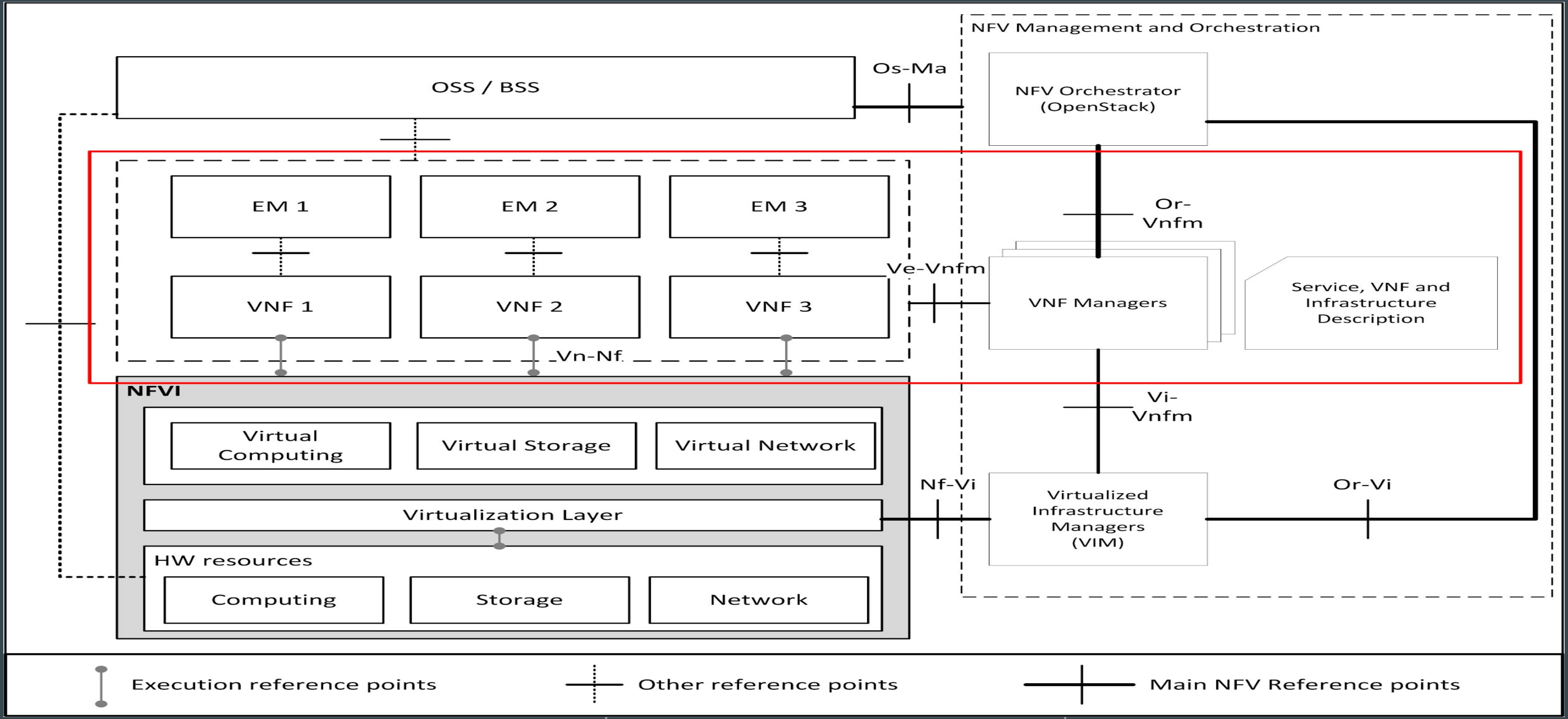
VNF Functional view :



VNF interfaces :

- SWA-1
- SWA-2
- SWA-3
- SWA-4
- SWA-5

VNF interfaces :

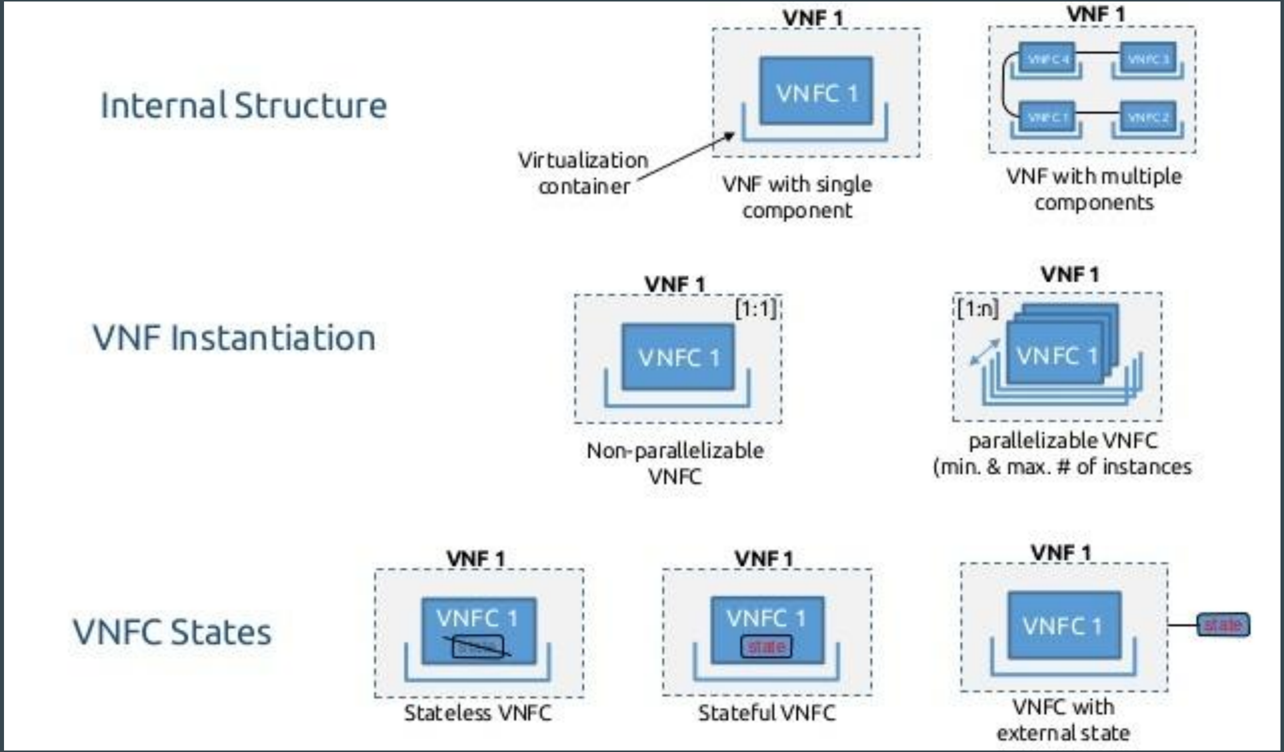


VNF - Design and properties :

It is divided as :

- Internal structure
- Life cycle
- VNFC states
- Load balancing

VNF - internal structure :



VNF - Instantiation :

There are two ways to achieve this :

- Parallelizable : instantiation multiple time per VNF instance but with constraints on the number.
- Non - parallelizable : instantiation once per VNF instance.

VNF - states :

Stateful VNFC

- VNFC that needs to handle state information of the VNF
- VNFC can be implemented stateless by sorting the state in the external repository to VNFC.

Stateless VNFC

- VNFC that does not need to handle the state information

VNF - load balancer :

VNF - internal load balancer

- 1 VNF instance seen as 1 logical NFV by peer NF
- VNF has at least one VNFC which can be replicated
- Internal load balancer VNFC which scatters/collects information/packets/flows/sessions.

VNF - load balancer :

VNF - external load balancer

- N VNF instances seen as 1 logical NFV by peer NF
- External load balancer which will be another VNF which scatters/collects information/packets/flows/session to/from the different VNF instances.

VNF - load balancer :

End-to-end load balancing

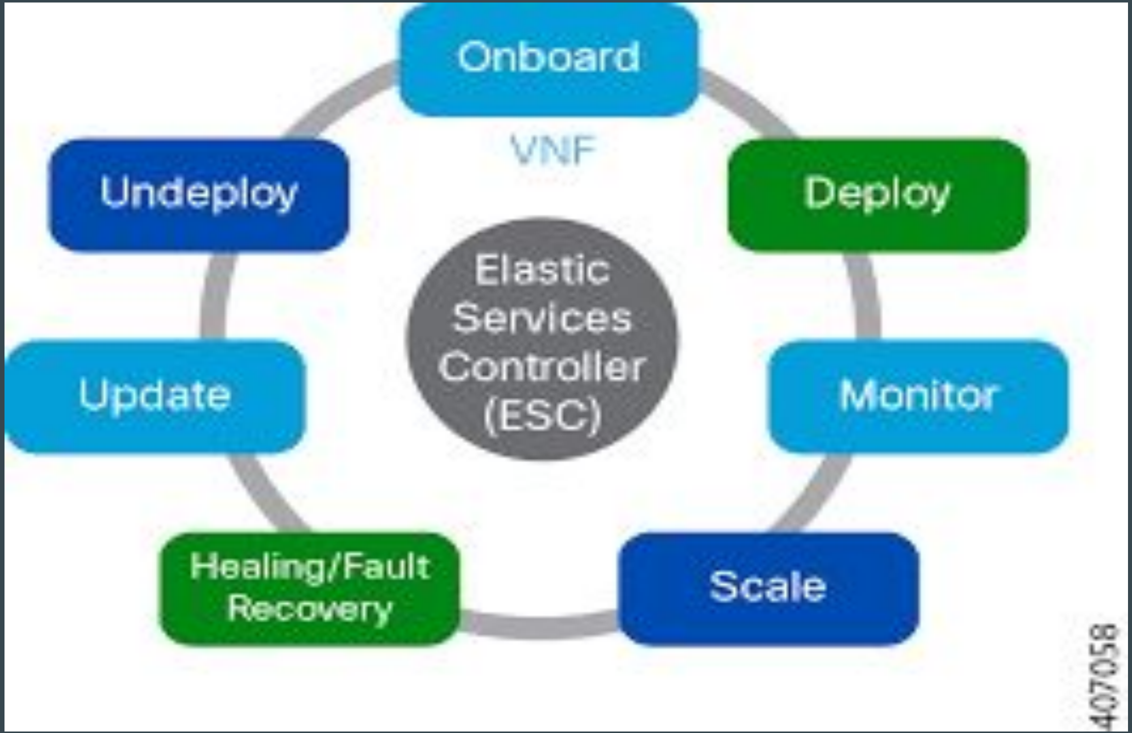
- N VNF instance seen as N logical NFV by peer NF
- Peer NF itself contains load balancing functionality

VNF - properties :

- Hardware independence
- Virtualization and container awareness
- Elasticity
- VNF policy management
- Migration operations

VNF lifecycle management

VNF lifecycle management :



VNF lifecycle management :

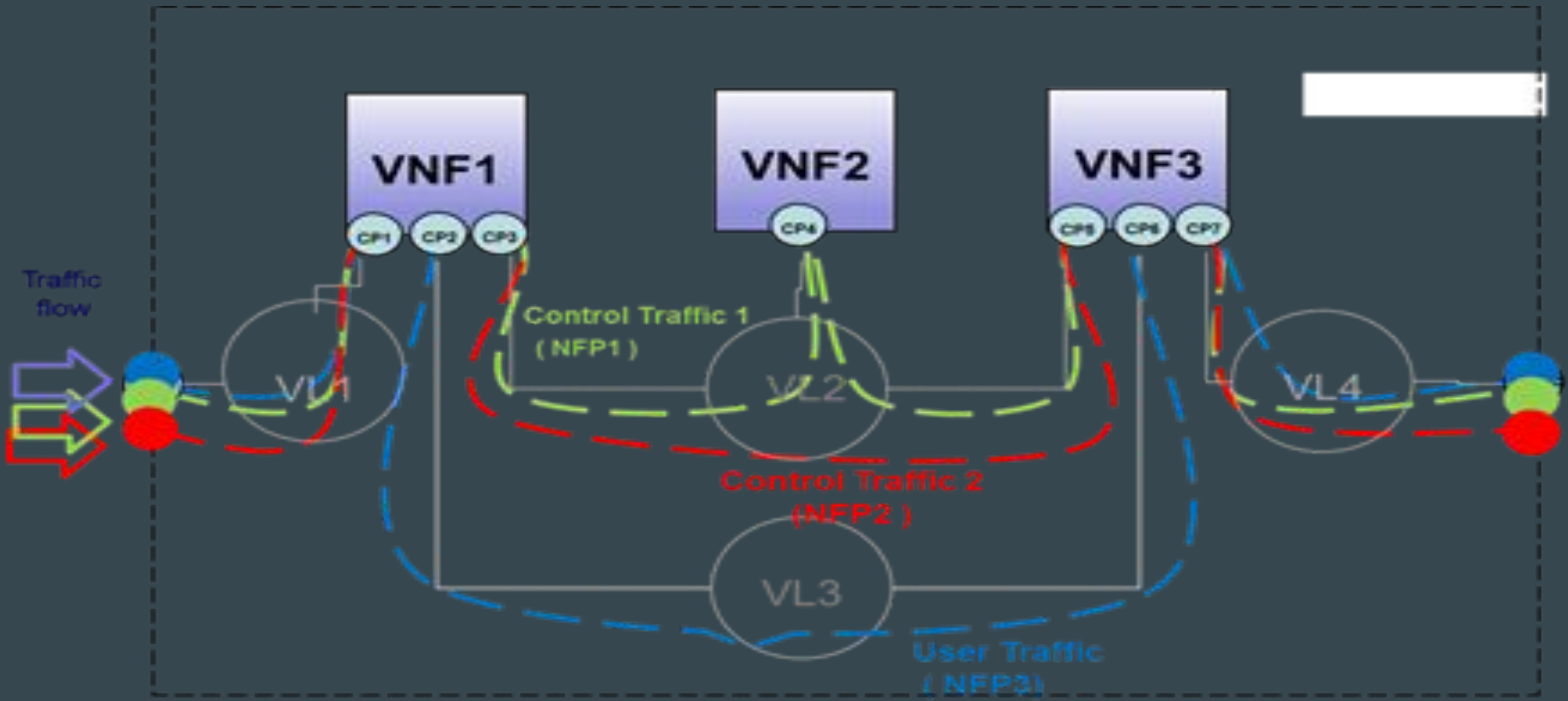
- Onboarding
- Deploying
- Monitoring
- Healing
- Updating
- Undeploy

VNF forwarding graph

VNF forwarding graph :

- It is used to orchestrate and manage traffic through VNFs.
- The VNF-FG shows the graph of logical links connecting VNF nodes for the purpose of describing the traffic flow between these VNFs.
- VNF-FG1 : for control traffic
- VNF-FG2 : for user traffic

VNF forwarding graph :

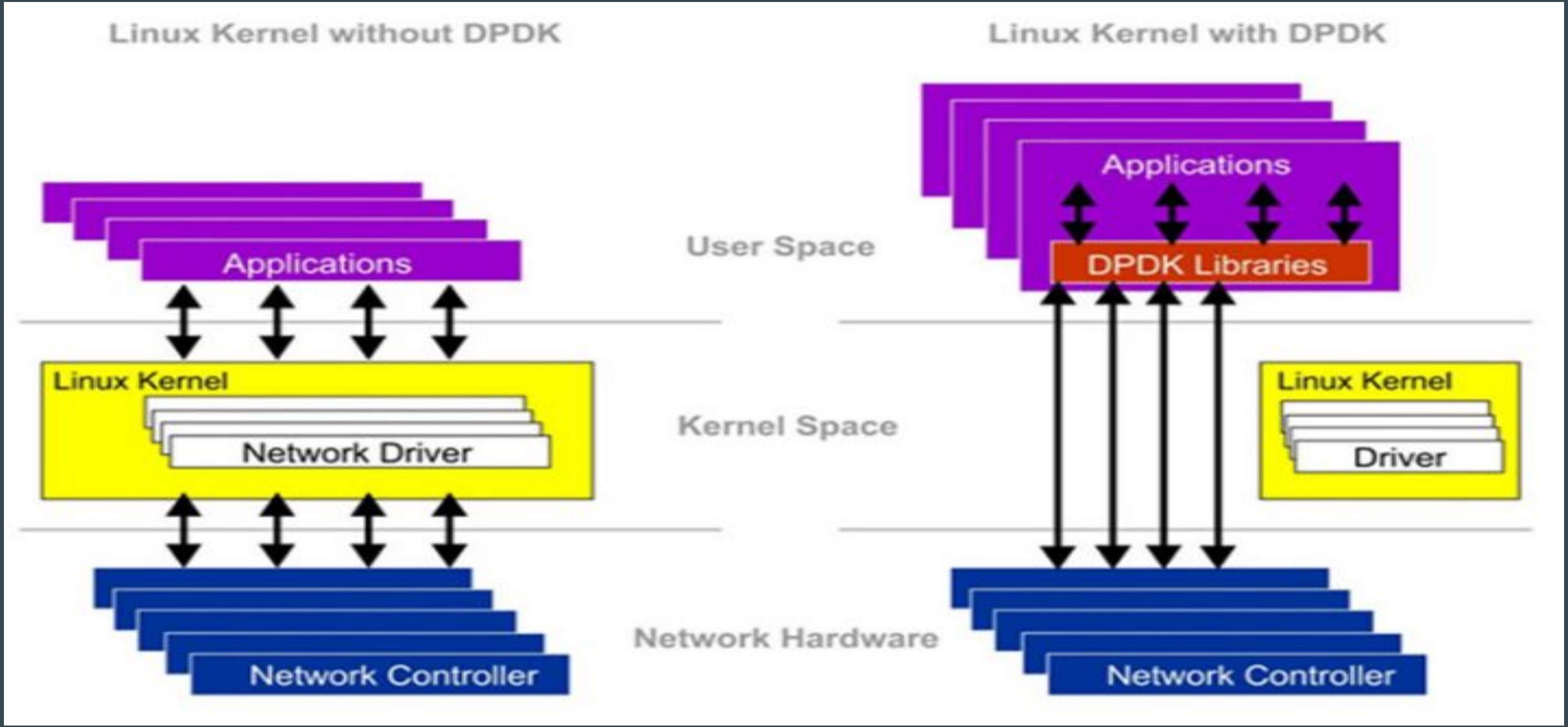


DPDK

DPDK :

- It is a set of data plane libraries and network interface controller drivers for fast packet processing.
- DPDK enables to build applications, that we can use to process packets faster.

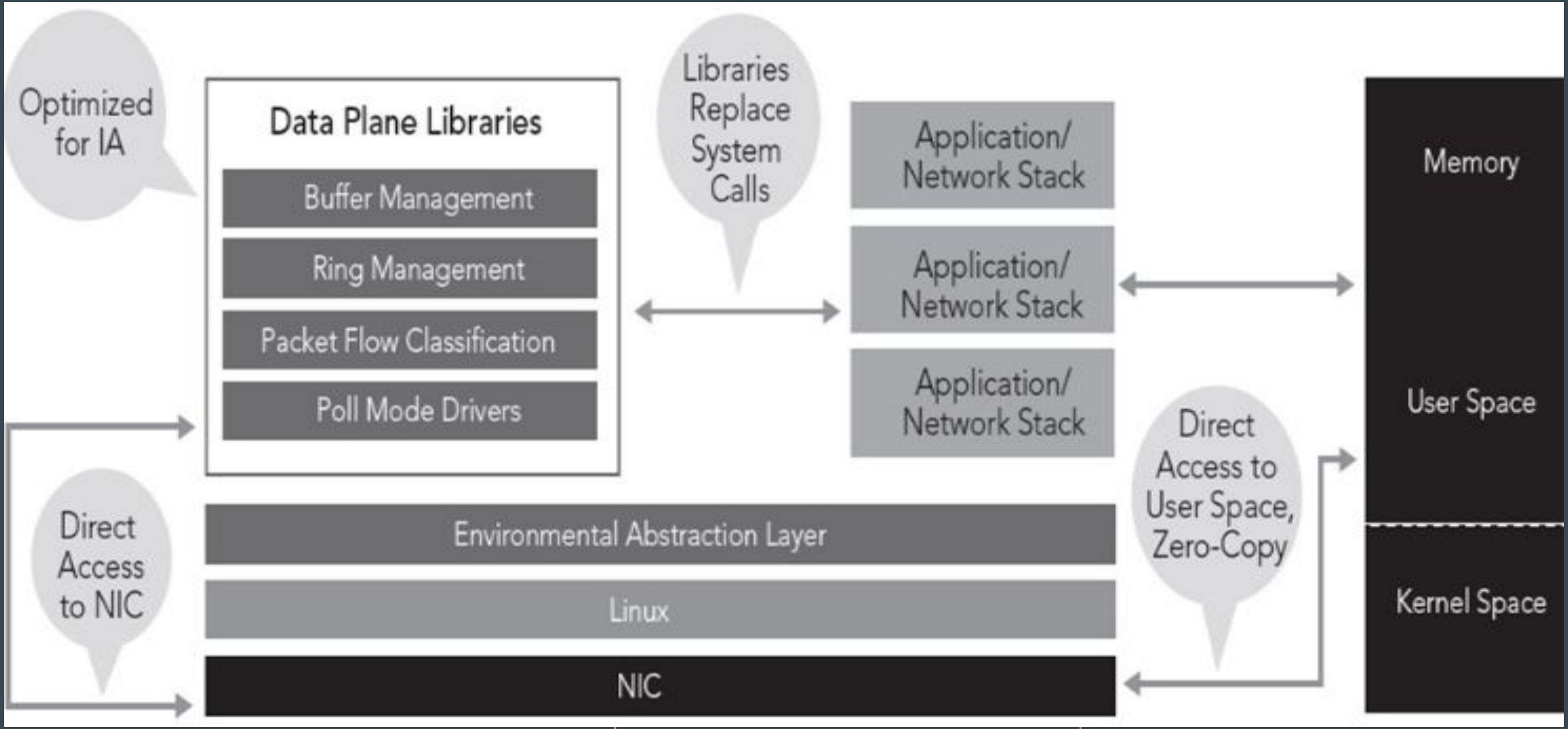
DPDK :


Linux Kernel without DPDK
Linux Kernel with DPDK
Applications
User Space
Applications
DPDK Libraries
Linux Kernel
Network Driver
Kernel Space
Linux Kernel
Driver
Network Hardware
Network Controller
Network Controller

DPDK components :

- EAL : Environment Abstraction Layer
- Memory manager
- Buffer manager
- Queue manager
- Packet flow classification
- Poll mode drivers

DPDK components :



Optimized for IA
Data Plane Libraries
Buffer Management
Ring Management
Packet Flow Classification
Poll Mode Drivers
Libraries Replace System Calls
Application/ Network Stack
Application/ Network Stack
Application/ Network Stack
Memory
User Space
Kernel Space
Direct Access to User Space, Zero-Copy
Direct Access to NIC
Environmental Abstraction Layer
Linux
NIC

SR-IOV

SR-IOV :

- It is a specification that allows a PCIe device to appear to be multiple separate physical PCIe devices.
- SR-IOV requires support in the BIOS as well as in the operating system instance or hypervisor that is running on the hardware.
- The SR-IOV offers different virtual functions to different virtual components (e.g network adapter) on a physical server machine.

SR-IOV :

- It uses physical functions (PF’s) and virtual functions (VF’s) to manage global for the SR-IOV devices.
- PFs are full PCIe functions that include the SR-IOV extended capability which is used to configure and manage the SR-IOV functionality.
- For SR-IOV enabled PCIe devices to function, you must have the appropriate BIOS and hardware support , as well as SR-IOV support in the guest driver or hypervisor instance.

SR-IOV :

- For SR-IOV PCIe devices to function, required to have appropriate BIOS and hardware support, as well as SR-IOV support in the guest driver or hypervisor instance.

Juniper contrail

Juniper Contrail :

- Juniper networks contrail is a open , standards - based software solution that delivers network virtualization and service automation for federated cloud networks.
- Using contrail, a tenant can define , manage and control the connectivity , services and security policies of the virtual network.

Juniper Contrail :

- Once created, policies can be applied across multiple network nodes , changed, added and deleted, all from a simple browser based interface.
- A browser-based user interface enables users to define virtual network and network service policies, then configure and interconnect networks simply by attaching policies.

Juniper Contrail :

- Contrail can be used with cloud orchestration systems such as openstack.
- It allows customers to build elastic architectures that leverage the benefits of cloud computing - agility, self-service, efficiency and flexibility.

Juniper contrail components

Major components :

- Contrail control nodes
- Contrail compute nodes - XMPP Agent & vRouter

Contrail containers :

- Contrail software releases are distributed as set of packages for each of the subsystem modules of a contrail system.
- Contrail subsystems are delivered as docker containers that group together related functional components.
- Each container file includes an INI based configuration file for configuring the device within the container.

Contrail system overview :

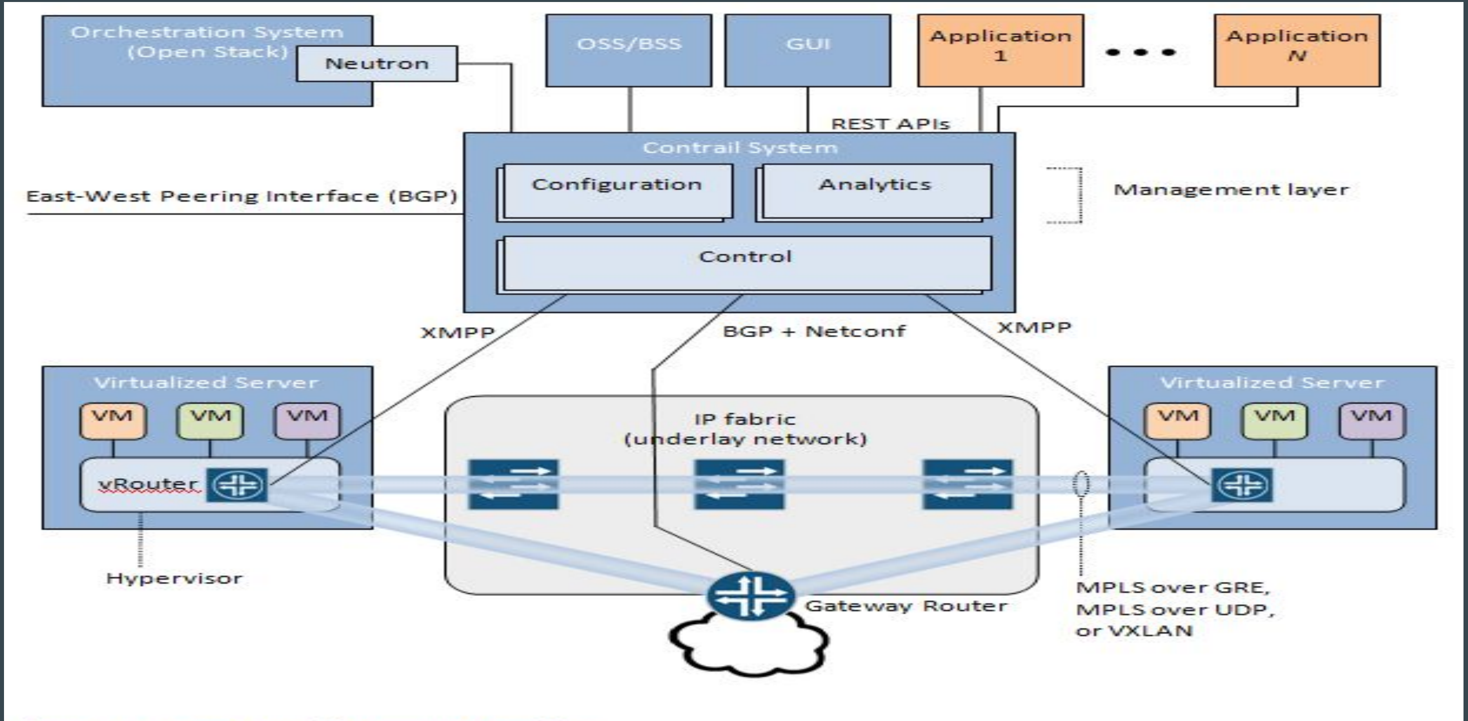


Figure 1: OpenContrail System Overview

Juniper contrail components

Nodes :

- Each node can be implemented as a separate physical server, or it can be implemented as a VM.
- All nodes of a given type run in an active - active configuration.

Nodes :

- Configuration node : it keeps a persistent copy of the intended configuration state and translate the high level data model suitable for interacting with network elements.
- Control node : it implement a logically centralized control plane that is responsible for maintaining ephemeral network state.
- Analytics node : it collect, store , correlate and analyze information from network elements , virtual or physical.

Nodes :

- Compute nodes are general purpose virtualized servers that host VMs.
- Gateway nodes are physical routers or switches that connect the virtual networks to physical networks.
- Service nodes are physical network elements providing network services such as WAN optimizers, load balancers.

Juniper contrail components

Contrail networking router :

- It runs on the compute node of the cloud or NFV infrastructure.
- The vRouters run in one of two high performance implementations : as a Linux kernel module or as an Intel data plane development kit (DPDK) based process.

Key features :

- Routing and bridging
- Load balancing
- Security & multitenancy
- Elastic, resilient VPN
- Gateway services
- High availability
- Analytics services
- API services

Key benefits :

- Simple
- Open
- High scale and performance
- United multi cloud policy
- Seamless integration

Use cases :

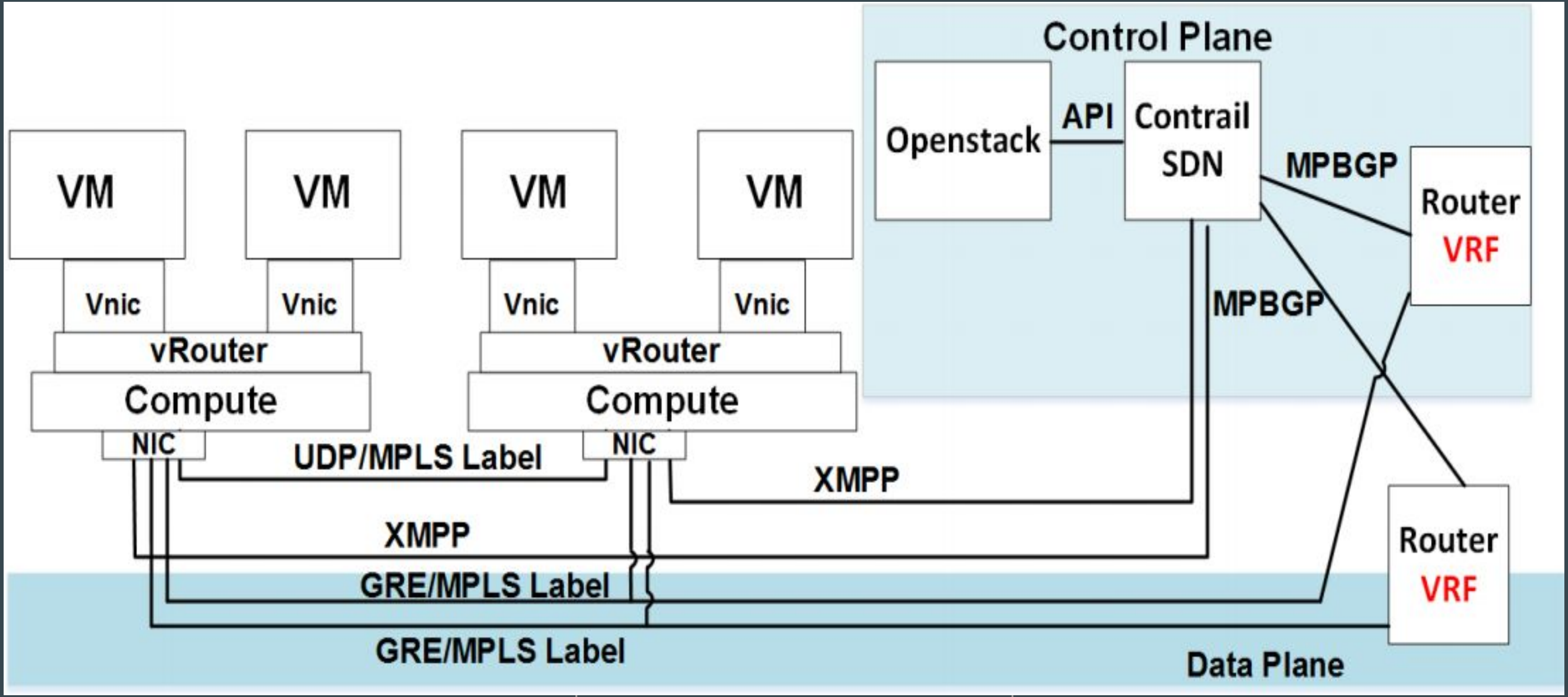
- Deploy private or public clouds
- Deploy hybrid clouds and create VPC in a service provider public cloud.
- Automate NFV through service chaining of any network and security service.

Use cases :

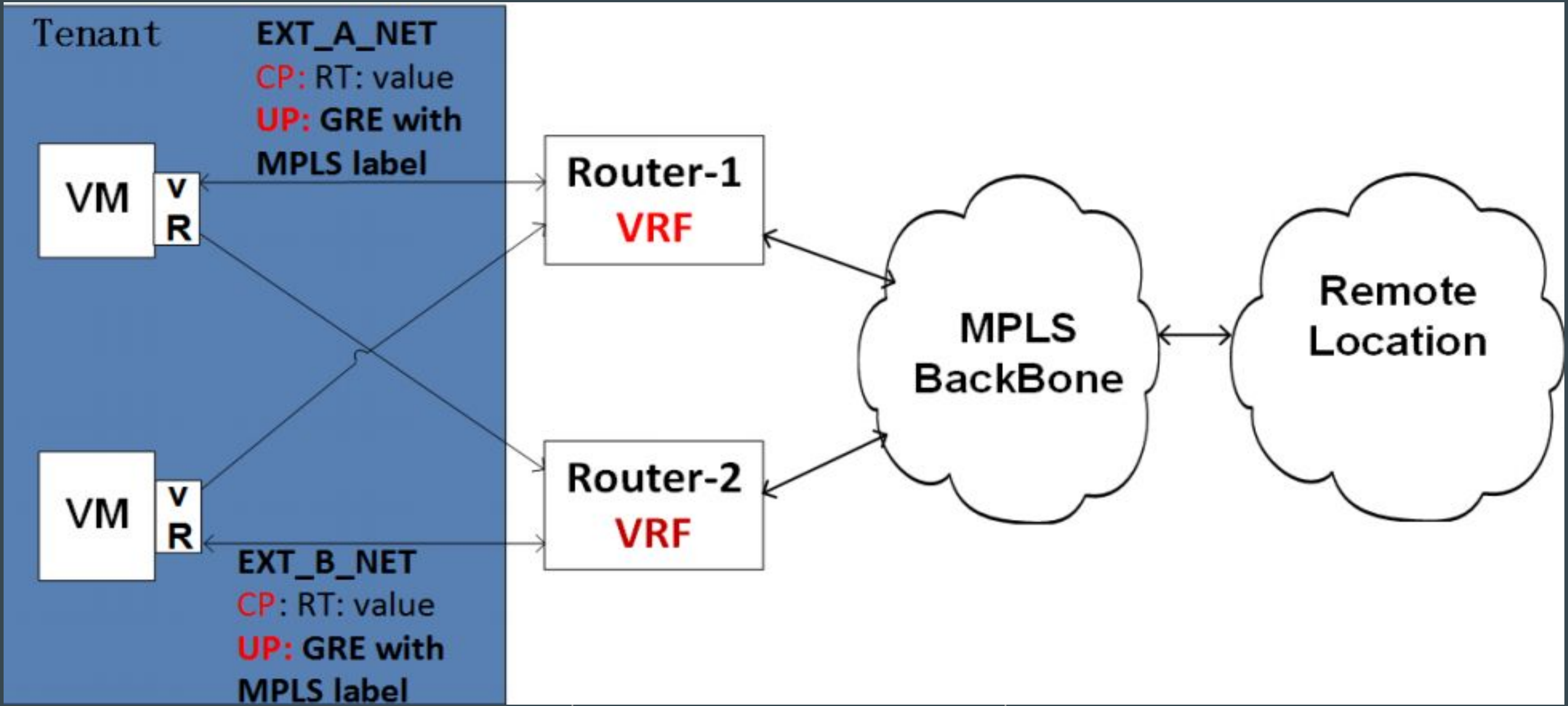
- Deploy private or public clouds
- Deploy hybrid clouds and create VPC in a service provider public cloud.
- Automate NFV through service chaining of any network and security service.

Juniper contrail

Contrail :

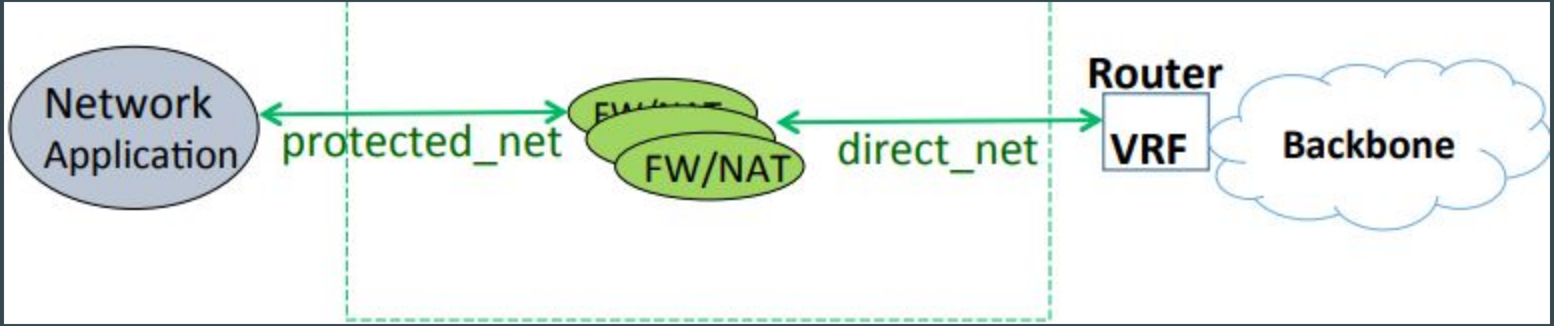

Control Plane
Openstack
API
Contrail SDN
MPBGP
Router
VRF
MPBGP
VM
VM
VM
VM
Vnic
Vnic
Vnic
Vnic
vRouter
vRouter
Compute
Compute
NIC
NIC
UDP/MPLS Label
XMPP
XMPP
GRE/MPLS Label
GRE/MPLS Label
Router
VRF
Data Plane

Connectivity to an external router :



Opencontrail features :

- Policy routing or service chaining
  With policy routing you can route traffic based on predefined policy rules.



Opencontrail features :

- Public IP addresses without NAT
- Turnking
- QoS marking and remarking
- Support of priority queues for QoS
- Jumbo frame support
- Full support of SCTP protocol
- High availability

Contrail releases

Opencontrail features :

- Contrail release 4.0.1 : openstack ocata, openstack Newton , Openstack Mitaka
- Contrail release 4.0 : openstack Newton , openstack Mitaka

Server requirements

Server Requirements :

Each server must have a minimum of :

- 64 GB memory
- 300 GB hard drive
- 4 CPU cores
- At least one ethernet port

Downloading installation software :

All components necessary for installing the contrail controller are available as :

- An RPM file
- A debain file

Downloading installation software :

The Contrail image includes the following software :

- All dependent software packages needed to support installation and operation of Openstack and Contrail.
- Contrail controller software - all components
- Openstack release currently in use for contrail

Installing the operating system :

- Install a centOS or Ubuntu minimal distribution as desired on all servers.
- Install Contrail server manager.
- Create an image .json with the ubuntu or centOS image to be used to reimage the target server.

Configuring the control node :

An important task after a successful installation is to configure the control node.

- The contrail controller base system image has been installed on all servers.
- The role based services have been assigned and provisioned.
- IP connectivity has been verified between all nodes of the contrail controller.
- You can access the Contrail user interface where IP address of the configuration node server that is running the service.
- Configure BGP

Support for multiple interfaces :

- Servers and nodes with multiple interfaces should be deployed with exclusive management control and data networks .
- In case of multiple interfaces per server, the expectation is that the management connectivity to the cluster , and the control and data network carries the control plane information and the guest traffic data.

Support for multiple interfaces :

Examples of control traffic include the following :

- XMPP traffic between the control nodes and the compute nodes.
- BGP protocol messages across the control nodes.
- Statistics, monitoring and health check data collected by analytics engine from different parts of the system.

Contrail global controller

Contrail Global controller :

- The global controller feature provides a seamless controller experience across multiple regions in a cloud environment by helping manage multiple Openstack installations , each having its own keystone , Neutron, Nova and so on.
- High availability is provided by using separate failure domains by regions.

Contrail Global controller :

- To handle the resource burdens when connecting and configuring servers and virtual machines over multiple, different regions, the global controller has responsibilities as :
  Resource identifier management
  Multiple location resource provisioning

Resource identifier management :

- The global controller uses centralized resource ID management to manage multiple types of identifiers , identifying such things as route targets , virtual networks , security groups and so on.
- The contrail global controller can interconnect virtual networks residing in different data centers using BGP VPN technology.

Resource identifier management :

- BGP VPN recognizes VPNs by using route target identifiers.
- A virtual network ID is used to identify the same virtual networks in different data centres, to prevent looping in service chains.

Multiple location resource provisioning :

- There are many cases in which the same resource , such as policy or services, needs to exist in multiple data centers.
- There might be security policy to apply a firewall for any traffic for an application server network that exists in multiple locations.
- Each location needs to have the same virtual network, network policy and firewalls.

Configuring Contrail

Configuring Contrail :

- Configuring virtual networks
- Deploying a multi-tier web application using contrail
- Configuring services
- Configuring service chaining

Configuring virtual networks

Configuring Contrail :

- Creating projects in openstack for configuring tenants in contrail.
- Creating a virtual network with Juniper networks contrail.
- Creating a virtual network with openstack contrail

Service chaining

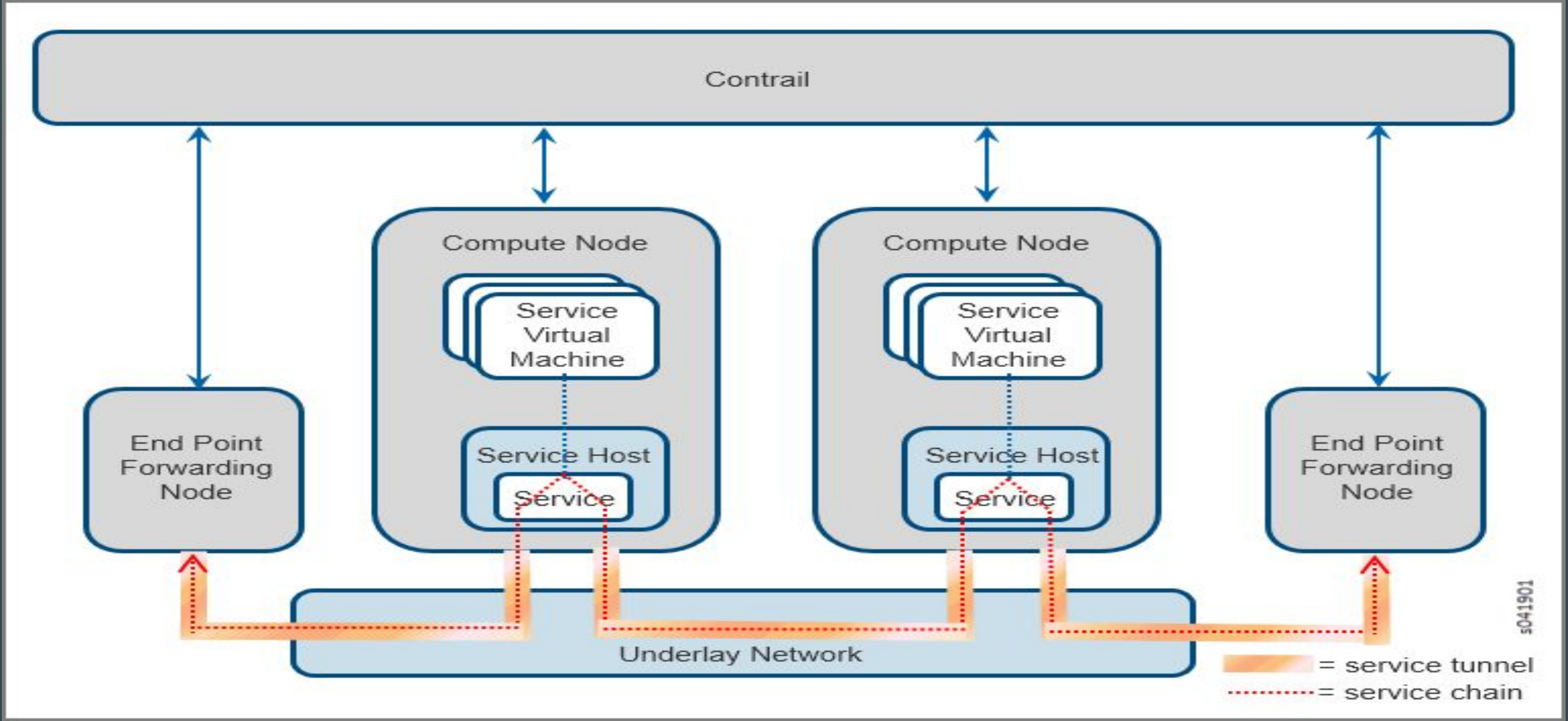
Basics :

- Services are offered by instantiating service virtual machines to dynamically apply single or multiple services to VM traffic.

Contrail service chain :

- Services are offered by instantiating service virtual machines to dynamically apply single or multiple services to VM traffic.

Contrail service chain :



Contrail service chain :

- Transparent or bridge mode
- In - network or routed mode
- In - network - nat mode

Overview of Contrail

Use cases

Use cases :

- Cloud networking : it enables private clouds for enterprises and service providers, IaaS and VPC for cloud service providers.
- NFV in service provider network : this provides VAS for service provider edge networks such as business edge networks, broadband subscriber management edge networks and mobile edge networks.

Contrail SDN controller & the vRouter :

- Contrail SDN controller : it is a logically centralized but physically distributed SDN controller that is responsible for providing the management, control and analytics functions of the virtualized network.
- Contrail vRouter : it is a forwarding plane that runs in the hypervisor of a virtualized server.

Virtual networks :

- Virtual networks are a key concept in the contrail system .
- VNs can be connected to and extended across physical MPLS L3 VPNs and ethernet VPNs using a data center edge router.
- Virtual networks are also used to implement NFV and service chaining.

Overlay networking :

- Virtual networks can be implemented using a variety of mechanisms. E.g each VPN can be implemented as a virtual LAN, VPN etc.
- VNs can also be implemented using two networks - a physical underlay network and a virtual overlay network.

Contrail and open source :

- The contrail system is integrated with open-source hypervisors such as kernel-based virtual machines(KVMs) and xen.
- The contrail system is integrated with open source virtualization orchestration systems such as openstack and cloudstack.
- The contrail system is integrated with open-source physical server management systems such as chef,puppet, cobbler and ganglia.

Graphical user interface :

- The contrail system also provides a GUI.
- This GUI is built entirely using the REST APIs .

Contrail Architecture details :

- Contrail SDN controller provides northbound REST APIs used by applications.
- These APIs are used for integration with the cloud orchestration system - e.g for integration with openstack via a neutron plugin.
- The contrail system provides three interfaces.

Contrail nodes

Contrail SDN controller components :

- Configuration nodes
- Control nodes
- Analytics nodes

Compute node :

- These are tenant VMs running customer applications such as web servers, database servers, enterprise applications or hosting virtualized services used to create service chains.
- The contrail vRouter forwarding plane sits in the linux kernel and the vRouter agent is the local control plane.

Compute node :

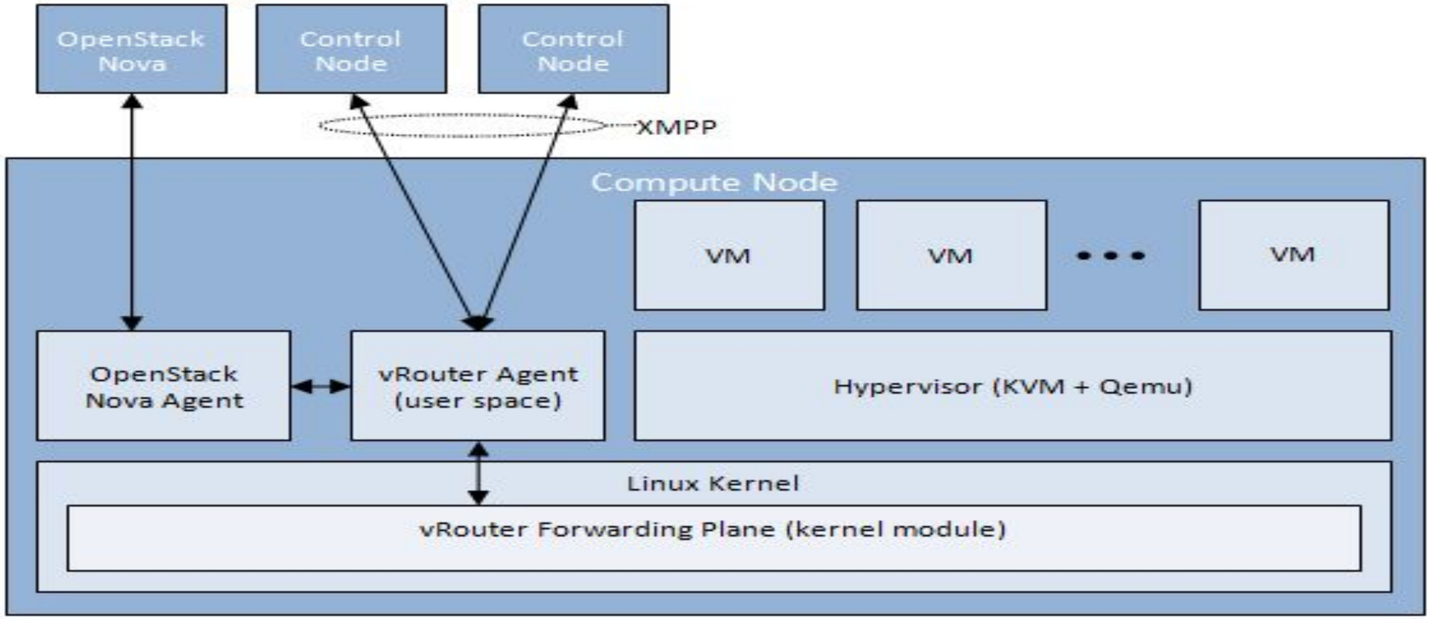


Figure 3: Internal Structure of a Compute node

vRouter Agent :

- It is a user space process running inside Linux.
- It acts as the local, lightweight control plane and is responsible for various functions.

Control node :

The control nodes communicate with multiple other types of nodes
:

- They receive configuration state from the configuration nodes.
- They exchange routes with other control nodes using IBGP to ensure that all control nodes have the same network state.
- They exchange routes with the vRouter agents on the compute nodes using XMPP.

Control node :

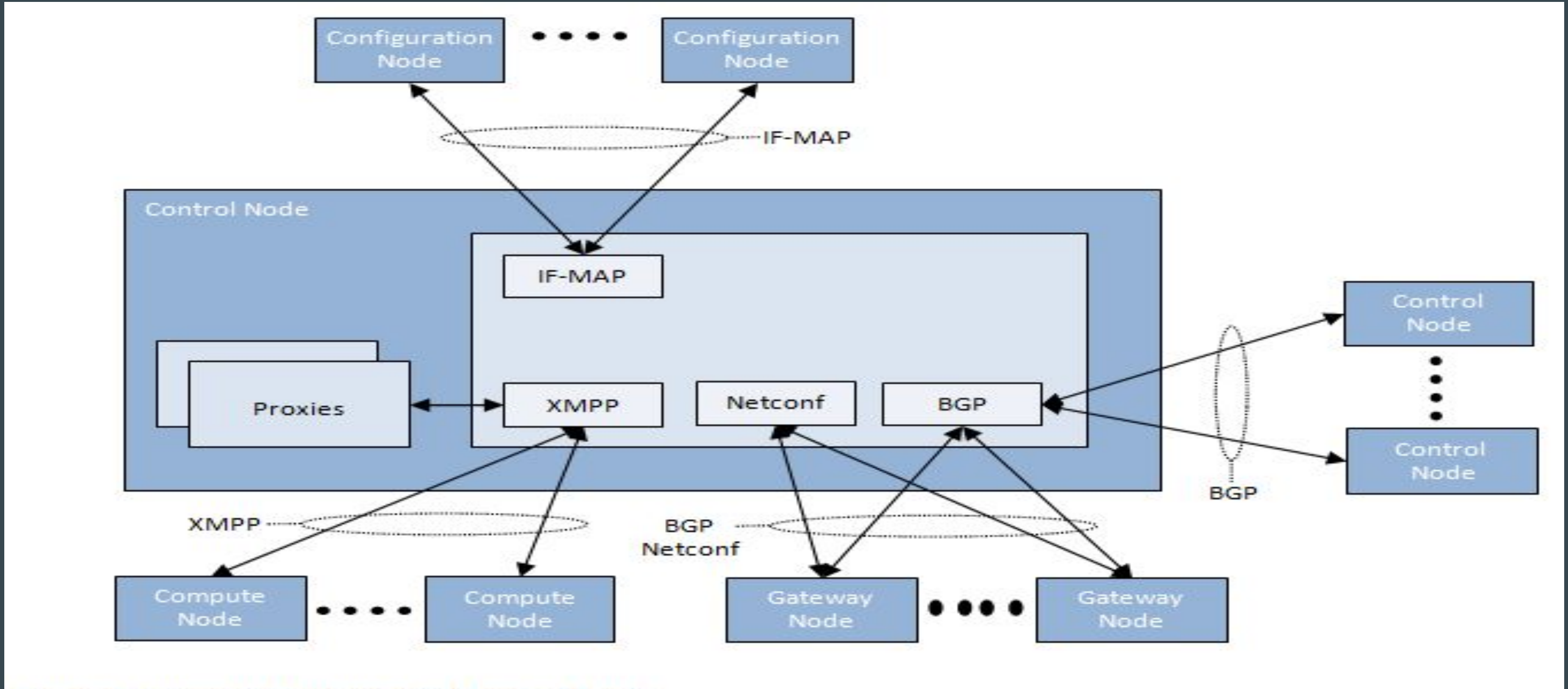


Figure 5: Internal Structure of a Control Node

Configuration node :

- It communicates with the orchestration system via REST interface.
- It provide a discovery service that the clients can use to locate the service providers.

Configuration node :

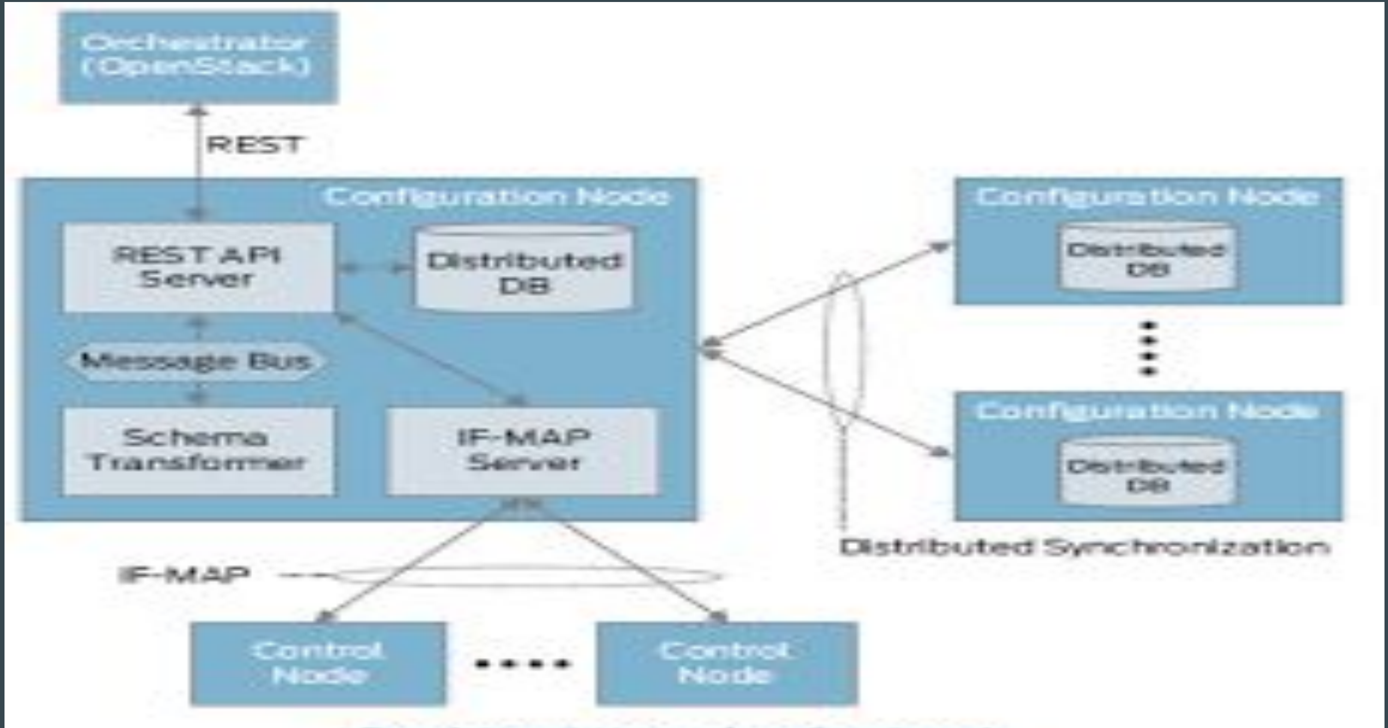
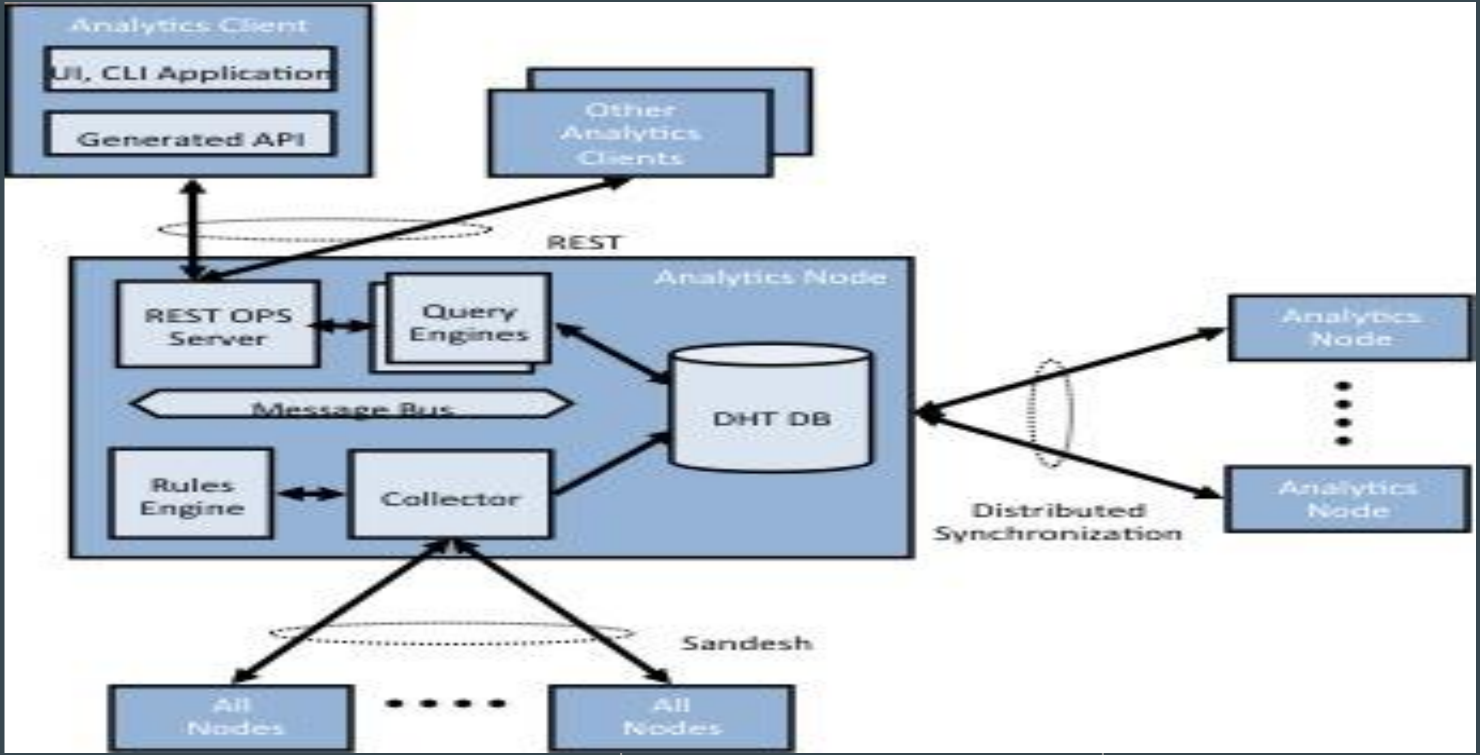


Figure 6: Internal structure of a configuration node

Analytics node :

- An analytics node communicates with applications using a northbound REST API.
- A collector exchanges sandesh messages with components in control nodes and configuration nodes to collect analytics information.
- A NoSQL database stores this information.

Analytics node :



Control & management plane protocols

IF-MAP :

- It is an open standard/server protocol developed by TCG.
- Original application of IF-MAP was to provide a common interface between MAPs , a database server acting a clearing house for info about security events and objects, and other elements of the TNC architecture.

XMPP :

- The extensible messaging and presence protocol is a communication protocol for message oriented middleware based on XML.
- Contrail uses XMPP as a general purpose message bus between the compute nodes and the control node to exchange multiple types of information including routes, configuration, operational state, statistics , logs and events.

BGP :

- Contrail uses BGP to exchange routing information among the control nodes.
- BGP can also also be used to exchange routing information between the control nodes and the gateway nodes.

Sandesh :

- It is an XML based protocol for reporting analytics information.
- The structure of the XML messages is described in schemas.

Role of orchestration in the data center

Role of orchestration :

In the data center, the orchestrator manages many critical aspects of the data center :

- Compute
- Storage
- Network
- Applications

Role of orchestration :

The SDN controller’s role is to orchestrate the network and networking services like load balancing and security based on the application it has assigned compute and storage resources.

- Create a virtual network for a tenant within a data center or across data centers.
- Attach a VM to a tenants virtual network.
- Connect a tenant’s virtual network to some external network e.g internet or VPN.

Role of orchestration :

- Physical switches
- Physical routers
- Physical service nodes
- Virtual services

Contrail security

Contrail security :

- It discovers application traffic flows and drastically reduces policy proliferation across different environments.
- With contrail security, you define policies once and automatically distributes them uniformly across all deployments.
- A v Router acts as a distributed element.

Contrail security advantages :

- Consistent, intent driven policy
- Multiple enforcement points
- Application traffic visibility & advanced analytics
- Unified operations and management

Contrail SD-WAN

Contrail SD-WAN :

- It delivers a simple , automated multi-cloud SD-WAN.
- Contrail SD-WAN uses contrail service orchestration to design, secure, automate and run the entire service lifecycle across service platforms, routers, gateways along with virtual firewall secure cloud endpoints.