HOW TO BUILD AGILE CLOUD SYSTEMS: CHALLENGES AND ENABLING TECHNOLOGIES

Pelin Angin
Department of Computer Science, Purdue University

June 27, 2016
San Francisco, CA

OUTLINE

- PART 1:
  - ❑ Cloud Agility Challenges
  - ❑ Containers and Microservices
  - ❑ Software-Defined Networking (SDN)
- PART 2:
  - ❑ Network Functions Virtualization (NFV)
  - ❑ NFV Use Cases & Projects
  - ❑ Future R&D Problems
- PART 3:
  - ❑ OpenStack Overview
  - ❑ OpenStack Hands-on lab

PART I

BUSINESS VALUE OF CLOUD COMPUTING

- Used to be defined in terms of Capital Expenses (CapEx) vs. Operational Expenses (OpEx)
  - Ability to avoid buying hardware and software
- Now: Agility
  - Harder to measure than CAPEX vs. OPEX
- Why the change?
  - CapEx vs. OpEx view not compelling enough from a cost standpoint
  - Better understanding of value metrics in agility
  - Better definition of worth of agility for internal business cases
  - Shorter time to market valuable for specific industries

Source: http://www.infoworld.com/article/2913273/cloud-computing/agility-is-king-for-cloud-value-but-roi-is-harder-to-measure.html

WHAT IS (CLOUD) AGILITY?

agile in Oxford Dictionary:

"Able to move quickly and easily"

Cloud Agility:

- Rapid provisioning of computer resources
  - New compute instances, networking and storage provided in minutes instead of weeks taken by IT
- Rapid business response to changing conditions
  - Shortened time-to-market

CLOUD AGILITY CHALLENGES

- Need less resource consumption than VMs
- Need lightweight, portable application components for quick deployment
- Geographically-distributed mission-critical applications may suffer from limitations in network quality of service (QoS)
  - Limited responsiveness to changes
- Provisioning networks to deliver optimal paths between new applications and users can take hours or days
  - Network configuration highly manual, device-by-device in data center network
- Enforcement of policies in network involves complex architecture
- Enterprises have limited control over cloud operations

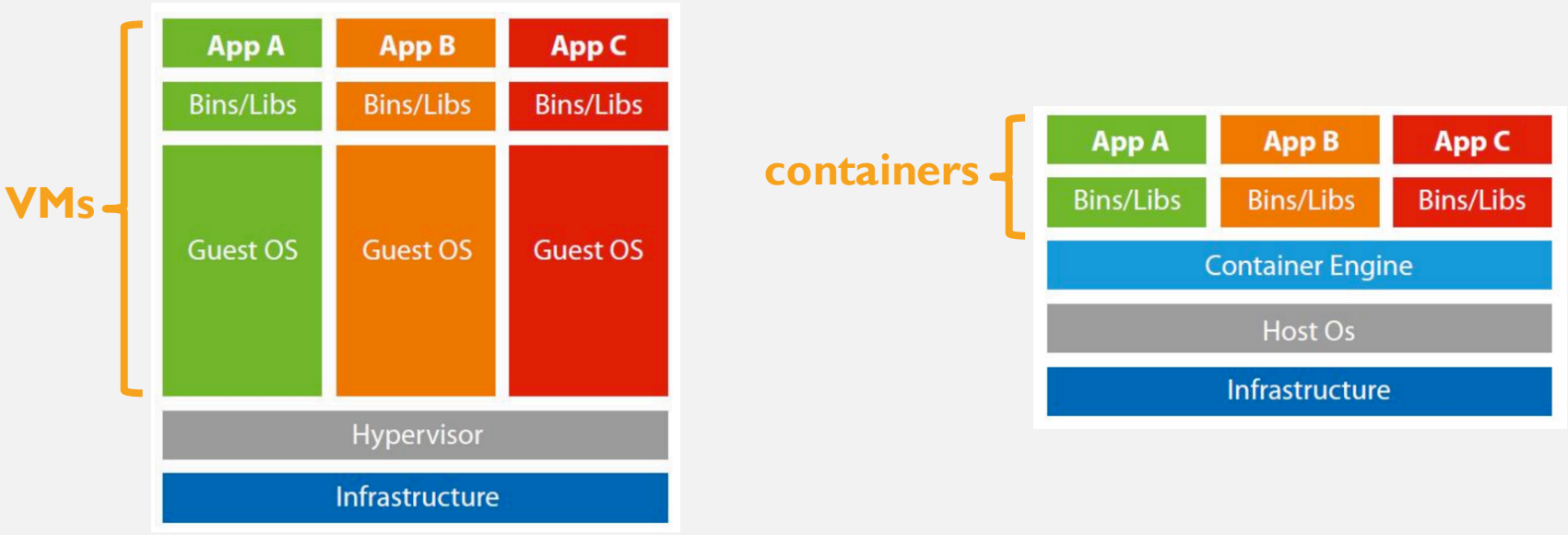
COMPONENTS OF CLOUD AGILITY

Enabling Technology	Agility Component
Containers	Rapid service deployment
	Lightweight, high-performance components
	Easy service migration
	High service uptime / Rapid update integration
Microservices	Flexible service composition
	Interoperability
SDN & NFV	Rapid network resource provisioning
	Easy network policy definition / enforcement
	Rapid network maintenance
	Quick network failure recovery

WHAT IS A CONTAINER?

- A lightweight structure in which an application can be stored, carried and run on a compatible OS
- Encapsulates application logic components provisioned with minimal resources (its own operating environment) required
- No need for embedded operating system
- Can be placed on any host without special configuration
- Virtualization at OS level rather than hypervisor level
- Required components:
  - Compatible OS
  - Runtime compatible with containers to run
  - Platform to manage applications
  - Tools for container orchestration, management, networking and security

CONTAINERS VS. VIRTUAL MACHINES (VM)



Adapted from https://www.serverspace.co.uk/blog/containerisation-vs-virtualisation-whats-the-difference

CONTAINER ADVANTAGES OVER VMS

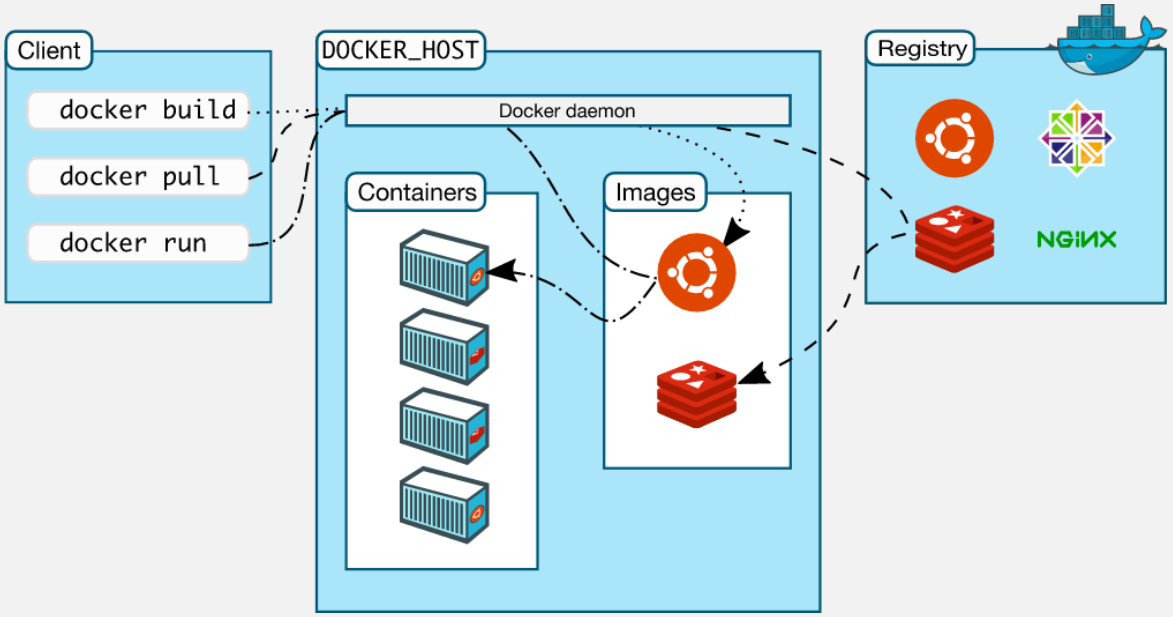
Both technologies based on virtualization, OS virtualization rather than hardware virtualization provides:

- Improved performance
  - Application instructions do not go through guest OS and hypervisor to reach CPU
- Reduced application size
  - Enables faster application startup
- Increased stability
  - Applications do not hang on the host OS
- Reduced footprint

EVOLUTION OF CONTAINERS

- Roots in Linux world, based on features of Linux kernel
- Early containers: FreeBSD Jails, Oracle Solaris Zones
- Google got involved beginning 2006:
  - Entire data-center architecture based on containers
  - Kubernetes for container / container cluster management
- Docker increased traction in container world
  - Started as an open source project to build containerized microservices in 2013, followed by company
- LXC (Linux Containers) rose as major competitor with Docker

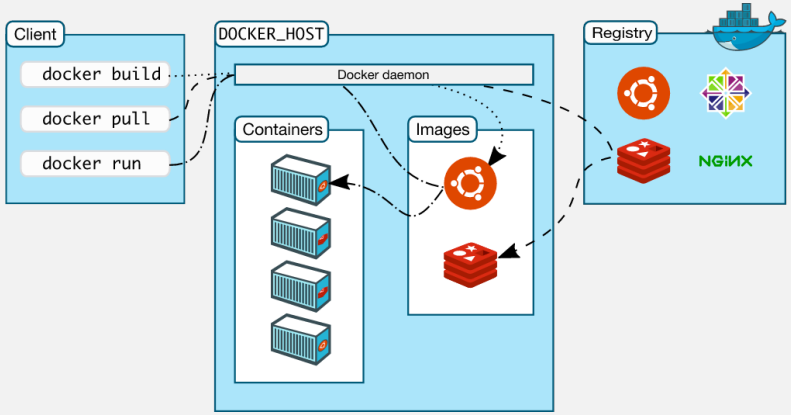
DOCKER



- Client-server architecture
- Docker daemon:
  - ❖ Building, running, distributing docker containers
- Docker client:
  - ❖ Binary user interface to Docker
  - ❖ Accepts commands from user, communicates with daemon
- Docker images (build):
  - ❖ Read-only template
- Docker registries (distribution):
  - ❖ Public or private stores for uploading/downloading images
- Docker containers (run):
  - ❖ Hold everything need to run app, created from Docker image

Source: https://docs.docker.com/v1.8/introduction/understanding-docker/

HOW DOCKER WORKS



Docker images:

- Consist of layers combined using union file systems to form single coherent file system
- Changing image builds new layer rather than replacing whole image, making distribution faster by only sending updates
- Images built from bases (e,g, Ubuntu image) using instructions

Docker container:

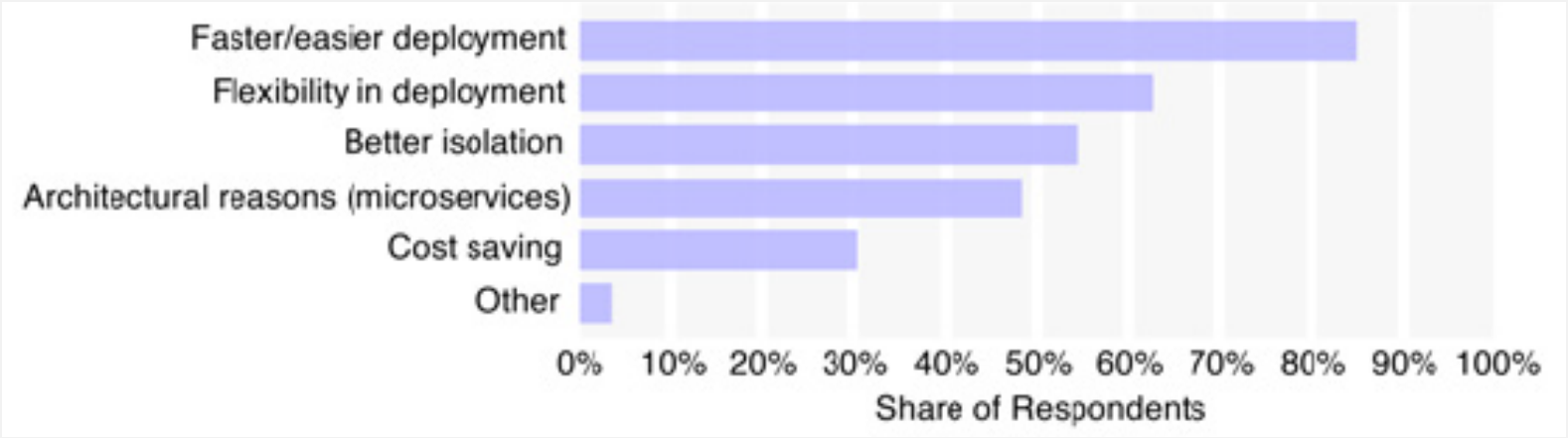
- OS + user-added files + meta-data
- When running, adds read-write layer on top of image in which app can run
- Client tells daemon to run container:
  - Image is pulled
  - New container created
  - Filesystem allocated, read-write layer added
  - Network interface allocated, IP address setup
  - App run, output provided

Source: https://docs.docker.com/v1.8/introduction/understanding-docker/

HOW DO CONTAINERS PROVIDE AGILITY?

- Containers can share the same OS
  - Less resource consumption
  - Increased efficiency
  - Fast application upload
  - No downtime during application updates
- Containers allow applications to be separated from underlying infrastructure
  - Increased portability
  - Potential security benefits

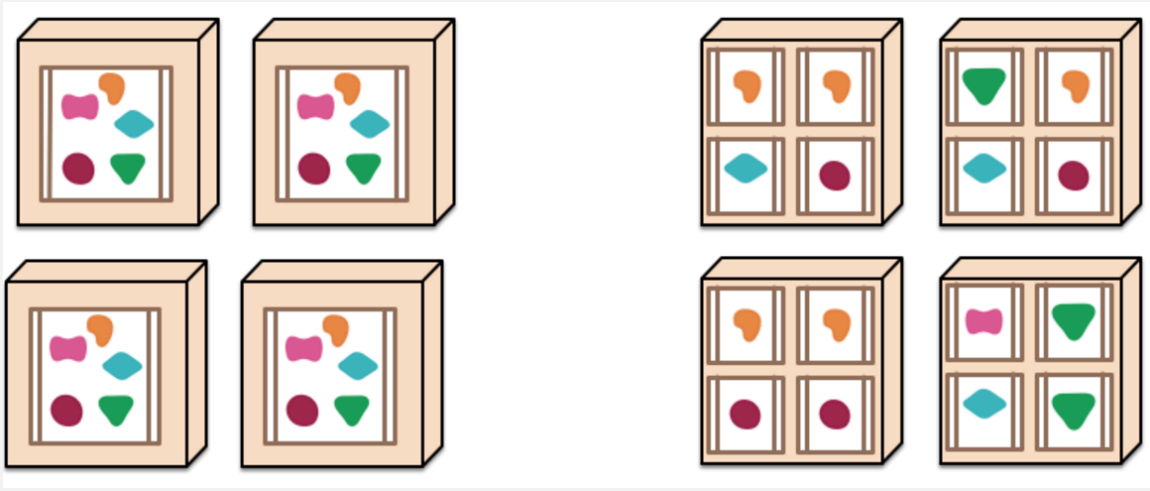
DRIVERS FOR CONTAINER ADOPTION IN ENTERPRISES



- O'reilly survey in 2015
- 138 respondents from a range of industries
- Half from companies of fewer than 500 employees
- 13% from companies of over 10000 employees

Source: "The State of Containers and the Docker Ecosystem: 2015"
http://www.dynatrace.com/en/ruxit/docker-report

MICROSERVICES



Monolithic applications | Microservices

- Microservices style: Developing applications as a set of small services, each running in its own process, communicating with lightweight mechanisms
- Evolved from Web services (service-oriented architecture)
- Scales by distributing services across servers, instead of distributing apps

Source: http://martinfowler.com/articles/microservices.html

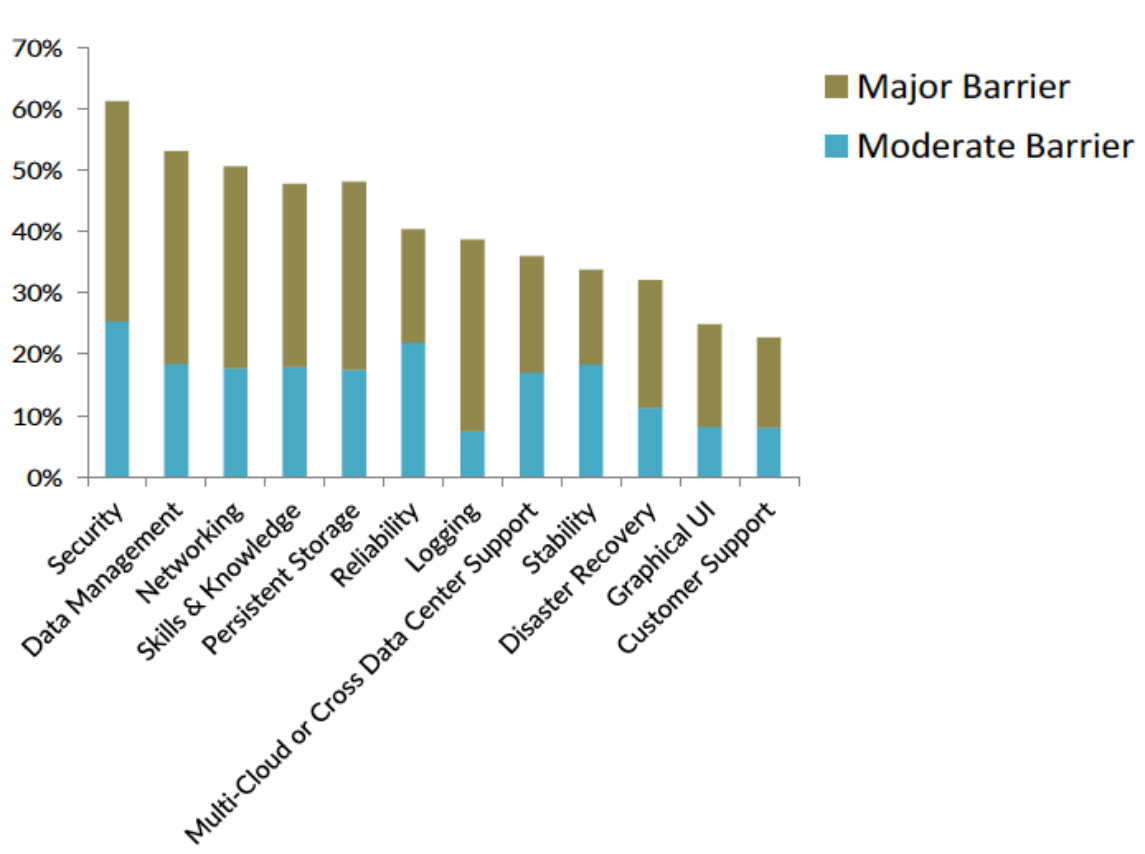
WHY MICROSERVICES?

- Independently deployable
  - Single service changes require redeployment of only that service
- More explicit component interfaces
- Decreased coupling between components
- Continuous delivery
- Resilience against failures by monitoring

INCREASED AGILITY

Used by Amazon, Netflix, Uber, Ebay, Karma, Sound Cloud, Groupon, Hailo, Gilt, Zalando, Capital One, Comcast…

MAJOR BARRIERS FOR CONTAINER ADOPTION



- DevOps.com & ClusterHQ survey
- Conducted in May 2015
- 285 respondents from organizations of mostly 1-500 employees, 18% with over 2500 employees

Source: https://clusterhq.com/assets/pdfs/state-of-container-usage-june-2015.pdf

FUTURE OF CONTAINERS

- Open Container Initiative (OCI) announced by Docker in 2015 to create open standard for container runtimes supporting technology based on Docker's container format
- OCI members: Amazon, Google, Mesosphere, Pivotal, Cisco, IBM, Microsoft, Intel, RedHat, Oracle, Verizon etc.
- Urs Hölze (senior vice president of technical infrastructure at Google at Interop keynote, 2015):

  "We have to go with containers. We need to think about applications instead of machines. The system manages placement on machines. You don't have to think about OS security patches, and configuration. A whole class of administrative tasks is removed."
- David Aronchick (lead product manager for Kubernetes):

  "When you talk to any large customers, their problems come down to two things – they want to move faster and they want to do it in a more cost effective way. Containers enable both of these."

FUTURE OF CONTAINER R&D

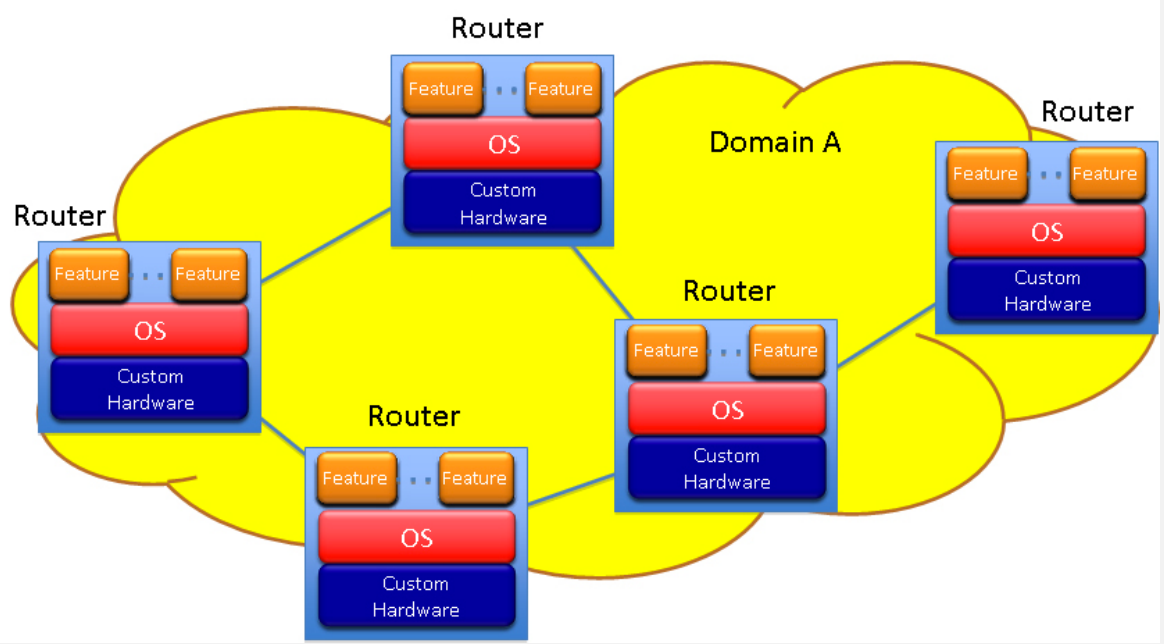
- Orchestration and Management
  - Various companies building new tools
  - Other functions can be integrated for monitoring, intelligence, providing catalog of container apps
- Networking
  - Containers need to talk to each other across networks
  - Current infrastructure may cause barriers to communications with firewalls
  - Need to adapt networking technology to handle containers
  - Already started with network virtualization
  - What is the most efficient way to connect? Should containers do their own routing? Should there be an independent controller?

FUTURE OF CONTAINER R&D (CONT.)

- Storage
  - How should containers connect to storage resources?
  - First containers stateless, but need state for data-heavy applications
  - Some solutions (e.g. ClusterHQ, Portworx) can connect containers to storage devices and storage area networks (SANs)
- Security
  - Containers not secure by default!
  - Many container elements rely on a shared kernel, exposing OS
  - Banyan investigation discovered 30% of public container images with vulnerabilities*
  - Need better isolation, better visibility, security screening

*"Over 30% of Official Images in Docker Hub Contain High Priority Security Vulnerabilities." Banyan. http://www.banyanops.com/blog/analyzing-docker-hub

HOW DO LEGACY NETWORKS OPERATE?



- Decision-making capability (routing, managing, monitoring, load balancing etc.) distributed across network devices
- Data plane and control plane tightly coupled

Source: http://www.netvolution.eu/scientific-approach.html

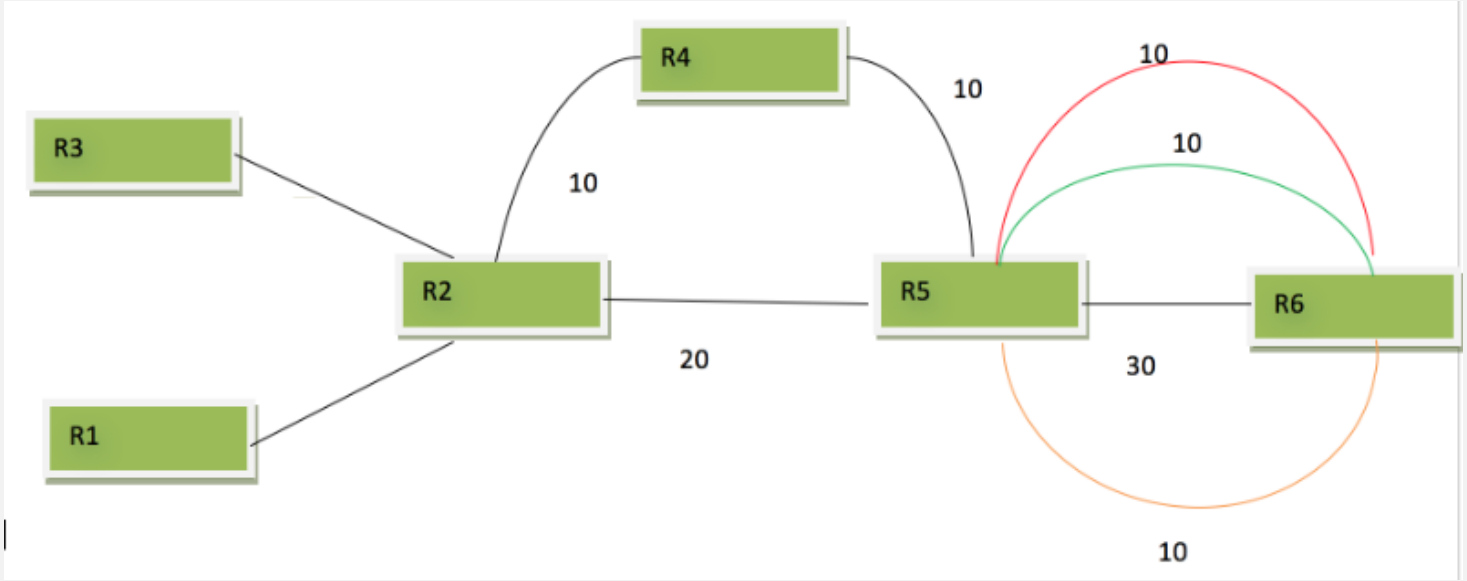
NETWORKING PLANES

- Data Plane: Activities related to data packets sent by end-user
  - Forwarding
  - Fragmentation/Reassembly
  - Replication for multicasting
- Control Plane: Activities not involving end-user data packets
  - Routing tables
  - Packet handling policies
  - Announcing service availability
- Management Plane: Provisioning & monitoring of networks
- Services Plane: Improvement of performance or security

PROBLEMS WITH LEGACY NETWORKS

- Inflexibility
  - Not fit for constantly shifting network traffic patterns
  - Adding/removing devices is complex
- Slow convergence after link failures
- High cost to update switches
- Vendor dependence
- Inability to scale
- Complexity of access control
- Manual configuration needs for supporting high-level network policies

LEGACY NETWORK EXAMPLE



If R5-R6 link fails:

- Takes time to reprogram all devices to for new network
- Sometimes network not able to find new state

https://www.linkedin.com/pulse/network-manager-running-external-controller-sdn-rakesh-raushan

NETWORK VIRTUALIZATION

- Independent of Physical Network Location or State
  - Logical network across any server, rack, cluster, data-center
  - VMs can migrate without requiring reworking of security policies, load balancing etc.
  - New workloads or networks do not require provisioning of physical network
  - Node failures in physical network **do not** disrupt workload
- Isolation for Multi-tenancy and Fault Tolerance
  - MAC and IPs private per tenant
  - Failures and configuration errors by tenants **do not** affect other tenants
  - Failures in virtual layer **do not** propagate to physical layer

Adapted from http://www.slideshare.net/priti_desai/secure-multi-tenant-cloud-with-opencontrail

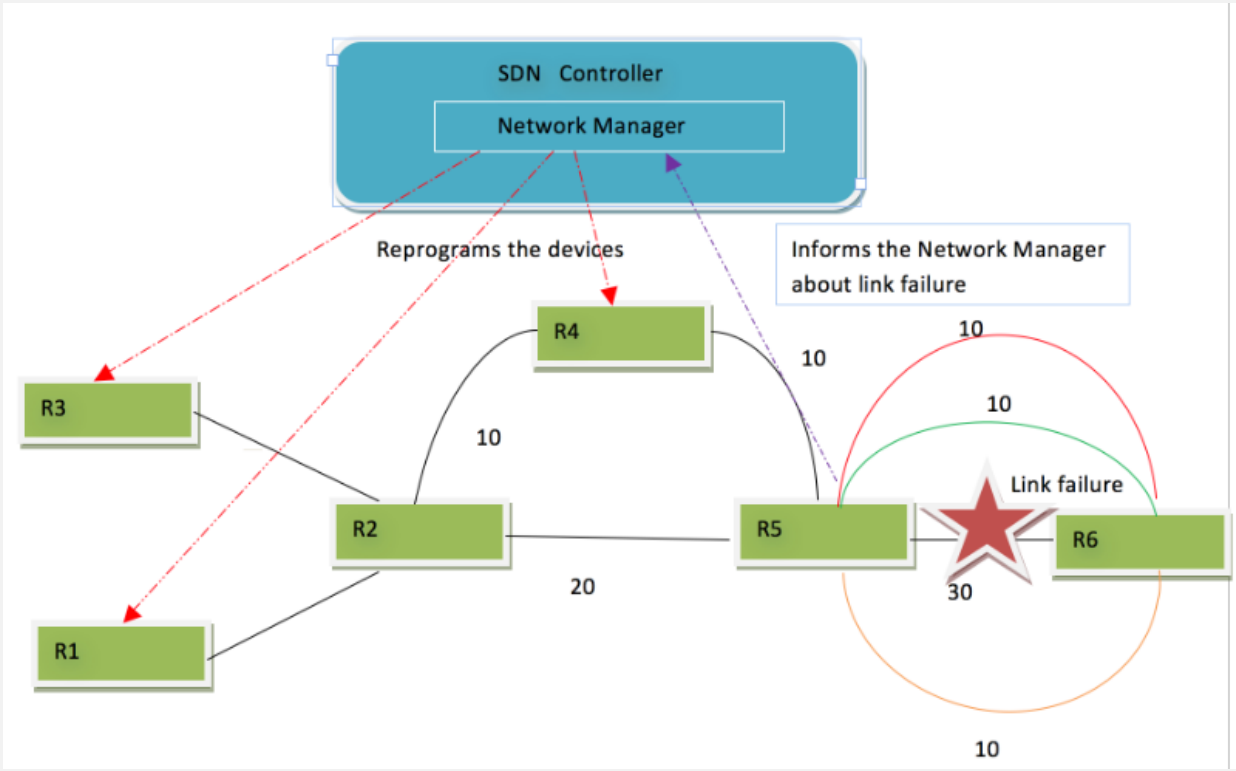
SOFTWARE-DEFINED NETWORKING (SDN)

Open Networking Foundation (ONF) definition:

The physical separation of the network control plane from the forwarding plane, and where a control plane controls several devices.

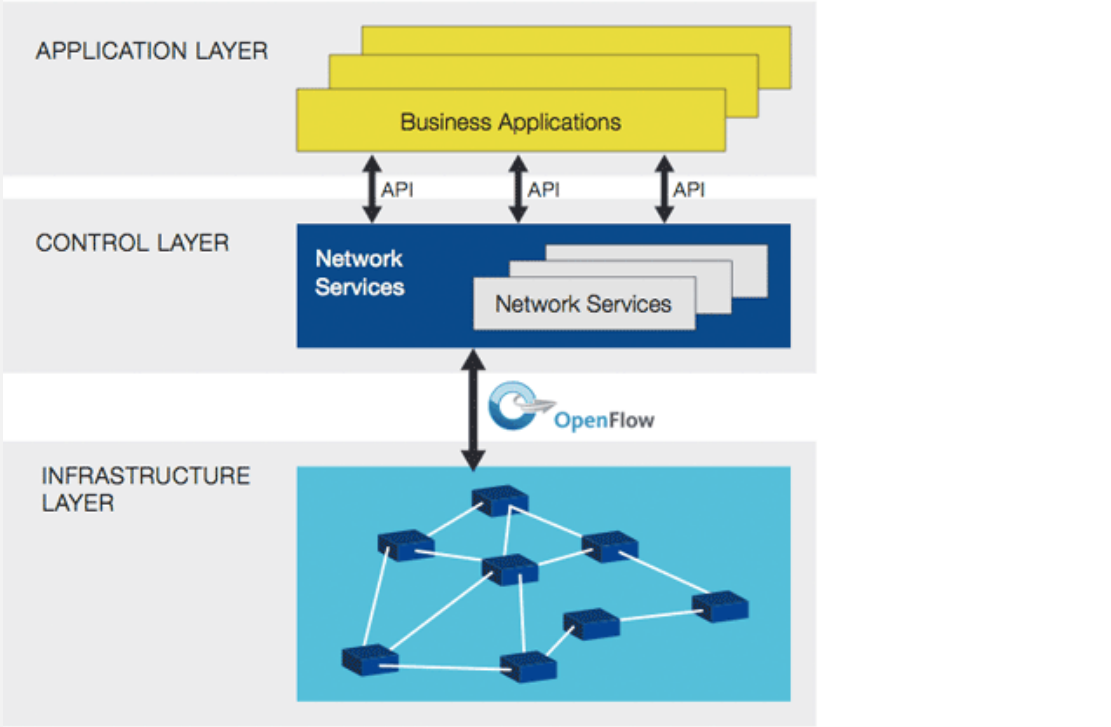
- Decouples network control and forwarding functions
- Abstracts underlying infrastructure for apps and services
- Directly programmable
- Agile
- Centrally managed
- Programmatically configured
- Open standards-based and vendor-neutral

NETWORK EXAMPLE REVISITED WITH SDN



https://www.linkedin.com/pulse/network-manager-running-external-controller-sdn-rakesh-raushan

SDN HIGH-LEVEL VIEW



Source: https://www.opennetworking.org/sdn-resources/sdn-definition

SDN COMPONENTS

- Control Plane
  - Abstract view of network infrastructure
  - Enables admins to apply custom policies/protocols across network hardware
- Northbound Application Interfaces
  - Software interfaces between modules of controller and SDN applications running on top of network platform
- East-West Protocols
  - Manage interactions between controllers (if multi-controller-based)
- Data Plane and Southbound Protocols
  - Forwarding hardware
  - Protocols to manage interface between network devices

SDN LAYERS

- Infrastructure Layer:
  - Contains SDN devices performing packet switching and forwarding
  - SDN devices composed of API for communication with controller, abstraction layer, packet-processing component
  - SDN device abstracted as a set of flow tables
- Control Layer:
  - Centralized control to supervised network forwarding behavior
  - Controls SDN devices making up network infrastructure
  - Implements policy decisions (routing, forwarding, load balancing etc.)
- Application Layer:
  - End-user apps utilizing SDN communications and network services
  - Affect behavior of infrastructure by configuring flows, load-balancing, reacting to link failures, redirecting traffic, authentication, etc.

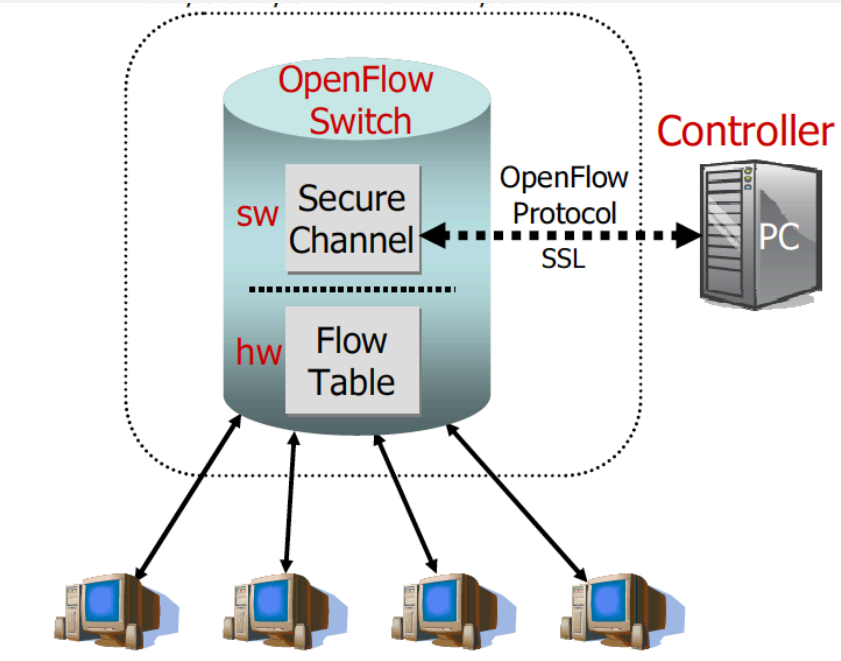
OPENFLOW

- Idea:
  - Separation of data and control planes
  - Centralized control
  - Flow-based control
- History:
  - 2006: Martin Casado (Stanford Phd student) et al. proposed network security architecture based on centralized instead of edge security
  - 2008: OpenFlow paper in ACM SIGCOMM CCR
  - 2009: Nicira co-founded by Martin Casado
  - March 2011: Open Networking Foundation (ONF) formed
  - October 2011: First Open Networking Summit
  - 2012: VMWare bought Nicira

OPENFLOW TERMINOLOGY

- Flow
  - Network traffic partitions (TCP connection, packets with same MAC address or IP, packets with same VLAN tag, packets arriving from same switch port etc.)
- Switch
  - Includes one or more flow tables and a group table
  - Performs packet look-up and forwarding
  - Each flow table made up of flow entries consisting of header fields, counters, actions
- Channel
  - Interface connecting each switch to a controller
- Controller
  - Responsible for determining how to handle packets without valid flow entries
  - Maintains all network protocols and policies and distributing appropriate instructions to network devices

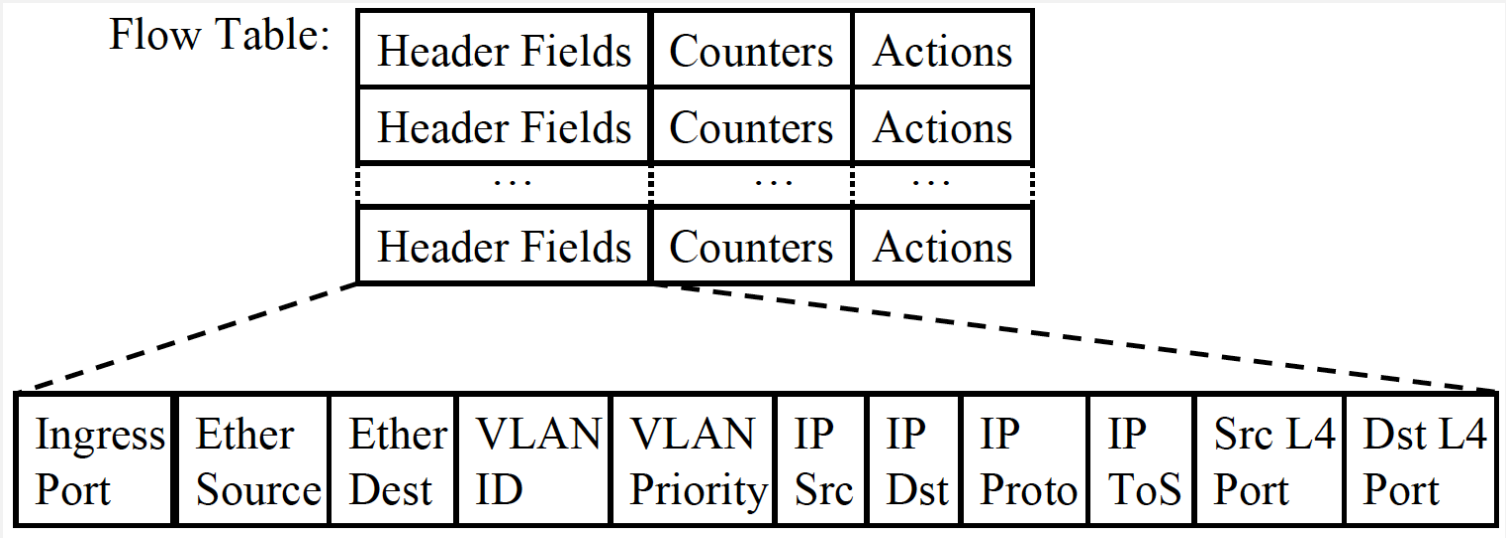
OPENFLOW SWITCH



- Control logic moved to controller
- Only forwarding elements at switches
- Identifying traffic flows through packet matching, packet forwarding reporting statistics and switch state
- One complex controller, many dumb switches
- Use OpenFlow protocol for communication of forwarding rules from controller to switches

Source: http://archive.openflow.org/documents/openflow-wp-latest.pdf

FLOW TABLES IN OPENFLOW (V1.0)



Source: http://www.cse.wustl.edu/~jain/tutorials/unsw14.htm

FLOW TABLE EXAMPLE

Port	Src MAC	Dst MAC	VLAN ID	Priority	EtherType	Src IP	Dst IP	IP Proto	IP ToS	Src L4 Port ICMP Type	Dst L4 Port ICMP Code	Action	Counter
*	*	0A:C8:*	*	*	*	*	*	*	*	*	*	Port 1	102
*	*	*	*	*	*	*	192.168.*.*	*	*	*	*	Port 2	202
*	*	*	*	*	*	*	*	*	*	21	21	Drop	420
*	*	*	*	*	*	*	*	0x806	*	*	*	Local	444
*	*	*	*	*	*	*	*	0x1*	*	*	*	Controller	1

Source: S. Azodolmolky, "Software Defined Networking with OpenFlow," Packt Publishing, October 2013, 152 pp., ISBN:978-1-84969-872-6 (Safari Book)

OPENFLOW SWITCHES

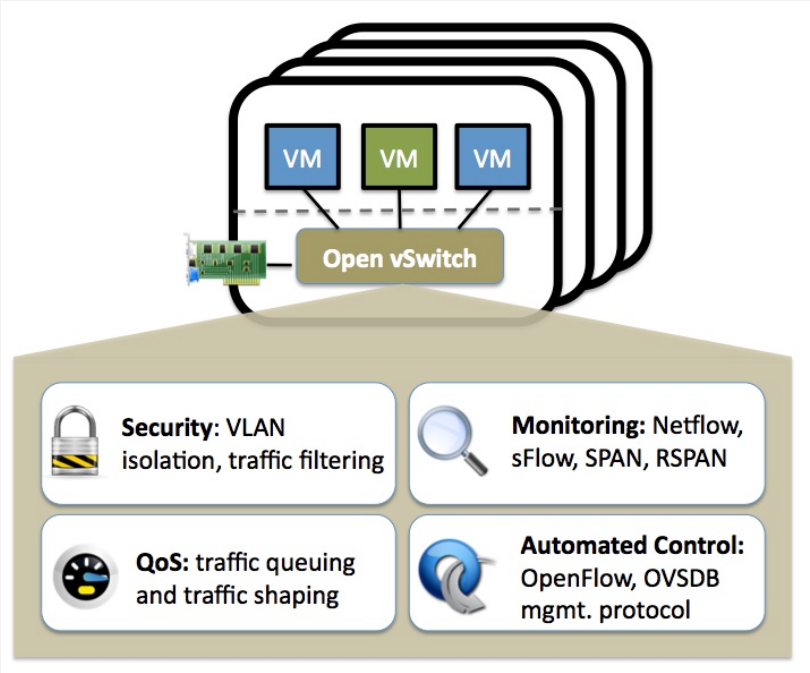
Hardware:

- Arista 7050
- Brocade MLXe, Brocade CER, Brocade CES
- Extreme Summit x440, x460, x670
- Huawei openflow-capable router platforms
- HP 3500, 3500yl, 5400zl, 6200yl, 6600, and 8200zl
- HP V2 line cards in the 5400zl and 8200zl IBM 8264
- Juniper (MX, EX)
- NEC IP8800, NEC PF5240, NEC PF5820
- NetGear 7328SO, NetGear 7352SO
- Pronto (3290, 3295, 3780)

Software:

- Indigo
- LINC
- Pantou
- Of13softswitch
- XORPlus
- Open vSwitch

OPEN VSWITCH



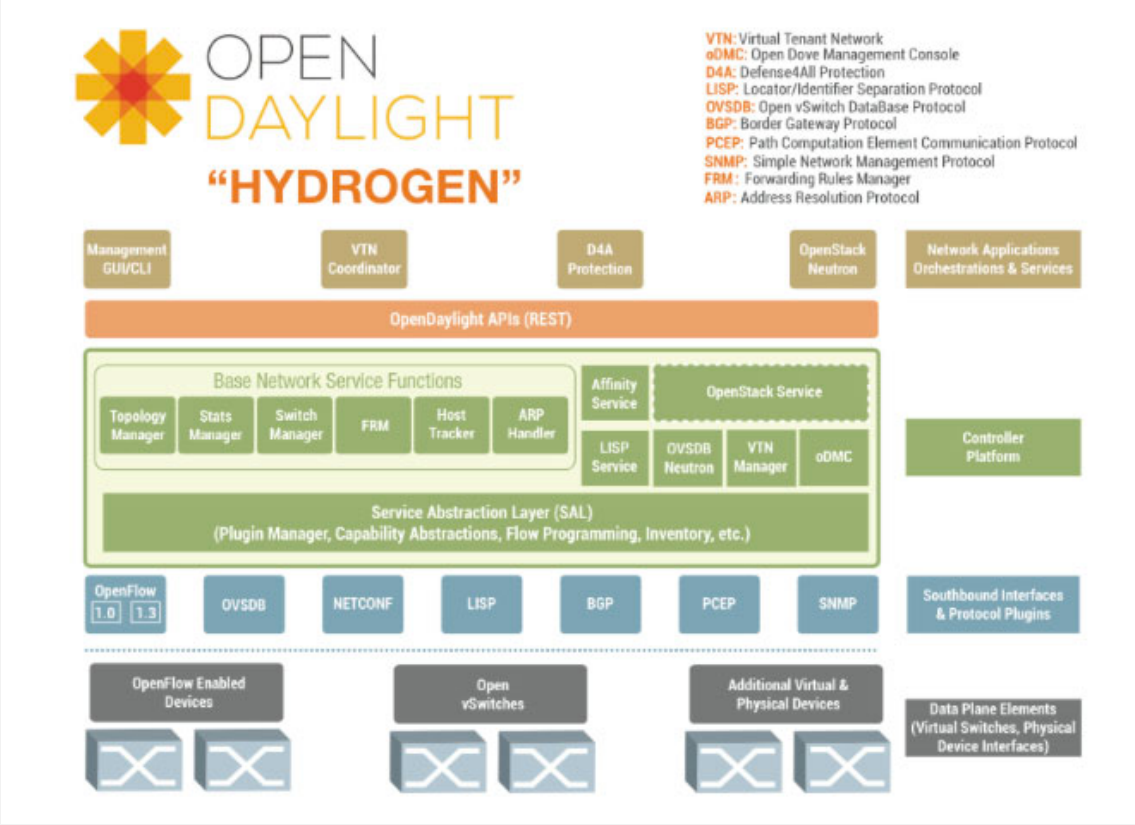
- Multi-layer, open-source virtual switch
- Designed for massive network automation, still supporting standard protocols (e.g. NetFlow, sFlow, IPFIX, 802.1ag …)
- Can operate both as a soft switch and as control stack for switching silicon
- Default switch in XenServer 6.0, Xen Cloud Platform
- Supports Xen, KVM, Proxmox VE, VirtualBox
- Integrated into VMS including OpenStack, openQRM, OpenNebula, oVirt
- Packages available for Ubuntu, Debian, FreeBSD, NetBSD

Source: http://vswitch.org/

OPENFLOW CONTROLLERS

- ONOS
- Floodlight
- ONIX
- NOX
- POX
- SNAC
- Beacon
- Trema
- Maestro
- …

SDN ≠ OPENFLOW



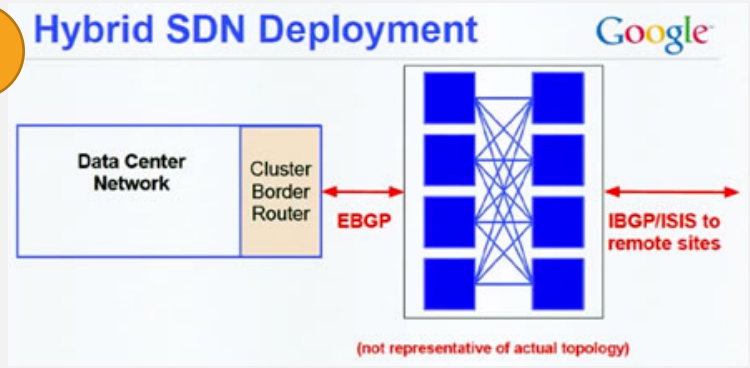
Source: https://www.sdxcentral.com/sdn/definitions/sdn-controllers/opendaylight-controller/

SDN CONTROLLERS

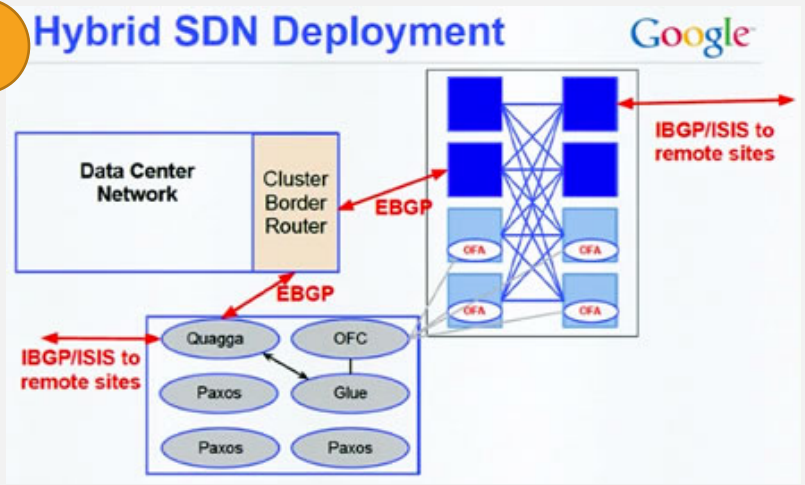
- Brain of SDN
- Maintains view of entire network
- Implements policy decisions
  - Routing
  - Redirecting
  - Load balancing
  - …
- Controls all SDN devices in network
- Provides northbound API for applications
- Come with set of common application modules (learning switch, firewall, load balancer etc.)

GOOGLE'S INCREMENTAL SDN DEPLOYMENT

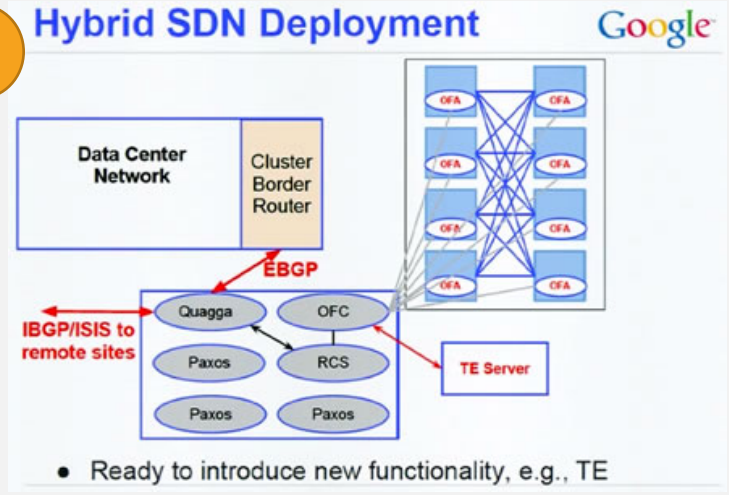
1



2



3



Source:
http://www.networkcomputing.com/networking/inside-googles-software-defined-network/512240144

HOW SDN PROVIDES AGILITY

- Globally-connected, intelligent environments
- Help with load-balancing
- Global traffic management by logically sending traffic to appropriate data center
- More fluid data center traffic flow automation
- Provide decreased downtime
- Increased data resiliency
- Improved disaster recovery planning

http://www.datacenterknowledge.com/archives/2014/10/16/with-cloud-data-center-technology-has-changed/

SDN & IOT

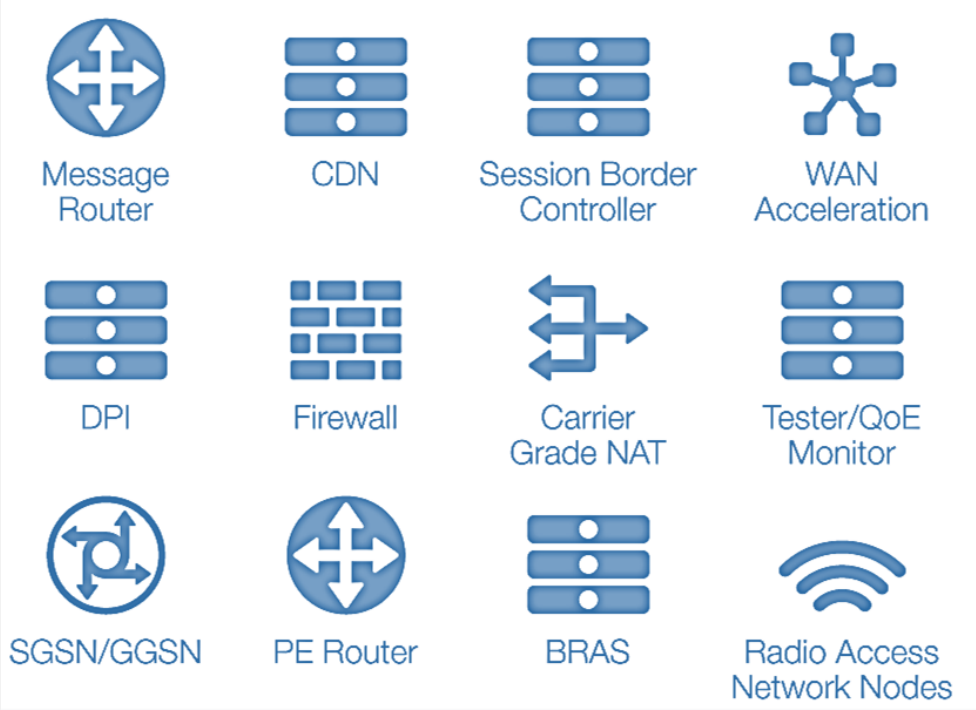
- SDN will be a key enabler for IoT
- Most useful features:
  - ❖ Service chaining
    - Enable operators to provision features like VPNs, firewalls etc., and set policy tolerances
  - ❖ Dynamic load management
    - Enable operators to monitor and orchestrate bandwidth changes automatically (facing exponential increase in devices)
  - ❖ Bandwidth calendaring
    - Enable operators to schedule when and how much traffic a customer or app will need at a specific time

http://www.networkworld.com/article/2601926/sdn/sdn-vital-to-iot.html

PART II

NFV MOTIVATION (CONT.)

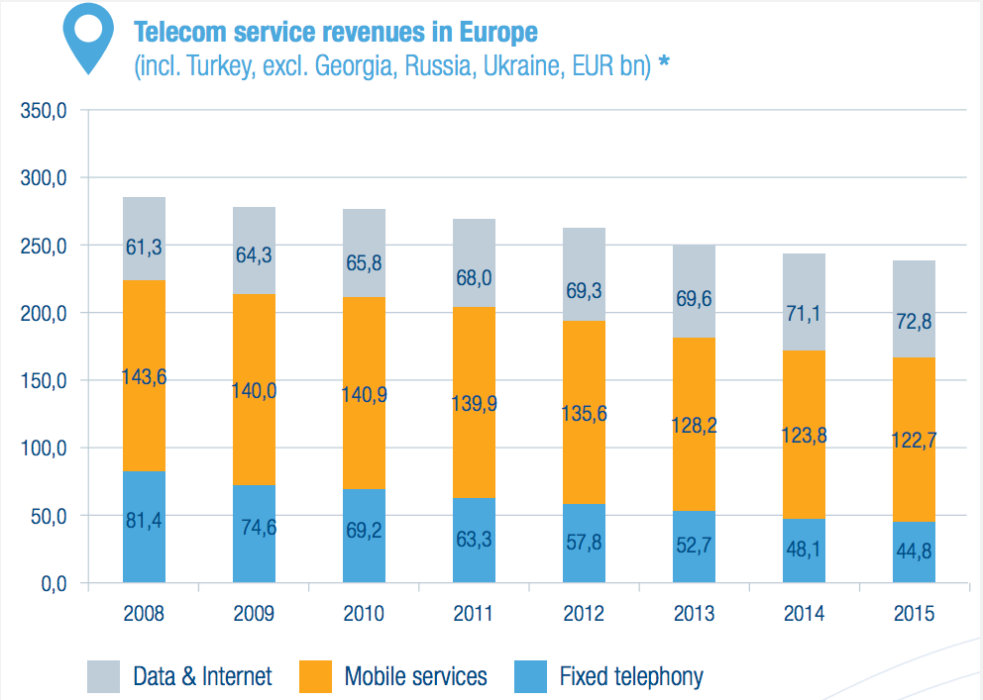
- Hardware-based appliances rapidly reaching end of life
- Procure-design-integrate-deploy cycles repeated with no revenue benefit



Adapted from: Network Functions Virtualization -Everything Old Is New Again. F5 White Paper. August 2013

NETWORK FUNCTIONS VIRTUALIZATION (NFV) – MOTIVATION

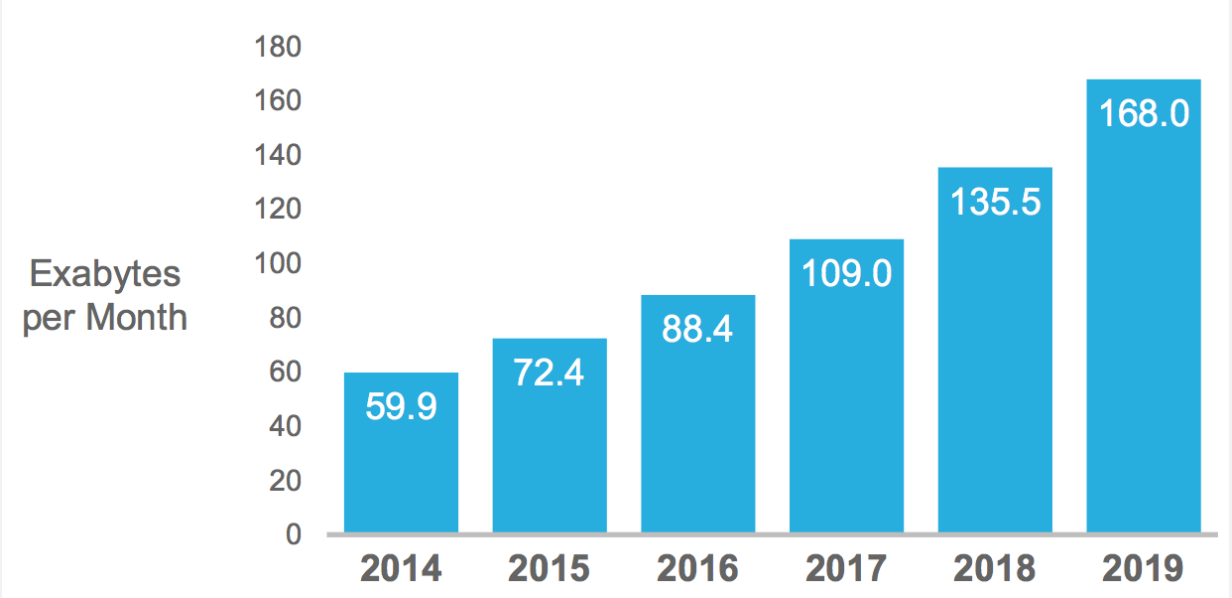
- Hardware lifecycles becoming shorter as technology and services innovation accelerates
- Inhibited roll out of new revenue earning network services



Source: European Telecommunications Network Operators' Association Annual Economic Report, 2015
https://www.etno.eu/datas/publications/economic-reports/AER2015_Final.pdf

NFV MOTIVATION (CONT.)

Big Data, Increasing CapEx



Cisco VNI Global IP Traffic Forecast, 2014–2019.

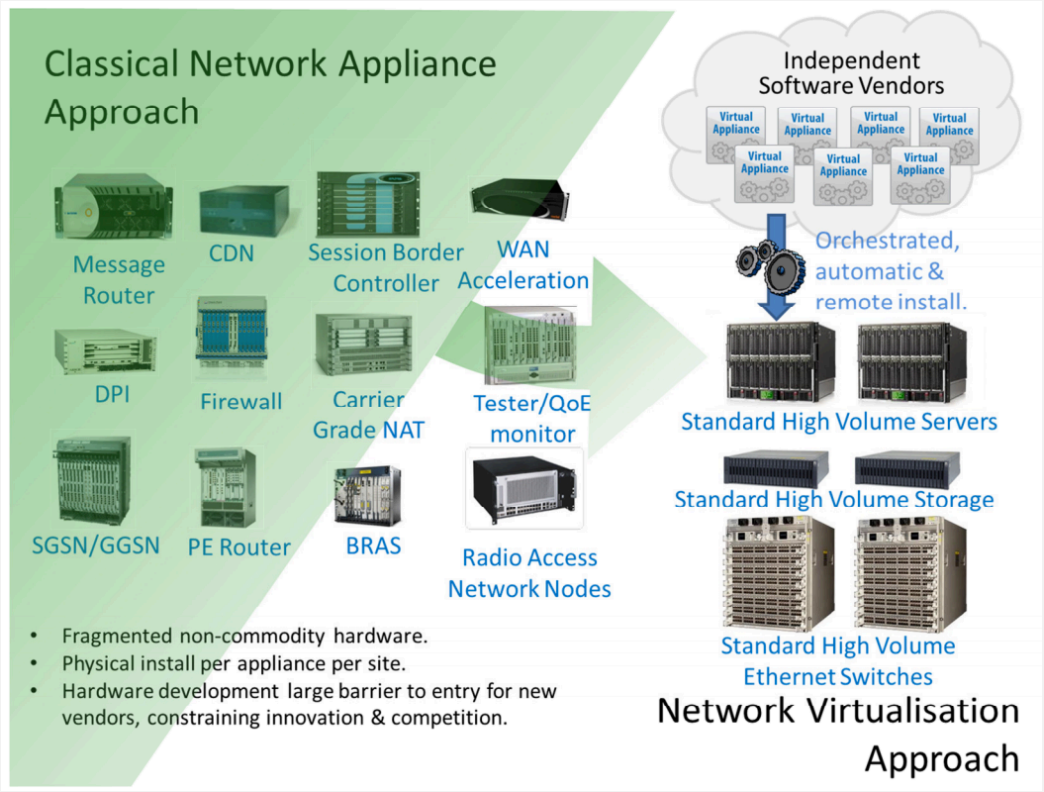
Source: https://www.ciscoknowledgenetwork.com/files/527_2015_VNI_Complete_Forecast_-Master-Global__CKN_6102015.pdf?PRIORITY_CODE=

WHAT IS NFV?

ETSI Definition of NFV:

Implementation of network functions in ***software*** *that can run on a range of industry* ***standard server hardware****, and that can be* ***moved to or instantiated in****, various locations in the network as required,* ***without*** *the need for installation of* ***new equipment****.*

NFV VS. LEGACY NETWORKS



Source: https://portal.etsi.org/NFV/NFV_White_Paper.pdf

NFV VS. LEGACY NETWORKS (CONT.)

- Decoupled software-hardware:
  - HW and SW evolve independently
- Flexible network function deployment:
  - Reassignment and sharing of infrastructure resources
  - HW & SW performing different functions at different times
  - Automated instantiation of network function software
- Dynamic operation:
  - Scaling VNF performance dynamically according to actual traffic

NFV TIMELINE

Date	Event
10/2012	Network operators call for action white paper
11/2012	New network operator-led Industry Specification Group (ISG) with open membership setup under ETSI (European Telecommunications Standards Institute)
1/2013	1st meeting of NFV ISG (100+ participants, 20+ carriers)
10/2013	2nd white paper + High-level NFV ISG documents
06/2014	Detailed NFV ISG documents
09/2014	OPNFV
10/2014	3rd white paper
06/2015	OPNFV ARNO
02/2016	Open Source MANO
03/2016	OPNFV Brahmaputra

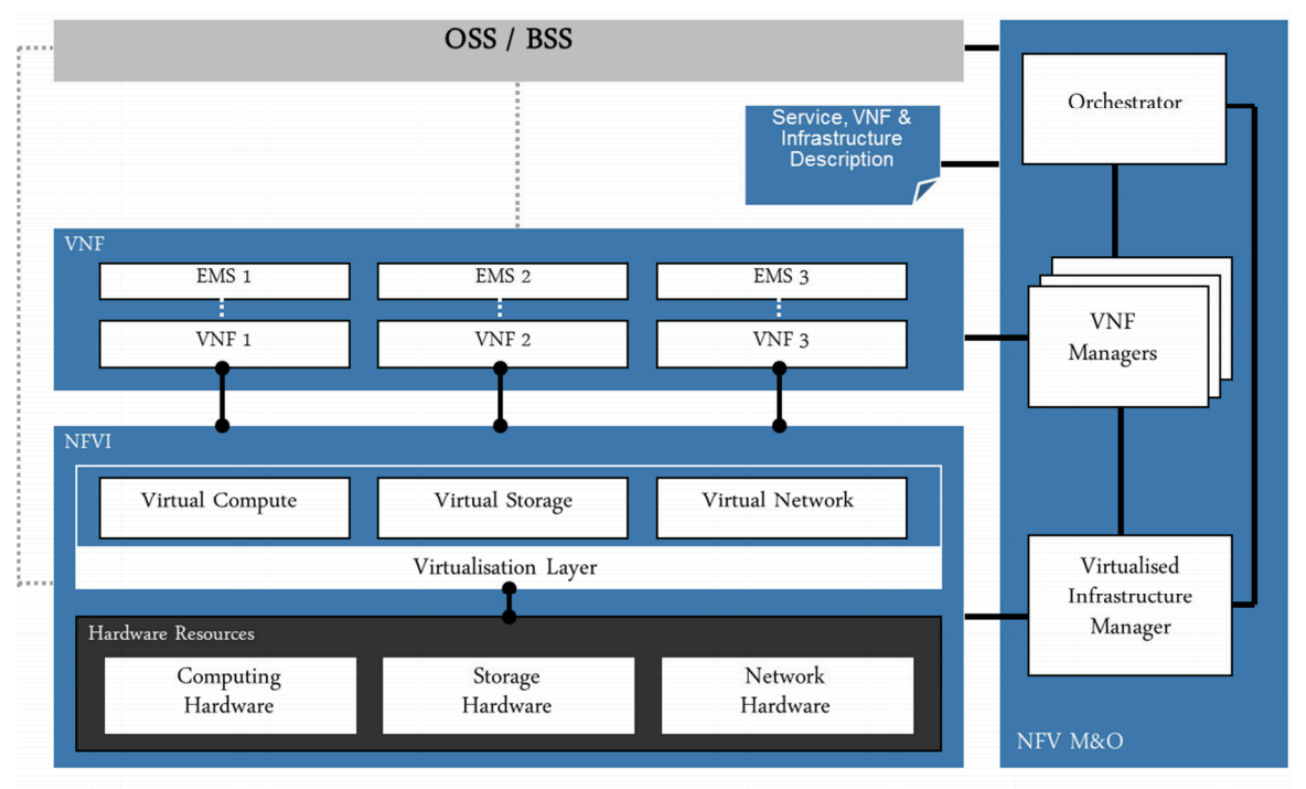
NFV BENEFITS

- Reduced equipment costs and power consumption
- Increased speed of time-to-market by minimizing network operator cycle of innovation
- Efficient test and integration, reduced development cost
- Availability of network appliance multi-version and multi-tenancy
- Targeted service introduction based on geography or customer sets
- Optimizing network configuration and/or topology in near real time based on actual traffic/mobility patterns
- Rapid service scalability (up/down)
- Automated reconfiguration to repair failures
- Wide variety of ecosystems and openness

NFV CHALLENGES

- Portability/interoperability
- Performance trade-off
- Migration and co-existence of legacy & compatibility with existing platforms
- Management and Orchestration
- Automation
- Security & Resilience
- Network Stability
- Simplicity
- Integration of multiple virtual appliances from multiple vendors

NFV ARCHITECTURAL FRAMEWORK



Source: https://portal.etsi.org/NFV/NFV_White_Paper2.pdf

NFV FRAMEWORK COMPONENTS

- **NFVI (Network Functions Virtualization Infrastructure):**
  - Provides virtual resources to support execution of VNFs
  - Includes COTS hardware, accelerator components, virtualizing software layer
- **VNF (Virtualized Network Function):**
  - Software implementation of a network function, capable of running on NFVI
  - Corresponds to legacy network nodes
- **NFV MANO (Management and Orchestration):**
  - Responsible for orchestration and lifecycle management of resources for infrastructure virtualization, and lifecycle management of VNFs
  - Interacts with OSS/BSS landscape, allowing NFV to be integrated into existing network-wide management landscape

NFVI

- Totality of all hardware and software components building up the environment in which VNFs are deployed, managed and executed
- Provides multi-tenant infrastructure based on industry-standard virtualization to support multiple use cases/fields of application simultaneously
- Implemented as a set of distributed set of NFVI nodes in various NFVI Points of Presence (PoP)
  - Supports locality
  - Provides low latency
- VNFs may be dynamically deployed on NFVI on demand within capacity limits of NFVI nodes

NFVI RESOURCES

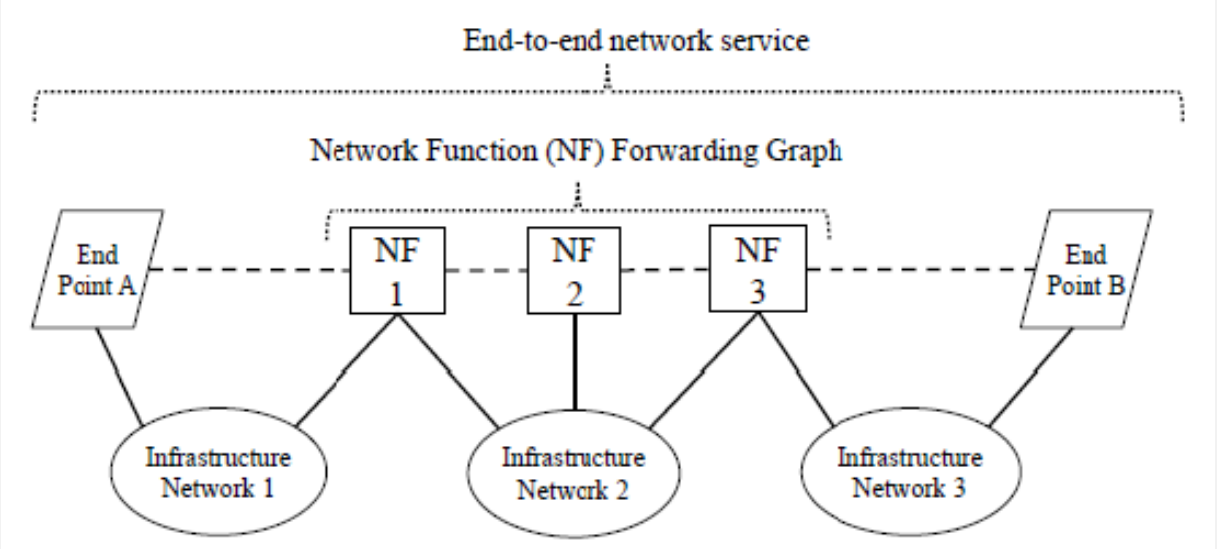
- Hardware resources:
  - Computing, storage, and network providing processing, storage and connectivity to VNFs through the virtualization layer
  - Computing resources: COTS
  - Storage resources: Shared network attached storage / storage on server
- Software (virtualized) resources:

Virtualization layer responsible for:

- Abstracting and logically partitioning physical resources
- Enabling software implementing VNF to use virtualized infrastructure
- Providing virtualized resources to VNF

NETWORK SERVICES

- A forwarding graph of network functions (NFs) and end points interconnected by supporting network infrastructure
- E.g. mobile voice/data, Internet access, VPN, firewall, load balancer, set of CDN servers etc.
- End points and NFs (devices, apps, physical server apps) represented as nodes



Source:
http://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.01.01_60/gs_NFV002v010101p.pdf

SERVICE FUNCTION CHAINING (SFC)

Definition by OPNFV:
Defining an ordered list of network services (e.g. firewalls, NAT, QoS) stitched together in the network to create a service chain.

- Current networks comprised of diverse NFs
- NFs connected or chained in a certain way to achieve desired overall functionality the network is designed to provide
- Most services defined by statically combining NFs in a way expressed using an NF Forwarding Graph or NF Set
- Order of connectivity matters in NF forwarding graphs (service chains) (e.g. web server tier firewall, NAT, load balancer) as opposed to NF sets (e.g. residential gateway)

SFC IN THE PAST

- Great time and effort spent on building service chains
- Required acquiring network devices and connecting them in needed sequence
- Specialized hardware for each service
- Device-specific configuration syntax
- Error-prone
- Problem in one component leading to failure in whole network
- Device resizing to support increased application load in future
- Chains usually built to support multiple apps, data passage through unnecessary devices consuming extra bandwidth & CPU cycles

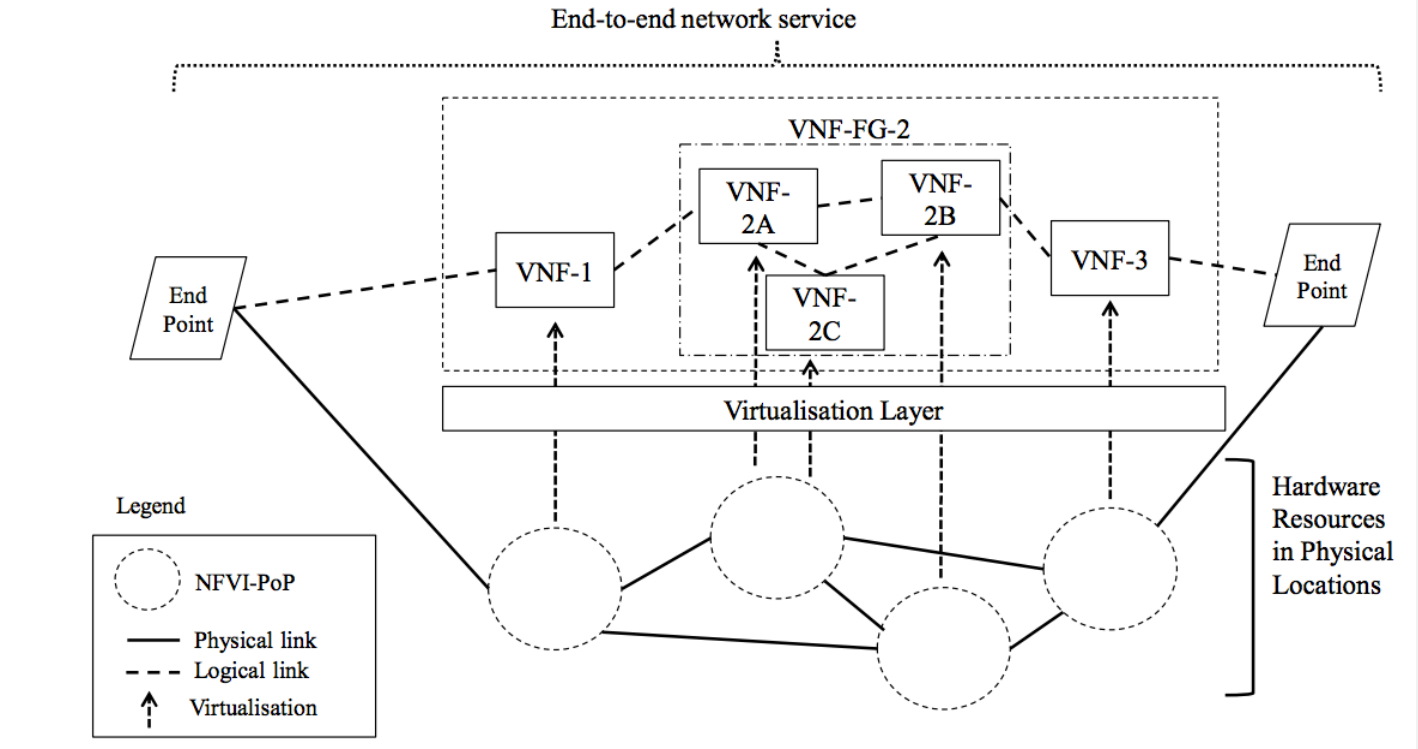
http://searchsdn.techtarget.com/tip/How-SDN-and-NFV-simplify-network-service-chain-provisioning

VIRTUALIZED NETWORK FUNCTIONS (VNFS)

- Software implementation of a network function capable of running over the NFVI
- Examples:
  - 3GPP Evolved Packet Core (EPC) network elements
  - Elements in a home network (residential gateway etc.)
  - Dynamic Host Configuration Protocol (DHCP) servers
  - Firewalls

  …
- Can contain multiple components (e.g. deployed over multiple VMs, each VM hosting a single component of the VNF)

E2E NETWORK SERVICE WITH VNFS



Adapted from
http://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.01.01_60/gs_NFV002v010101p.pdf

HOW SDN & NFV IMPROVE SFC

- SDN moves management out of hardware
  - Standardized configuration protocol between controller and network devices
  - Entire service chains provisioned and constantly configured from controller
  - Reduced chance of inconsistent device configurations

- NFV moves network functions out of hardware
  - No need to acquire hardware
  - Increased bandwidth achieved with additional VMs
  - No need for overprovisioning

http://searchsdn.techtarget.com/tip/How-SDN-and-NFV-simplify-network-service-chain-provisioning

NFV MANO

- Responsible for management and configuration of VNFs and other subsets of NFV reference framework
- Elements:
  - Virtual Infrastructure Manager (VIM)
    - Management and allocation of virtual resources (e.g. addition/removal/update of computer/storage/network resources)
  - VNF Manager
    - Configuration, lifecycle management, element management of VNFs
  - Orchestrator
    - Orchestration and management of NFV infrastructure and software resources, and realization of network services on NFVI

NFV APPLICATION FIELDS

- Switching elements: BNG, CG-NAT, routers
- Mobile network nodes: HLR/HSS, MME, SGSN, GGSN/PDN-GW, RNC, Node B, eNode B
- Functions contained in home routers and set top boxes to create virtualized home environments
- Tunneling gateway elements: IPSec/SSL VPN gateways
- Traffic analysis: DPI, QoE measurement
- Service Assurance, SLA monitoring, Test and Diagnostics
- NGN signaling: SBCs, IMS
- Converged and network-wide functions: AAA servers, policy control and charging platforms
- Application-level optimization: CDNs, Cache Servers, Load Balancers, Application
- Accelerators
- Security functions: Firewalls, virus scanners, intrusion detection systems, spam protection

…

NFV USE CASES (BY ETSI)

Cloud	Networks Functions Virtualization as a Service (NFVIaaS)
	Virtual Network Function as a Service (VNFaaS)
	Virtual Network Platform as a Service (VNPaaS)
Mobile	VNF Forwarding Graphs
	Virtualization of Mobile Core Network and IMS
	Virtualization of Mobile Base Station
CDN	Virtualization of Content Delivery Networks (vCDN)
Access	Virtualization of the Home Environment
	Fixed Access Network Functions Virtualization

OPNFV

- Carrier-grade, integrated, open source platform to accelerate the introduction of new NFV products and services
- Brings together the work of standards bodies, open source communities and commercial suppliers to deliver a standard open source NFV platform
- Focus: Building NFV Infrastructure (NFVI) and Virtualized Infrastructure Management (VIM) by integrating components from projects like OpenDaylight, OpenStack, Ceph Storage, KVM, Open vSwitch, and Linux
- Goals:
  - Develop tested open source platform to build NFV functionality
  - Include participation of end users to validate meeting of needs
  - Ensure consistency, performance and interoperability among components
  - Establish NFV solutions ecosystem based on open standards
  - Promote OPNFV as the preferred platform for NFV

Source: https://www.opnfv.org/about

OPNFV MEMBERS

Source: https://www.opnfv.org/about/join, as of 06/22/2016

OPEN SOURCE MANO (OSM)

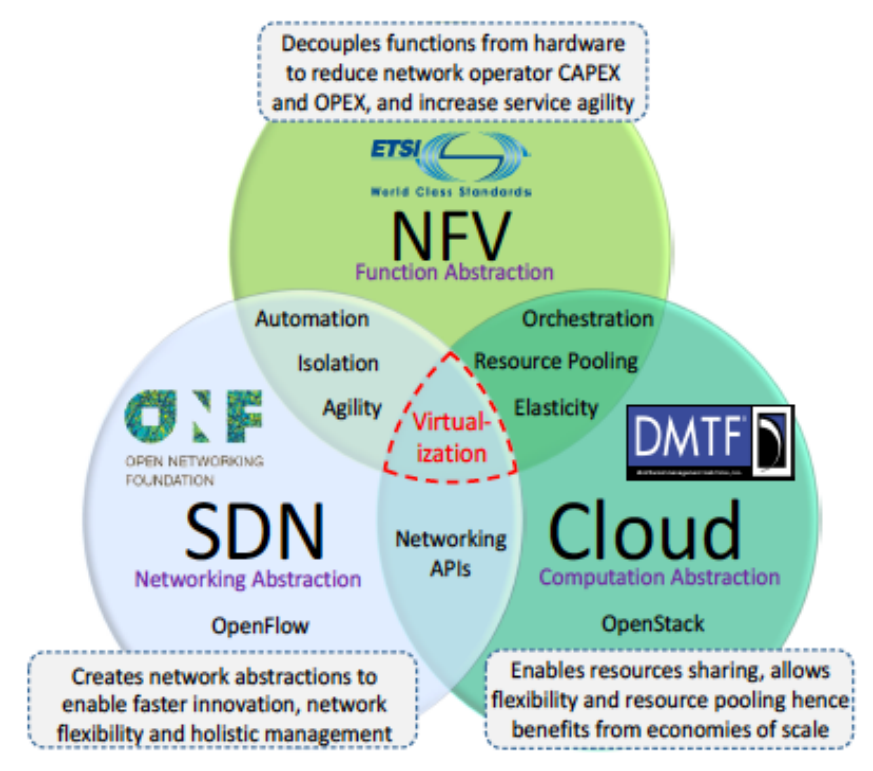
- ETSI-hosted project to develop open source NFV Management and Orchestration (MANO) stack aligned with ETSI NFV
- Will provide regularly updated reference implementation of NFV MANO
- Members:
  - Benu Networks, BT Group Pic, Canonical Group Ltd, Comptel Corporation, EURECOM, Intel Corporation Ltd, Mirantis Inc, Red Hat Limited, RIFT.io Inc, Sprint Corporation, Telefonica S.A., Telenor ASA, University of the Basque Country
- Participants:
  - Ampliphae Ltd, Atrinet, Brocade, DELL Inc., Indra Sistemas S.A., Layer123, Metaswitch Networks, Nextworks, Radware Ltd., Sigscale Global Inc., SK Telecom, VMWare Inc., XFLOW Research Inc.

Source: https://osm.etsi.org/

ENTERPRISE NFV

http://www.cisco.com/c/en/us/solutions/enterprise-networks/enterprise-network-functions-virtualization-nfv/index.html

SDN-NFV-CLOUD RELATIONSHIP



Source: Rashid Mijumbi, Joan Serrat, Juan Luis Gorricho, Niels Bouten, Filip DeTurck, Raouf Boutaba, "**Network Function Virtualization: State-of-the-art and Research Challenges**". IEEE Communications Surveys and Tutorials. First Q., 2016.

NFV-SDN RELATIONSHIP

- NFV approaches relying on separation of control and data forwarding (as in SDN) can:
  - enhance performance
  - simplify compatibility with existing deployments
  - facilitate operation and maintenance procedures
- SDN Controller maps to network controller component of NFVI
  - Control physical/virtual switching
  - Network monitoring
- NFV can support SDN by providing infrastructure upon which SDN software can run
- NFV and SDN objectives closely aligned in terms of using commodity servers and switches

SDN-NFV USE CASES (BY SDXCENTRAL)

Use case	Description	Subcases
Network Access Control	Setting appropriate privileges for users or devices accessing the networks, including access control limits, incorporation of service chains and quality of service.	Campus NAC, Remote Office NAC, M2M NAC, Unified Communications Optimization
Network Virtualization	Creating an abstracted virtual network on top of a physical network, allowing a large number of multi-tenant networks to run over a physical network	Data Center VNs, Campus/Branch VNs, Data Center Micro Segmentation, Network Functions as a Service
Virtual Customer Edge	Creating a virtualized platform on customer premises or pulling functions closer to the core on a multi-tenant platform	On-premises vCPE, vCE
Dynamic Interconnects	Creation of dynamic links between locations, as well as dynamically applying appropriate QoS and BW allocation to those links.	BWoD, Virtual Private Interconnects, Dynamic Enterprise VPN, Cross-Domain Interconnect, Multi-layer Optimization
Virtual Core and Aggregation	Virtualized core systems for service providers	vEPC, vIMS, vPE, GiLAN, Mobile Network Virtualization
Datacenter Optimization	Optimizing networks to improve application performance, orchestrating workloads with networking configuration	Big Data Optimization, Mice/Elephant Flow Optimization

Source: https://www.sdxcentral.com/sdn-nfv-use-cases/

NFV R&D PROBLEMS

- Virtualization Layering and NFVI Support:
  - Supporting VNF portability over heterogeneous virtualization layer
  - Need to design cost-efficient, high-performance virtualization layer and VNF deployment techniques to support diverse NFs in operator network

- VNF Software Architecture:
  - Smaller functional modules for scalability, reusability, faster response
  - Multiple VNFs composed to reduce management and forwarding graph complexity
  - Need to exploit flexibility of virtualization to realize modular, portable, hardware-independent, reusable, scalable software design

NFV R&D PROBLEMS (CONT.)

- Management and Orchestration:
  - E2E service to NFV network mapping
  - Instantiating VNFs at appropriate locations
  - Allocating and scaling hardware resources
  - Keeping track of VNF instance locations
  - Determining faults and correlating for recovery
- Performance:
  - VNF may have less per-instance capacity than NF on dedicated physical hardware
  - Need to split workload across distributed VNF instances
  - Need to minimize performance degradation when porting VNF instance

NFV R&D PROBLEMS (CONT.)

- Reliability
  - VNF should be at least as reliable as equivalent legacy NF
  - Need to investigate single-point-of-failure, fault detection, recovery methods etc.
- Security
  - Multiple vendors involved
  - Virtualization-relevant issues
  - Usage of shared storage and networking
  - Interconnectivity among NFV components exposing new vulnerable interfaces
  - Isolation of VNFs

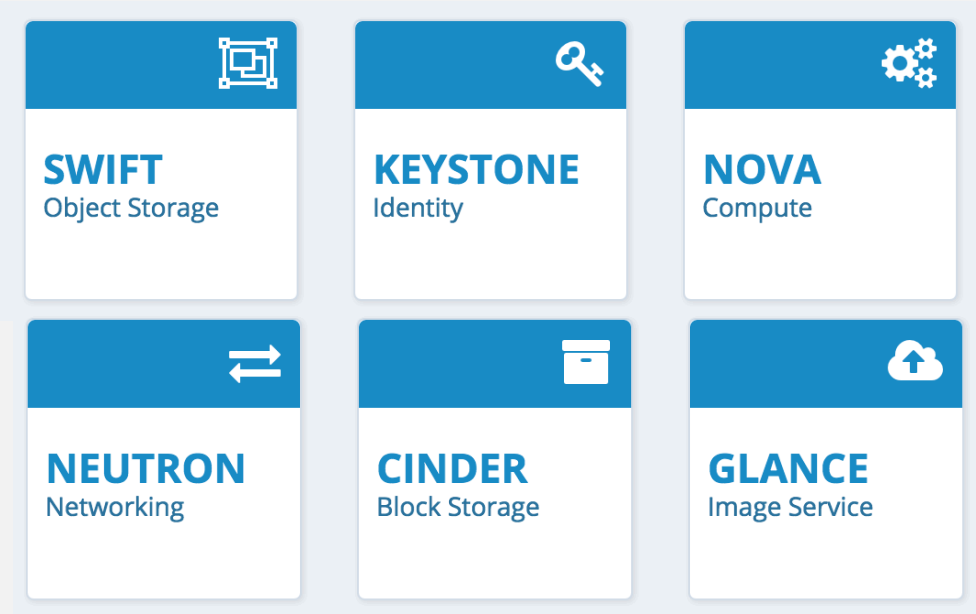
NFV CONTAINERS

- Performance penalty for hypervisor-based virtualization too high
- Hypervisor-based virtualization not elastic enough to support web-based services at scale
- Why not run VNFs in containers?
- Telcos can reduce VM overhead with containers
- Faster turn-up time, reduced CPU and memory footprint
- Fit more VNFs on a single server
- Reduced CapEx
- AT&T already looking into microservices with containers into NFV solutions

PART III

OPENSTACK

- Open-source platform for orchestrating and maintaining clouds
- Creates abstraction layer above compute, storage and network resources



http://openstack.org

OPENSTACK OPTIONAL SERVICES

HORIZON
Dashboard

CEILOMETER
Telemetry

HEAT
Orchestration

TROVE
Database

SAHARA
Elastic Map Reduce

IRONIC
Bare-Metal Provisioning

ZAQAR
Messaging Service

MANILA
Shared Filesystems

DESIGNATE
DNS Service

BARBICAN
Key Management

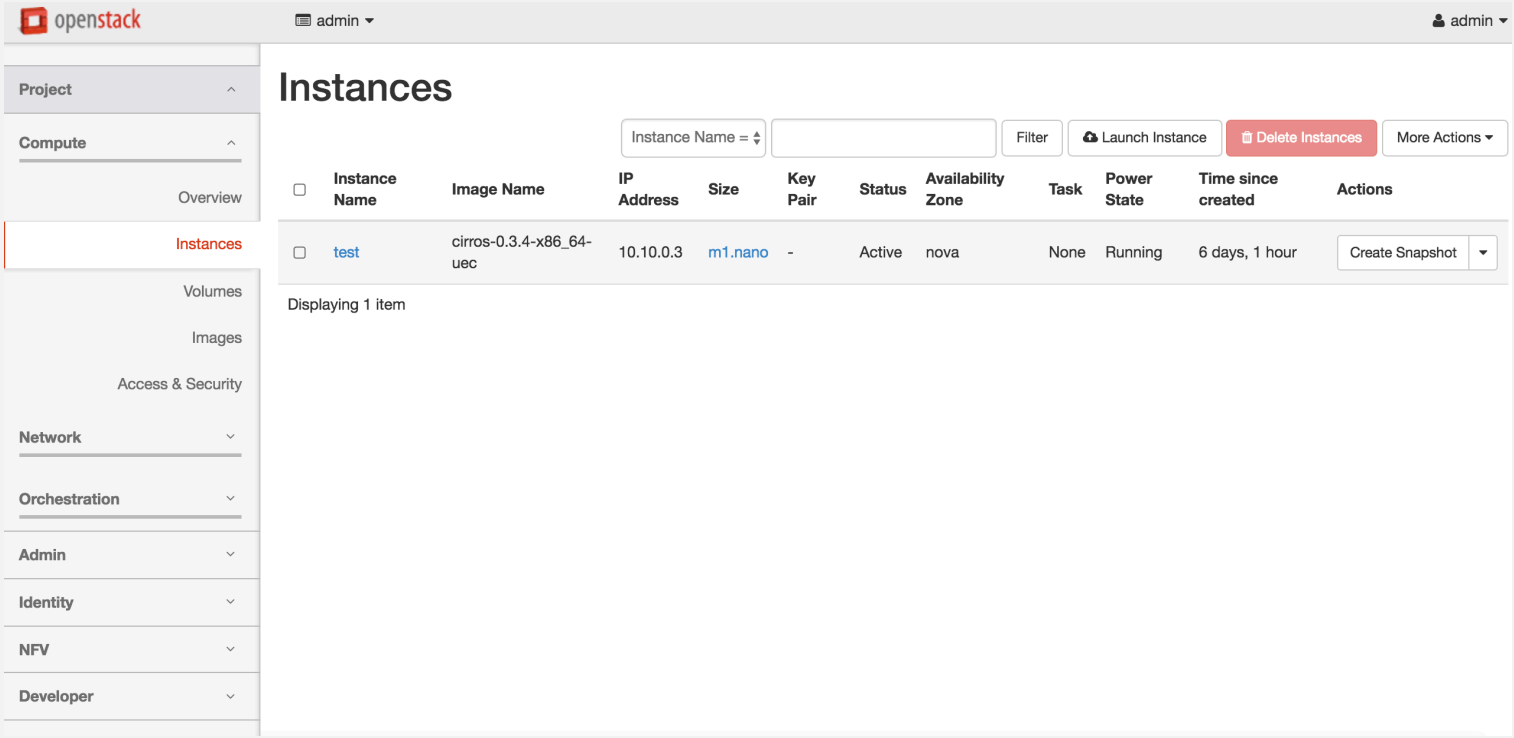
MAGNUM
Containers

MURANO
Application Catalog

CONGRESS
Governance

http://openstack.org

OPENSTACK DASHBOARD



LAB SETUP

- All-in-one OpenStack setup on VMs provided by Ravello Systems
  - Nested virtualization
  - Mitaka + Tacker
- Multi-user access
- Access OpenStack through Horizon (dashboard)

LAB TASKS

- Create virtual networks
- Launch VM instances in virtual networks
- Check connectivity between in-network/outside-network instances
- Implement virtual routers between networks
- Implement virtual firewall
- Implement virtual load balancer

LAB

1. Login to OpenStack dashboard (Horizon) with provided user information (e.g. username: user-X, password: user-X)
2. On the left pane, click on "Networks"
3. On the Networks page, click on "Create Network"
4. On the network creation window, type a network name (net-user-X-1) and click on "Next"
5. On the subnet creation window, type a subnet name (subnet-user-X-1), for network address type 20.X.1.0/24, for Gateway IP type 20.X.1.1, where X is your user number, click on "Next" and click on "Create"
6. Create another network with the information below following the same process as on step 5 above:
   - **Network name**: net-user-X-2, **subnet name**: subnet-user-X-2, **network address**: 20.X.2.0/24, gateway IP: 20.X.2.1

LAB (CONT.)

7. Click on "Network Topology" on the left pane, you should see the new networks added to the topology
8. Click on "Compute" on the left pane, then click on "Instances"
9. Click on "Launch Instance"
10. On the instance creation window,:
    - Type a VM instance name (instance-user-X-0), click on "Next"
    - As image choose "cirros-0.3.4-x86_64-uec" by clicking on the "+" next to it, and click on "Next"
    - As flavor, choose "m1.nano" by clicking on the "+" next to it, and click on "Next"
    - As networks, choose the first network created on step 4 (net-user-X-1) and click on "Next"
    - Click on "Launch instance"
    - This creates an instance in the first network created. As soon as the power state of the instance changes to "Running", the instance is available for use.

LAB (CONT.)

11. Launch another instance in the same network, with the same properties following the process on step 10.
12. Click on "Network Topology", and you should see the instances you just created attached to the first network created.
13. Click on the first VM instance created (instance-user-X-0), click on "Open Console" to open the command line interface of instance
14. On the console, type "cirros" as username and "cubswin:)" as password to login to the instance
15. Type "ping IPx", where IPx is the IP of the second instance created in the first network (instance-user-X-1). You should see reply messages printed on the console, which means the two instances you created are communicating, as they are in the same network. Type Ctrl+C to stop pinging.

LAB (CONT.)

16. On the instances page, click on "Launch Instance" and create a new instance (instance-user-X-2) following the process on step 10, in the second network created (net-user-X-2).
17. On the console of the first instance created (instance-user-X-0) type "ping IPy", where IPy is the IP of instance-user-X-2. This will not print any reply messages, as the two instances are in different networks not connected to each other.
18. On the "Network Topology" view, click on "Create Router".
19. Enter a router name (router-X) and click on "Create Router".
20. Click on the icon of the created router, and click on "Add Interface".
21. On the Add Interface window, select the first network created (net-user-X-1) from the Subnet dropdown and click on "Submit".

LAB (CONT.)

22. Click on the router's icon again, and add another interface for the other network (net-user-X-2). This creates a connection between the two networks created.
23. Go to the console of instance-user-X-0 and ping the IP of instance-user-X-2. This should print a series of messages now, as the networks have been connected through the created router.
24. Under Networks on the left pane, click on Firewalls.
25. Click on the "Firewall Policies" tab and "Add Policy".
26. Enter a policy name (firewall-policy-X) and click on "Add".
27. Click on the "Firewall Rules" tab.
28. Enter a firewall name (firewall-rule-X), select "ICMP" as the protocol and select "DENY" as the action and click on "Add". This creates a firewall rule that drops all ICMP packets.

LAB (CONT.)

29. Click on the "Firewall Policies" tab, and select "Insert Rule" option in the actions dropdown of the firewall policy.
30. Select the created rule and click on "Save Changes".
31. Click on the Firewalls tab and click on "Create Firewall".
32. On the Add Firewall page, enter a name for the firewall (firewall-X), and select the firewall policy just created. Then click on the "Routers" tab and select the created router from the "Available Routers" list by clicking on the "+" sign next to it. Then click on "Add". This associates the firewall we just created with the router connecting the two networks.
33. Go to the console of instance-user-X-0 and ping the IP of instance-user-X-2. This will not print any reply messages, as all ICMP packets going through the router will be dropped.

LAB (CONT.)

34. Click on Firewalls, and delete the created firewall using the dropdown menu of the created firewall.
35. Under "Networks", click on "Load Balancers"
36. Click on the "Pools" tab, then click on "Add Pool".
37. On the Add Pool page, enter a name for the pool (pool-X), select the first network (subnet-user-X-1) from the "Subnet" dropdown, select "HTTP" as the protocol, and "Round Robin" as the load balancing method. Then click on "Add".
38. Click on the "Members" tab, then click on "Add Member". Select the pool just created in the Pool dropdown. From the member instances list, select the two instances in the first network (instance-user-X-0 and instance-user-X-1). Type 80 as the protocol port and click on "Add". You should now see the two instances in the members list.

LAB (CONT.)

39. Click on the "Monitors" tab and click on "Add Monitor".
40. On the Add Monitor page, select "HTTP" as type, type 3 for delay, timeout and max tries and click on "Add".
41. Click on Pools tab, and select "Add VIP" from the dropdown menu of the pool created.
42. On the Add VIP window, enter a VIP name (VIP-X), select the first subnet (subnet-user-X-1) from the VIP Subnet dropdown, enter 80 as the Protocol Port, select "HTTP" as the Protocol and click on "Add". This allocates an IP address for the load balancer.
43. Click on "Associate Monitor" in the dropdown menu of the the pool, select the monitor created and click on "Associate Monitor". This associates the health monitor with the load balancer we just created.

LAB (CONT.)

44. Go to the console of instance-user-X-0 and type the following command to create a very simple HTTP server on port 80 of the instance using netcat:

while true; do echo -e "HTTP/1.1 200 OK\r\nContent-Length: 8\r\n\r\nserver1" | sudo nc -l –p 80 ; done

45. Repeat step 44 for instance-user-X-1, replacing *server1* with *server2*. We should see some HTTP get requests being printed on both instances' consoles due to the health monitor check in action.

46. Go to the console of instance-user-X-2 and type "wget VIPx", where VIPx is the IP of the load balancer. This will create an index.html file with the text either "server1" or "server2" in it. If you repeat this a few times (deleting index.html each time), you will see that the server1 and server2 alternate, which means the load balancer is in action.