Cloud Networking (VITMMA02)
Software Defined Networking (SDN) in the Cloud

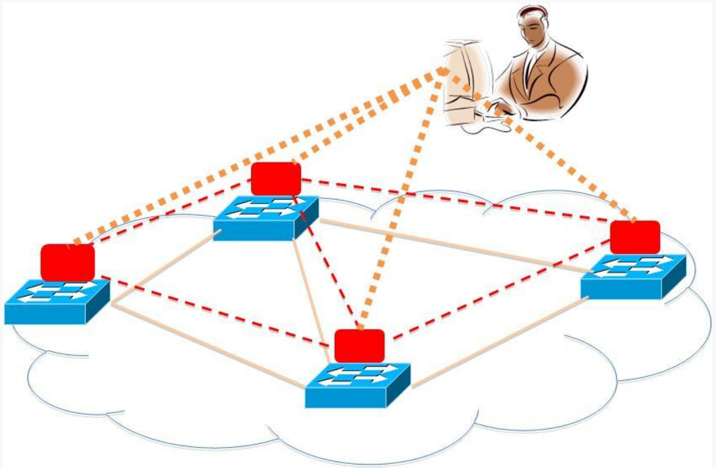
Markosz Maliosz PhD

Department of Telecommunications and Media Informatics
Faculty of Electrical Engineering and Informatics
Budapest University of Technology and Economics

Spring 2017

Traditional Computer Network

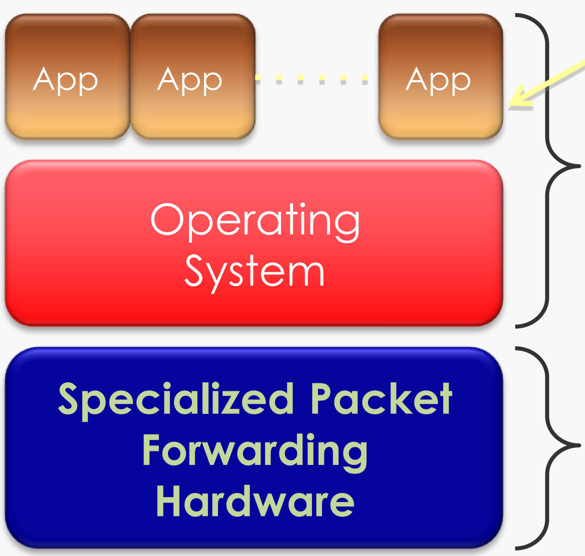
- Data plane: wire-speed time scale (fast)
  - packet handling: Forward, filter, buffer, mark, rate-limit, and measure packets
- Control plane: slower time scale (per control event)
  - distributed algorithms
  - tracking topology changes, computing routes, installing forwarding rules
- Management plane: human time scale
  - centralized
  - collecting measurements and configuring the equipment



Traditional Network (Device) Architecture

- Networks used to be simple: Ethernet, IP, TCP….
- New **control** requirements led to great complexity
  - Isolation → VLANs, ACLs
  - Traffic engineering → MPLS, ECMP, Weights
  - Packet processing → Firewalls, NATs, middleboxes
  - Payload analysis → Deep packet inspection (DPI)
  - …..
- Many complex functions built into the infrastructure
  - OSPF, BGP, multicast, differentiated services, Traffic Engineering, NAT, firewalls, MPLS, …
- An industry with a “mainframe-mentality” – monolithic

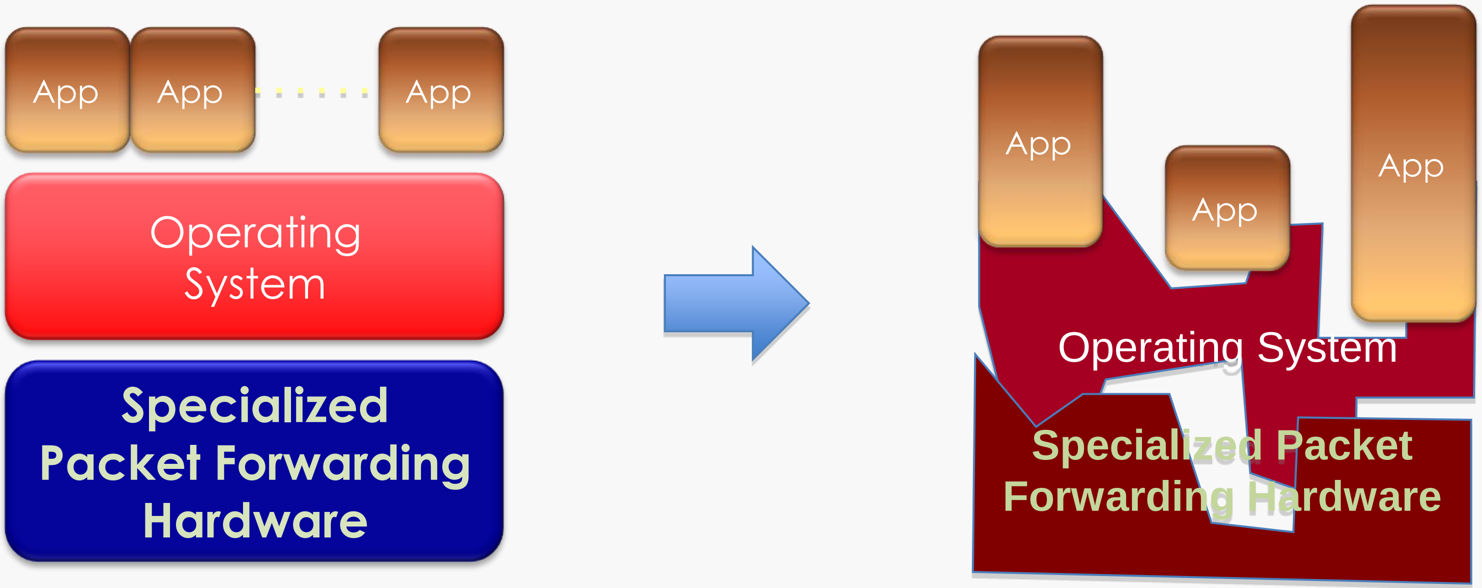
Routing, management, mobility management, access control, VPNs, …



Million of lines of source code | 5400 RFCs | Barrier to entry

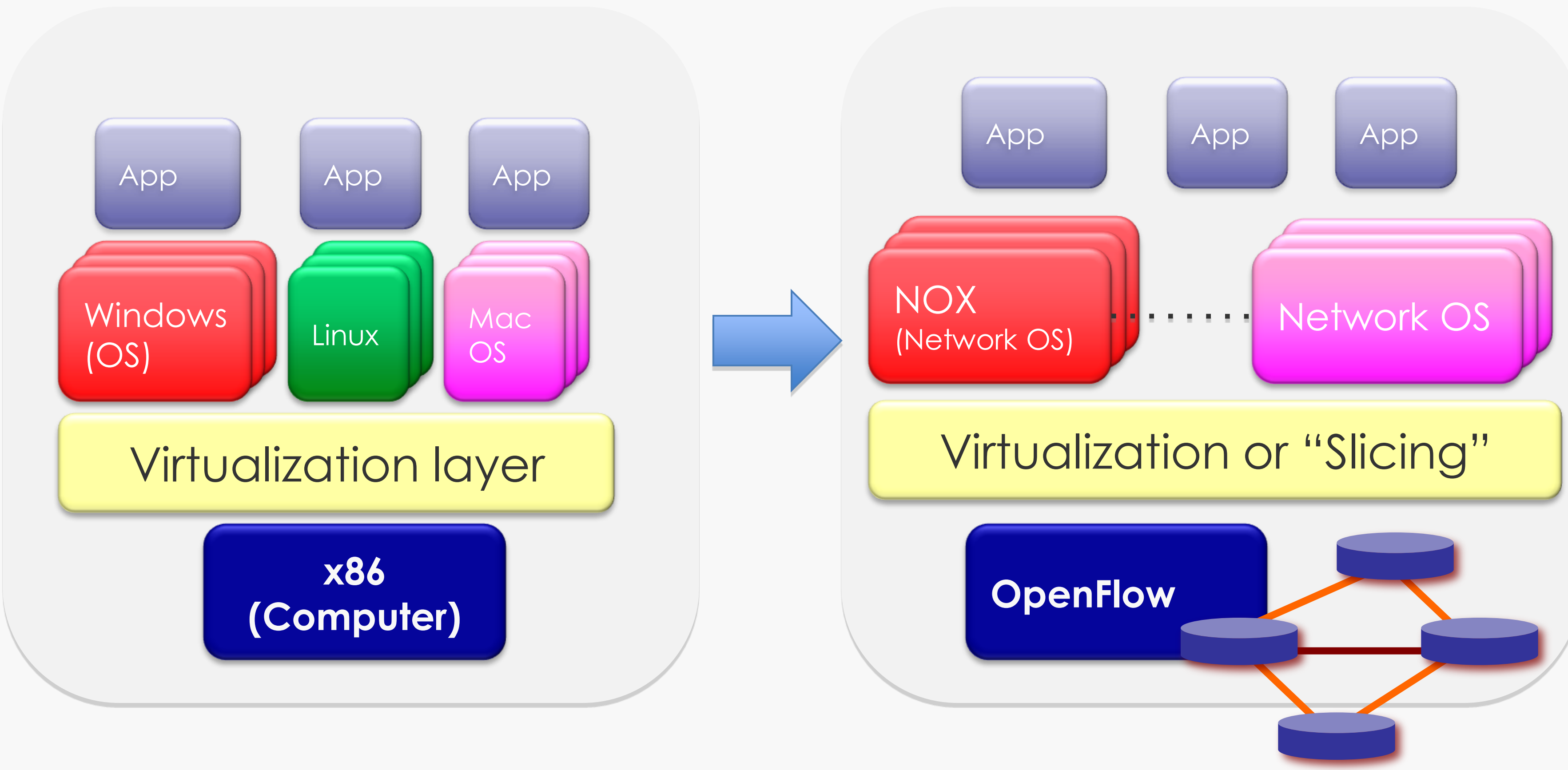
500M gates 10Gbytes RAM | Complex | Power Hungry

Ideal vs. Real Architecture



- Lack of competition hinders innovation
  - few people can innovate
  - slow standardization process
- Closed architecture means blurry, closed interfaces
  - software bundled with hardware
- Vertically integrated, complex, closed, proprietary
  - vendor specific interfaces
- Not suitable for experimental ideas
- Not good for network owners & users, researchers

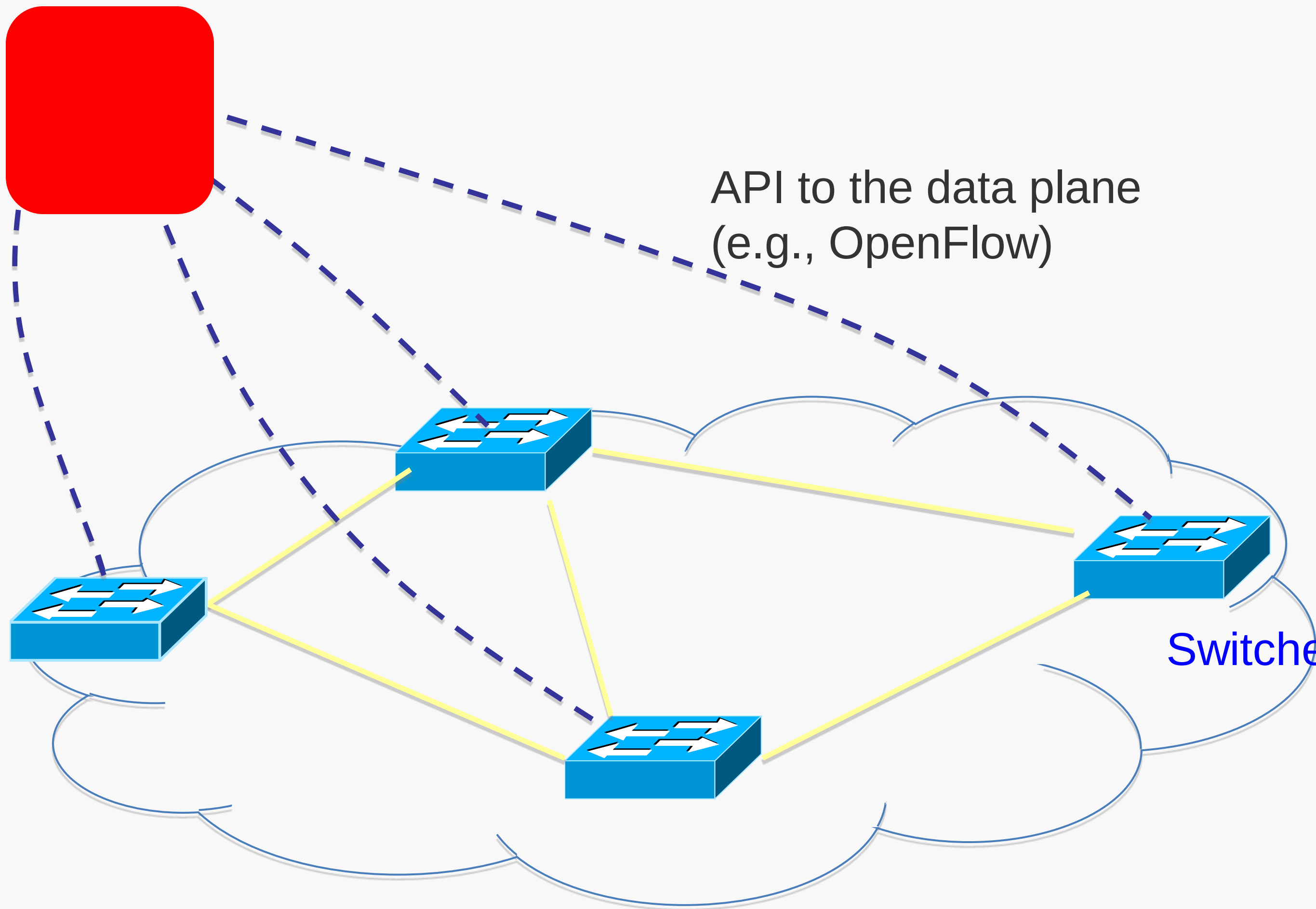
Similarities



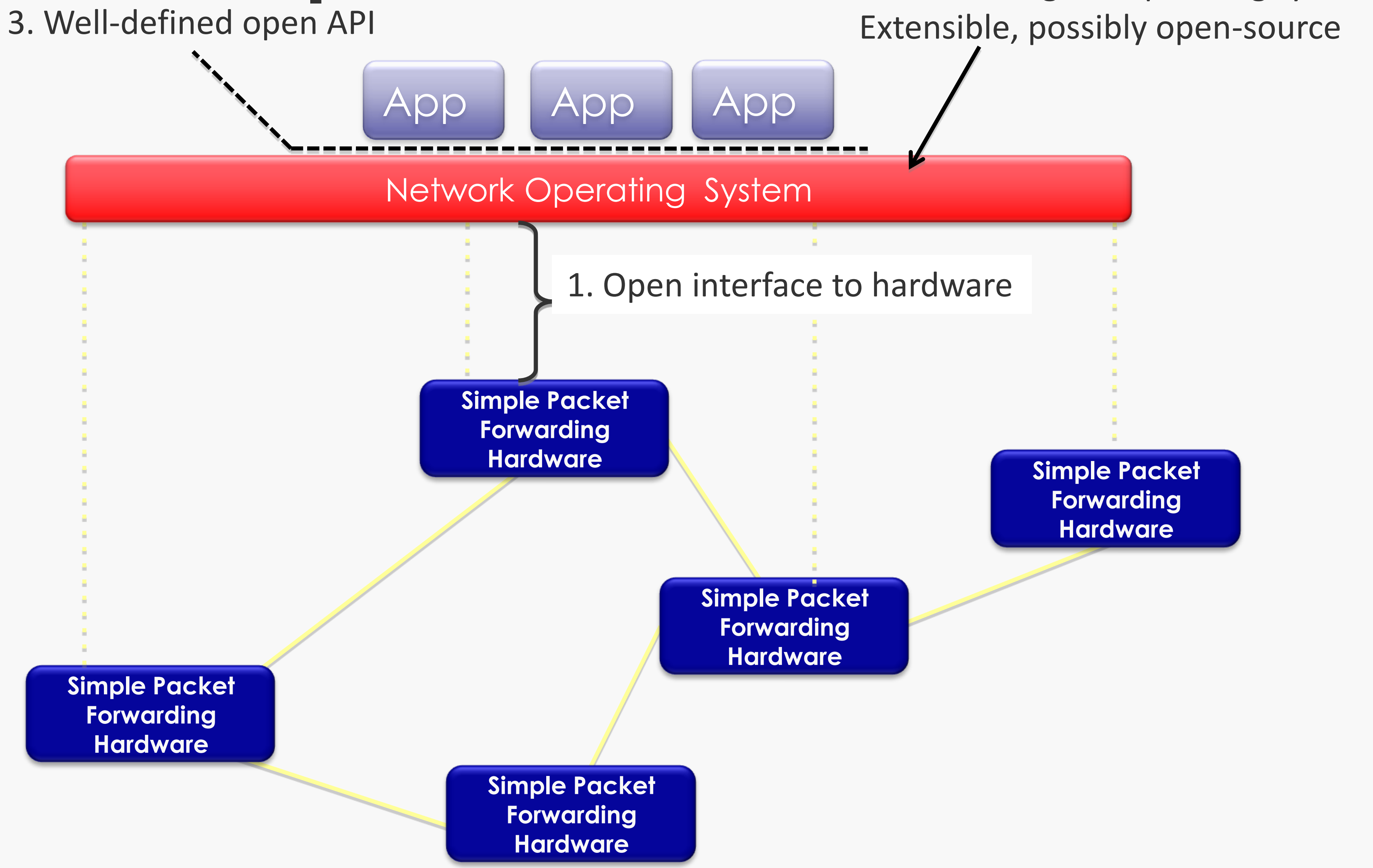
Computer Industry

Network Industry

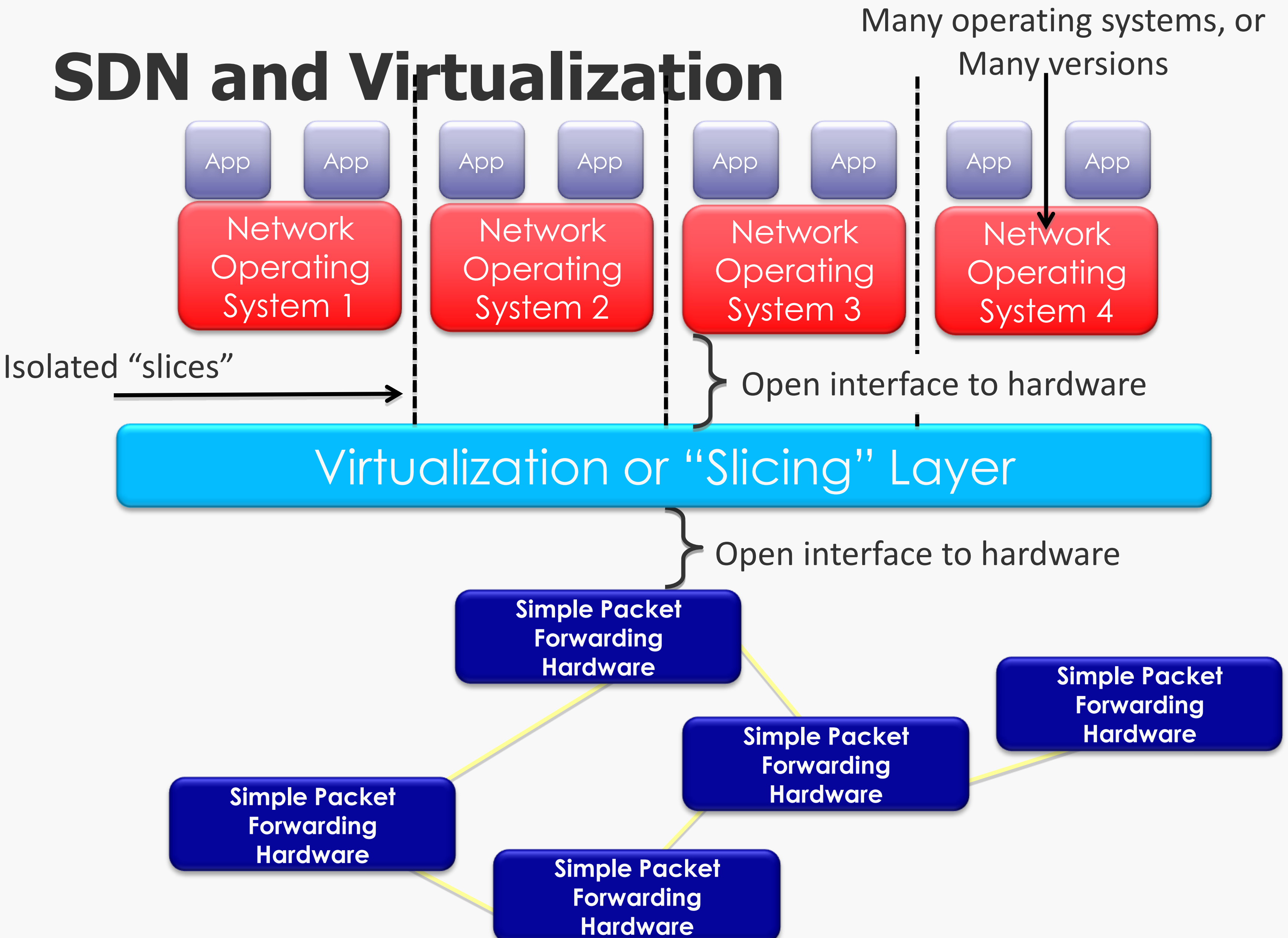
Software Defined Networking (SDN)



SDN Components

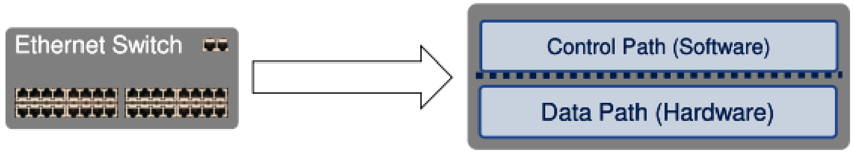


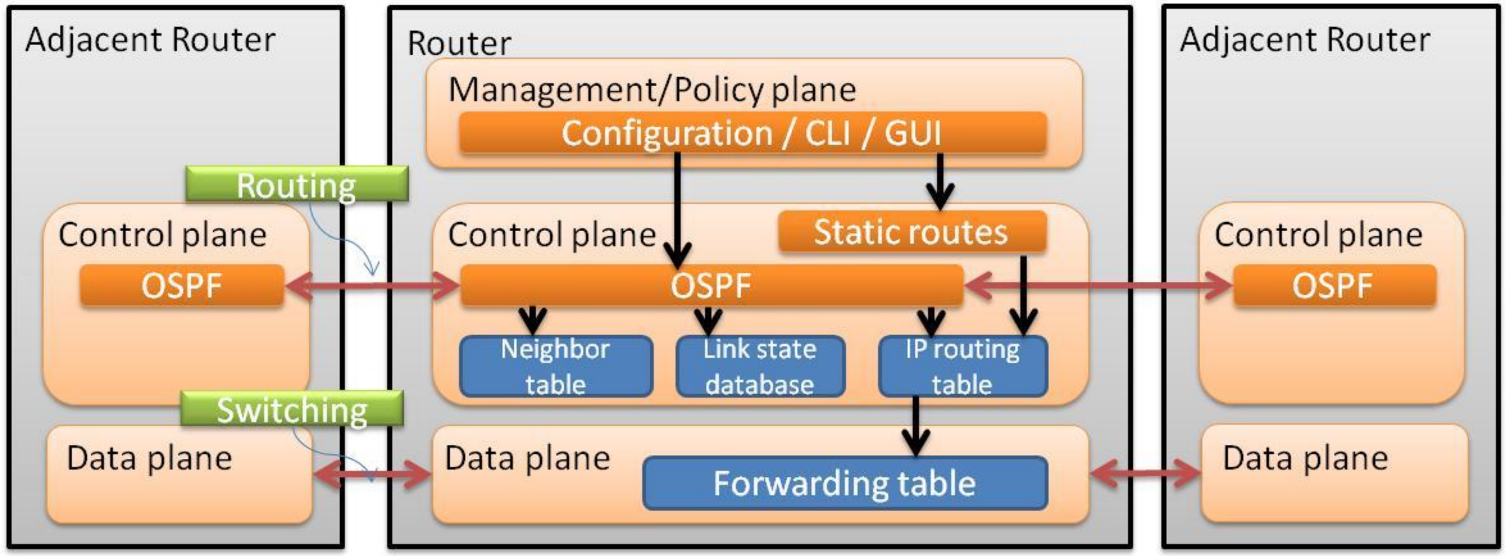
SDN and Virtualization



Traditional Switch/Router

- Operations can be partitioned into planes
  - Management plane / Configuration
  - Control plane / Decisions
  - Data plane / Forwarding





Concept of SDN

- Separate Control plane and Data plane entities
  - Network intelligence and state are logically centralized
  - The underlying network infrastructure is *abstracted* from the applications
- Execute or run Control plane software on general purpose hardware
  - Decouple from specific networking hardware
  - Use commodity servers
- Have programmable data planes
  - Maintain, control and program data plane state from a central entity
- An architecture to control not just a networking device but an entire network

Control Software Program

Control program operates on view of network

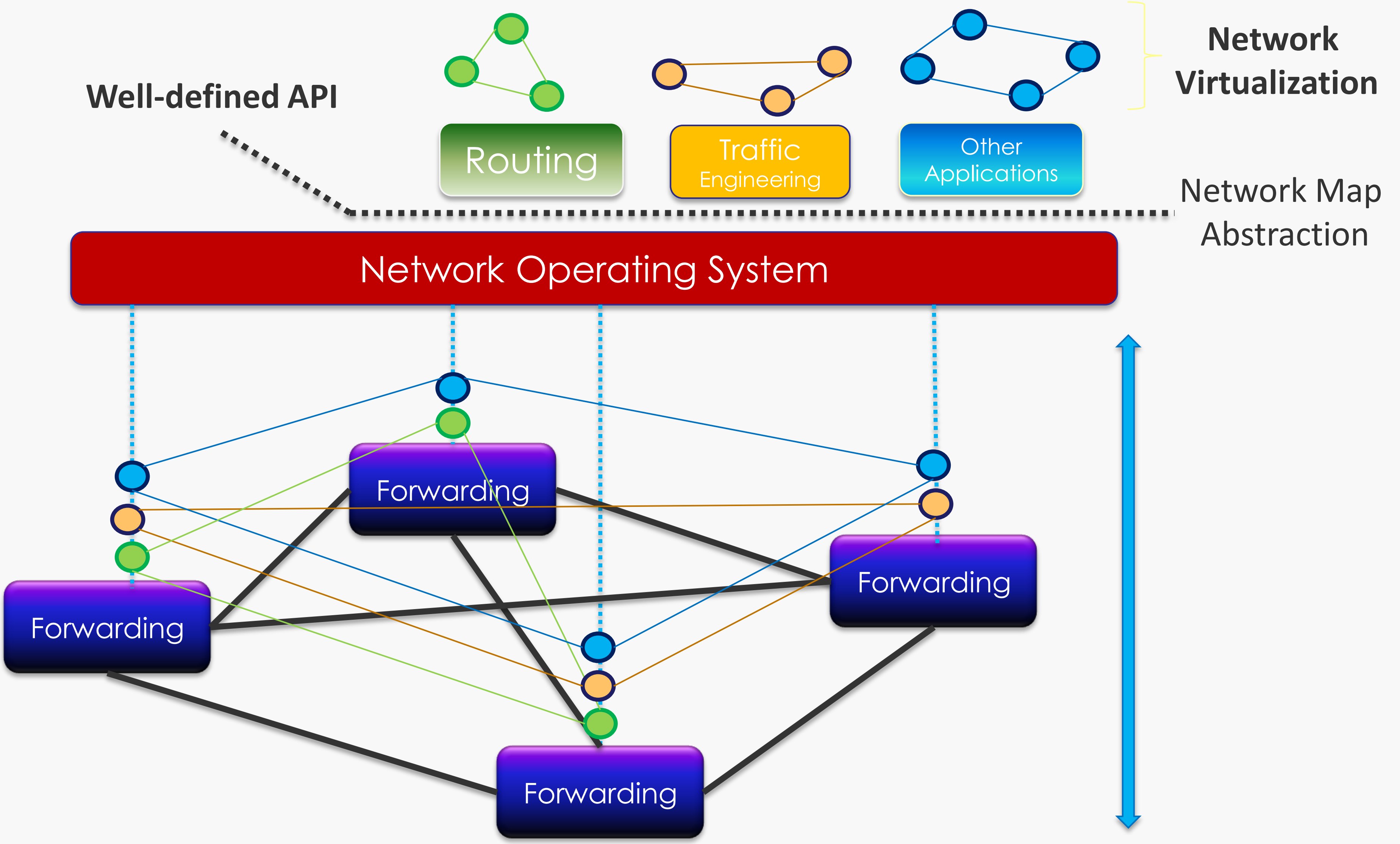
- **Input**: global network view (graph/database)
  - Annotated network graph provided through an API
  - Receives events from switches
    - Topology changes
    - Traffic statistics
    - Arriving packets

- **Output**: configuration of each network device
  - Control mechanism is a program, implementing e.g. a graph algorithm
  - Sends commands to switches
    - (Un)install rules
    - Query statistics
    - Send packets

Control program is **not** a distributed system

- Abstraction hides details of distributed state

SDN with Abstractions in the Control Plane



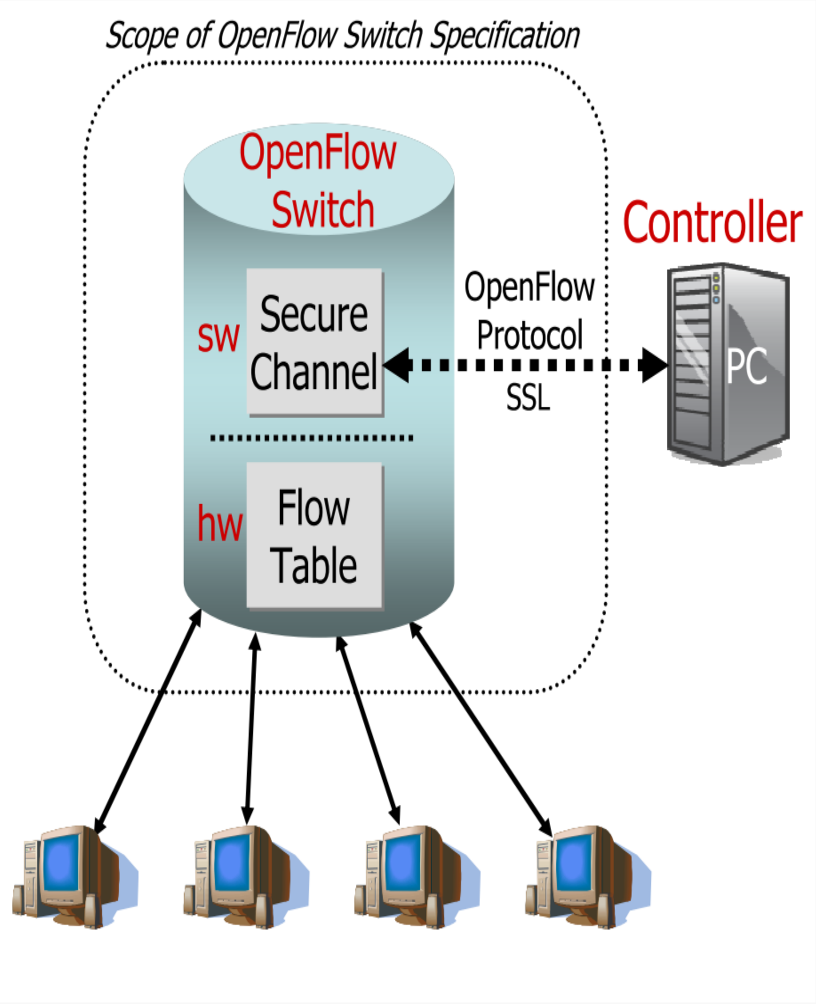
Forwarding Abstraction

- Purpose: Abstract away forwarding hardware
- Flexible
  - Behavior specified by control plane
  - Built from basic set of forwarding primitives
- Minimal
  - Streamlined for speed and low-power
  - Control program not vendor-specific

- OpenFlow is an example way of such an abstraction

What is OpenFlow?

- Provides open interface to “black box” networking node
  - (i.e. Routers, L2/L3 switch) to enable visibility and openness in network
- Separation of control plane and data plane
  - The datapath of an OpenFlow Switch consists of a Flow Table, and an action associated with each flow entry
  - The control path consists of a controller which programs the flow entry in the flow table
- OpenFlow is based on an Ethernet switch, with an internal flow-table, and a standardized interface to add and remove flow entries



OpenFlow Devices

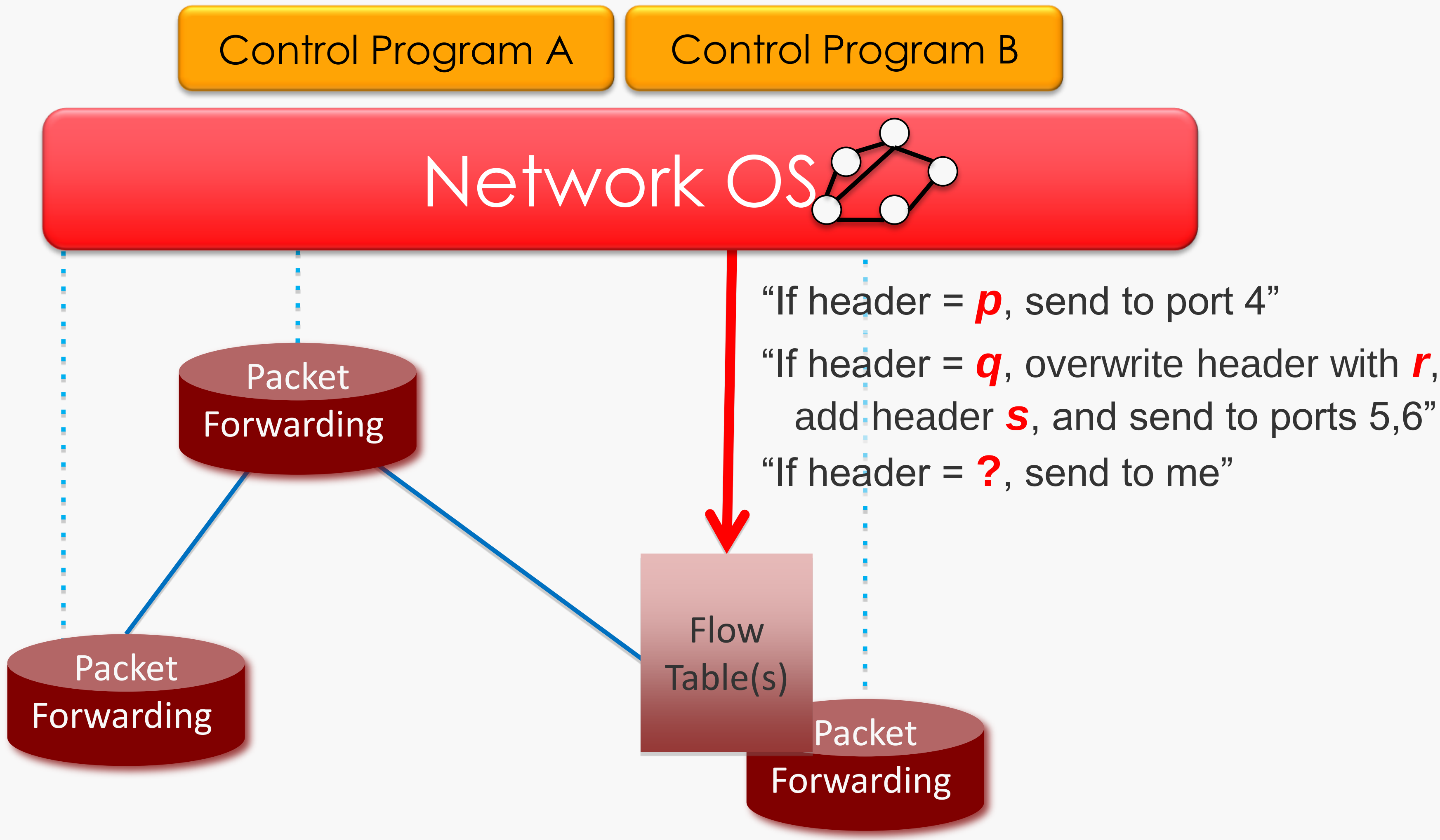
Controller/NOS

- POX: (Python)
  - general SDN controller
  - features: queriable topology graph and support for virtualization
- NOX: (C++)
  - was the first OpenFlow controller
- IRIS: (Java)
  - features : Horizontal Scalability for carrier-grade network; High Availability with transparent failover from failure; (Multi-domain support with recursive network abstraction based on Openflow
- Beacon: (Java)
  - event-based and threaded operation
- Floodlight: (Java)
  - enterprise level
- OpenDaylight (Java)
  - NFV
- Ryu: (Python)
  - an open-sourced Network Operating System (NOS)
- NodeFlow (JavaScript)
  - an OpenFlow controller written in pure JavaScript for Node.JS
- ovs-controller (C)
  - Trivial reference controller packaged with Open vSwitch

Switches

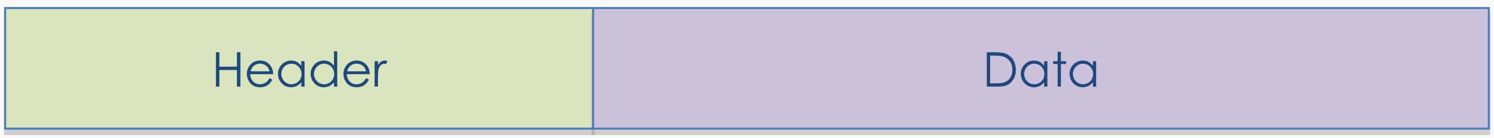
- Software Switches
  - Stanford Reference Implementation v1.0
  - Ericsson implementation v1.1 & v1.2
    - Linux-based Software Switch running in User Space
  - Open vSwitch
    - Linux-based Software Switch running in Kernel Space
    - Not just an OF switch, widely used by virtual machines (VirtualBox, XEN)
    - Firmware of some devices based on Open vSwitch
  - OpenFlow 1.3 Software Switch
    - CPqD in technical collaboration with Ericsson Research, Brazil
- Software → Hardware
  - Commercial off-the-shelf (COTS) devices
    - running OpenWRT
    - software switches can be ported
    - run by CPU
    - in user-space
  - NetFPGA-based implementation
- Hardware vendors
  - HP, Cisco, Juniper, IBM, Arista, NEC, Netgear, Pronto, …

OpenFlow Basics



OF Primitives: <Header match, Action>

- Simple packet-handling rules
- Match arbitrary bits in headers:



Match: 1000x01xx0101001x

  - Match on any header, or new header
  - Allows any flow granularity
- Action
  - Forward to port(s), drop, send to controller
  - Overwrite header with mask, push or pop
  - Forward at specific bit-rate

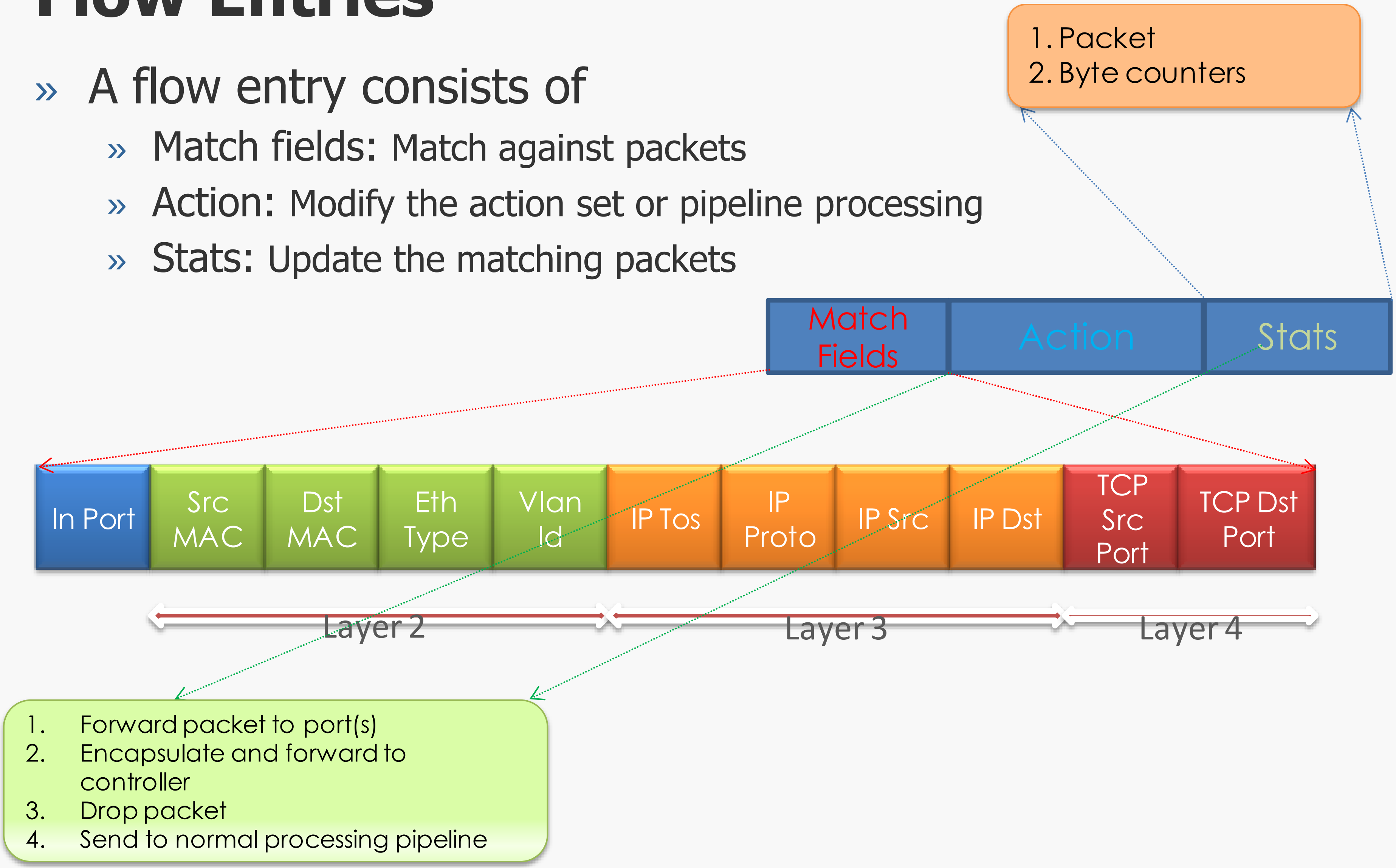
Flow Table

» Flow table in switches, routers, and chipsets

Flow 1.	Rule (exact & wildcard)	Action	Statistics
Flow 2.	Rule (exact & wildcard)	Action	Statistics
Flow 3.	Rule (exact & wildcard)	Action	Statistics
	....		
Flow N.	Rule (exact & wildcard)	Default Action	Statistics

Flow Entries

- A flow entry consists of
  - Match fields: Match against packets
  - Action: Modify the action set or pipeline processing
  - Stats: Update the matching packets



Examples (1/2)

Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f:..	*	*	*	*	*	*	*	port6

Flow Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
port3	00:20..	00:1f..	0800	vlan1	1.2.3.4	5.6.7.8	4	17264	80	port6

Firewall

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	*	*	*	22	drop

Examples (2/2)

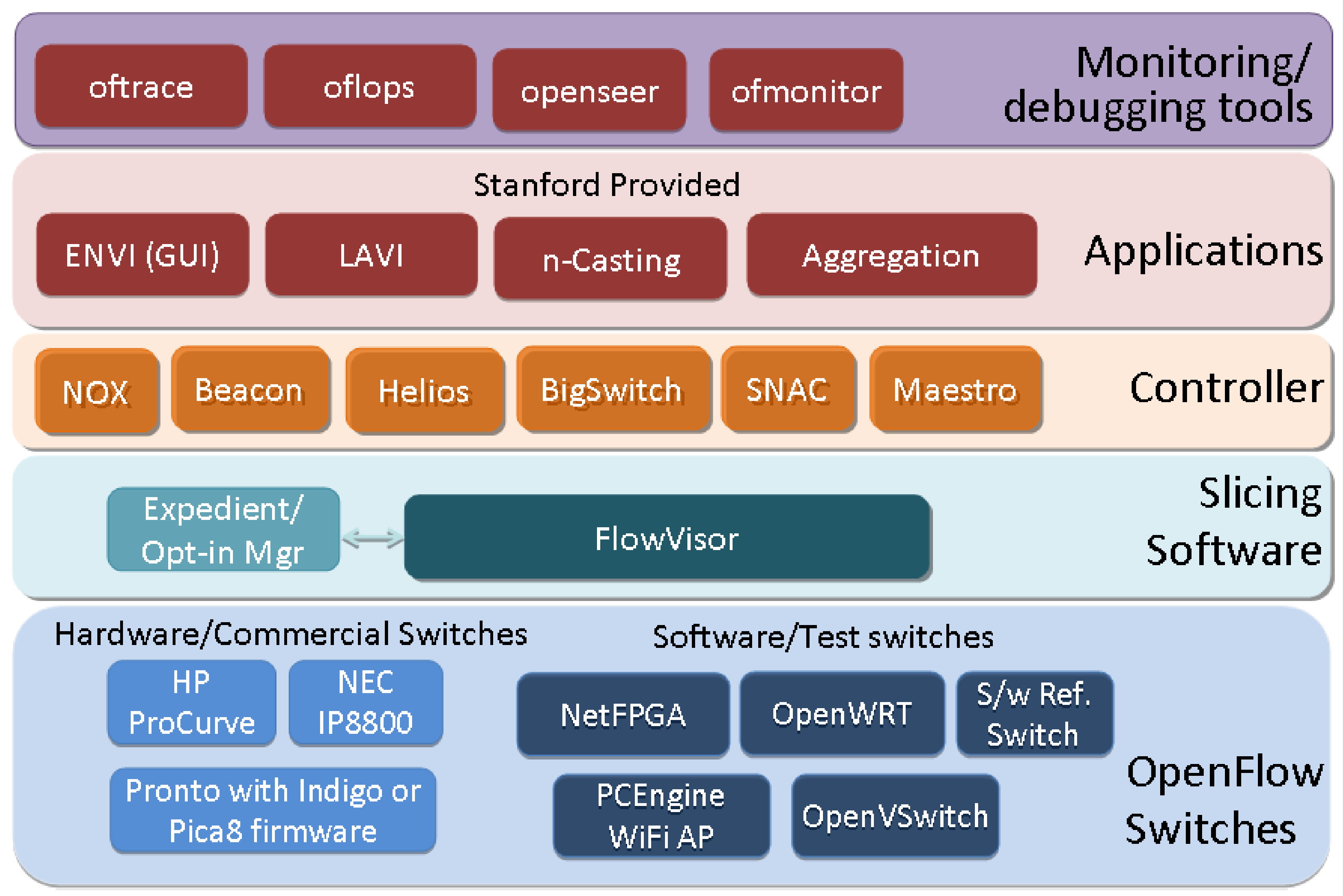
Routing

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	5.6.7.8	*	*	*	port6

VLAN Switching

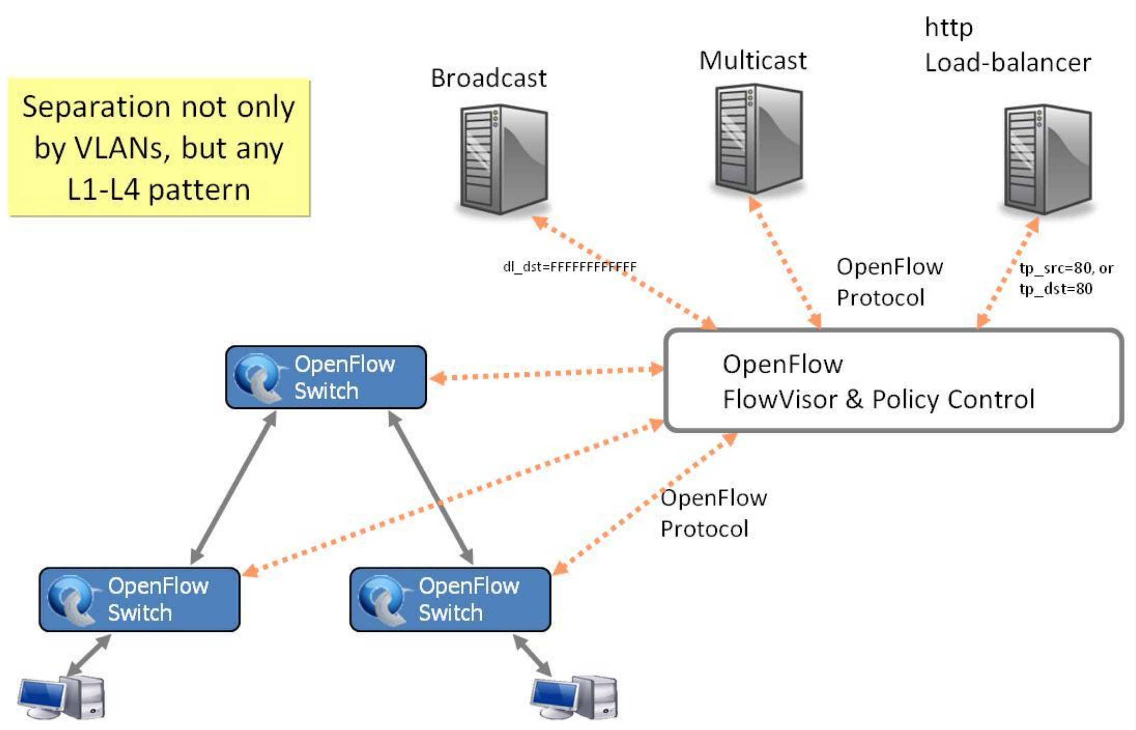
Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f..	*	vlan1	*	*	*	*	*	port6, port7, port9

OpenFlow Building Blocks



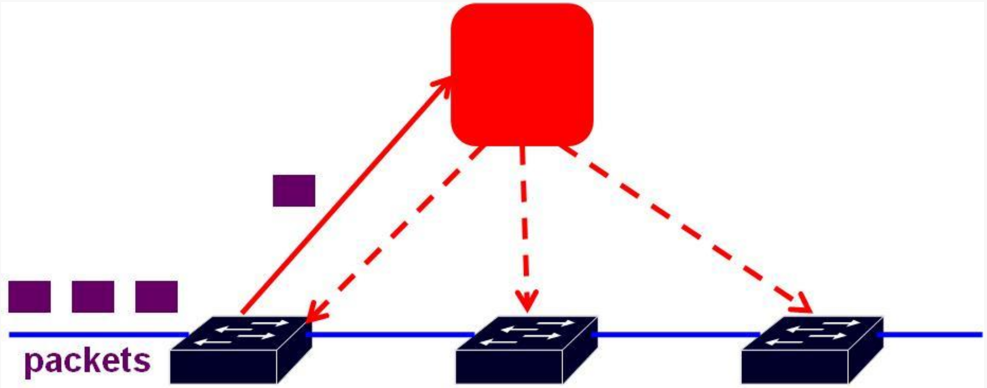
Slicing

- FlowVisor
  - software proxy between the forwarding and control planes of network devices
  - it assigns hardware resources to “slices”
  - topology discovery is per slice
- Separate VLANs for Production and Research Traffic



Reactive operation

- Packets are managed as flows
  - The 1st packet of a flow is sent to the controller
  - The controller programs the actions of datapath for a flow
    - Usually one rule, but may be a list
    - Actions include: Forward to a port or ports, Mirror, Encapsulate and forward to controller, Drop
  - And returns the packet to the datapath
  - Subsequent packets are handled directly by the datapath

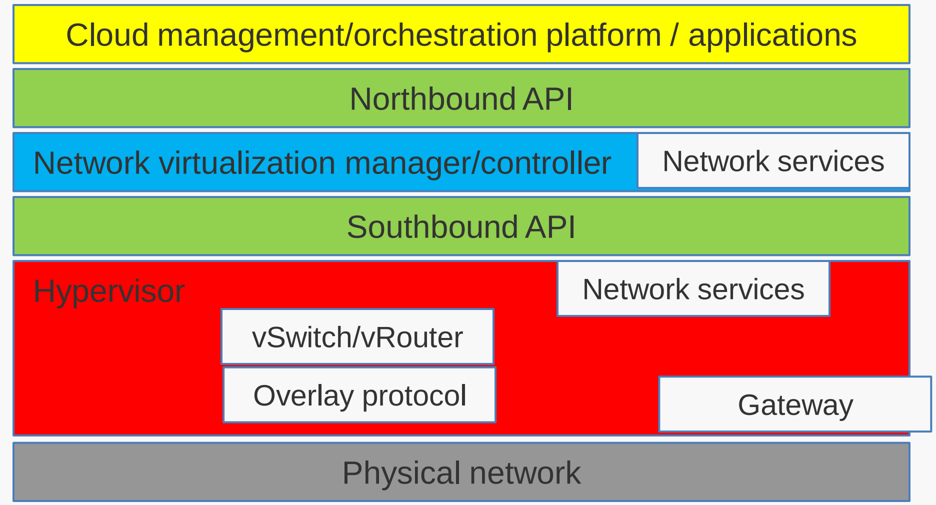


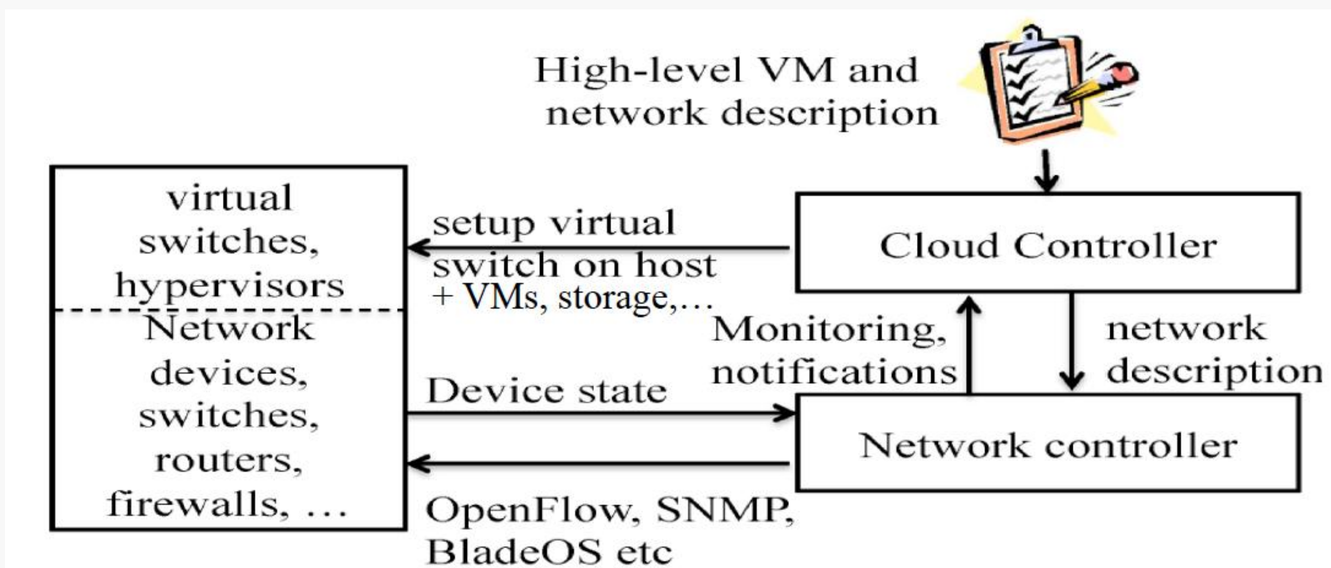
SDN in the Cloud

- Instead of reactive operation…
  - 1st packet sent to controller ⇨ delay
  - end-to-end: many rule entries, scalability problem
  - tenant and VM changes affect all physical switches
- …pro-active operation with overlay networks
  - physical network provides L2/L3 connectivity
  - controller pre-programs devices in advance ⇨ low delay
  - tunnels: tenant state only in endpoints (servers: hypervisor, virtual switch/router), scalable
  - less entries in forwarding tables
    - not for each VM, but only for physical servers
  - tenant and VM changes do not affect physical switches

Cloud Management and SDN

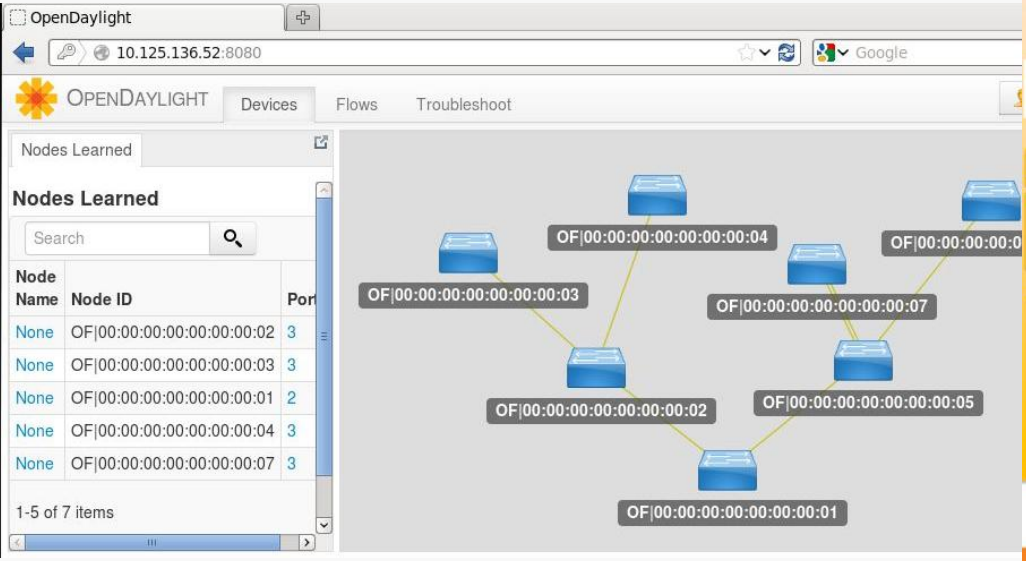
- Orchestration functions by OpenStack
  - higher level abstraction
    - deals with virtual resources
  - not only for network, but for a whole application system
    - VMs, storage, etc. + network
  - CLI or horizon dashboard
  - automation: Heat templates
- SDN
  - lower level network realization of the above

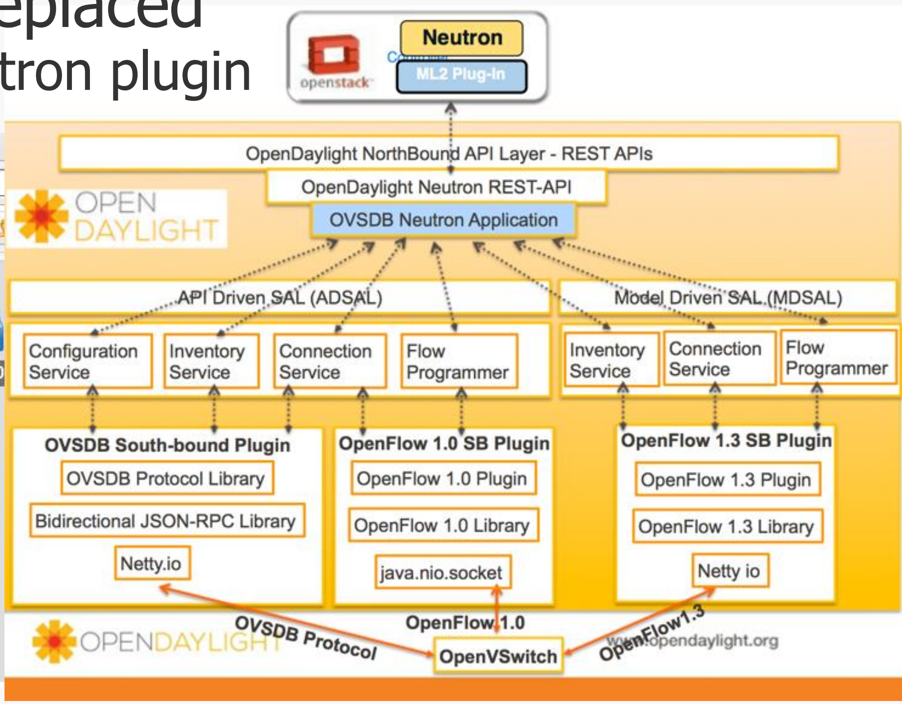




OpenStack

- OVS Neutron plugin
  - OpenFlow for programming virtual switch tables
    - mapping VM MAC address and server hypervisor transport IP address – known by the orchestration
    - proactive
    - northbound interface: Neutron
    - southbound interface: OpenFlow
- SDN controller plugins can be replaced
  - e.g. OpenDaylight OpenStack Neutron plugin





SDN in the Cloud

» Not only for virtual switches/routers, but also for physical network devices

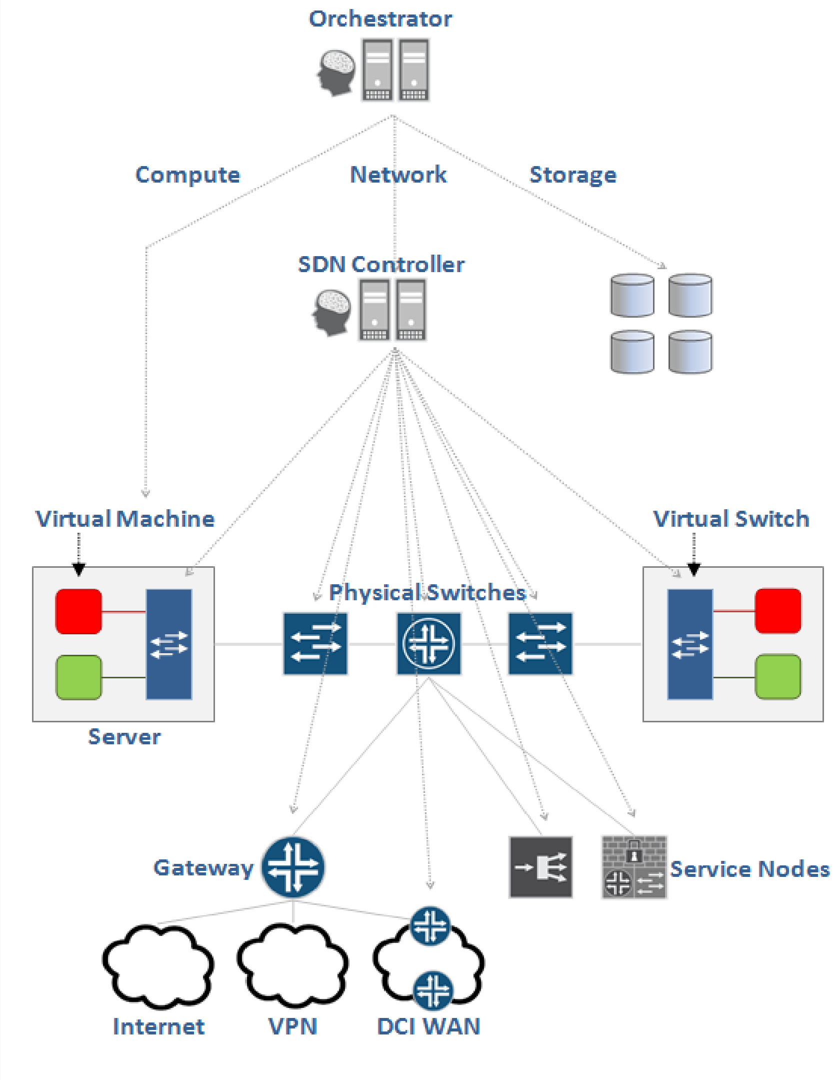


Figure 22: The Role of Orchestration in the Data Center

Source: http://www.opencontrail.org/opencontrail-architecture-documentation/

Data Center Network Requirements

- Minimizing the configuration and stored states of network devices
  - automation, as much as possible
- Effective traffic forwarding, high performance
  - no loops
  - adaptation to traffic changes
  - meet tenant SLA
- Quick and easy VM migration
  - transparent migration
- Fast and effective fault detection/recovery
  - quite frequent because of the large number of elements
  - network must adjusted to the fault recovery

Traditional networking solutions

- Layer 3
  - + hierarchical addressing ⇨ small forwarding tables
  - + OSPF fast fault handling
  - + IP TTL: to prevent loops
  - - high administration burden (to configure sub-networks, DHCP, etc.)
- Layer2
  - + Flat MAC addressing (locality independent)
  - + to prevent loops: STP
  - + less administration burden
  - - broadcast traffic (not very scalable)
  - - STP: unused links in the topology
- VLAN
  - scalalbility limit (max. 4K)
  - disadvantages from static configuration

Networking with SDN

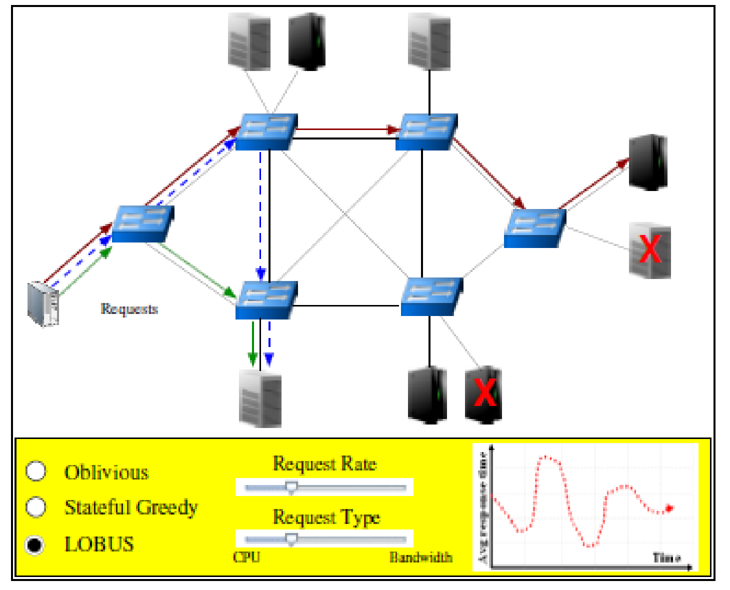
- controller is aware about the whole network
  - device discovery
  - MAC, IP addresses, connections
- realizing the network on a lower level according to orchestration tasks
- quick and dynamic network provisioning
  - flexible: tenant self-service
  - automated network resource allocation and management
  - optimizing traffic, even between data centers
- scalable
- NFV

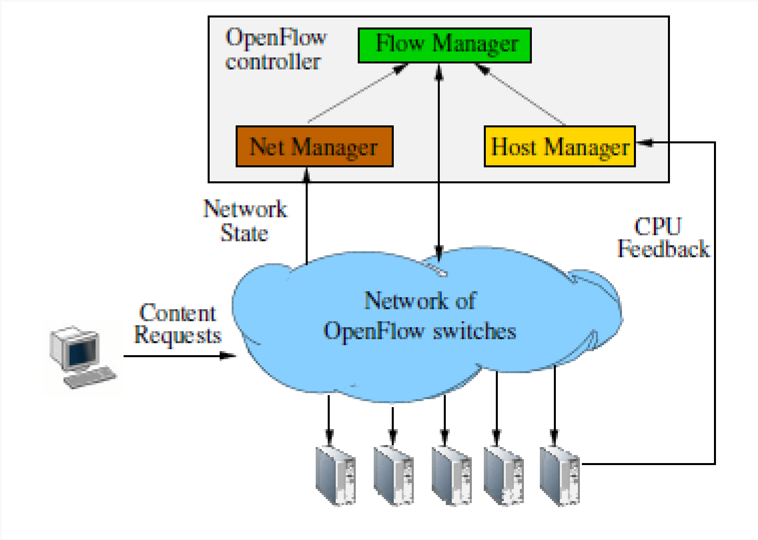
Examples of cloud specific tasks

- Load Balancing – LB
- inter-DC tunnel
- VM migration
- scalable packet forwarding

Load Balancing

- Dynamism
  - timer for OpenFlow rule entries
- Required operations for Load Balancing
  - rewrite public IP to server IP
  - forwarding to server output port
  - the opposite operations in the backward direction
- To do
  - hash based routing
  - TCP flag checking to identify new flows
- Plug-n-Serve: Load-Balancing Web Traffic using OpenFlow
  - Load balancing according to network and server loads in a distributed way

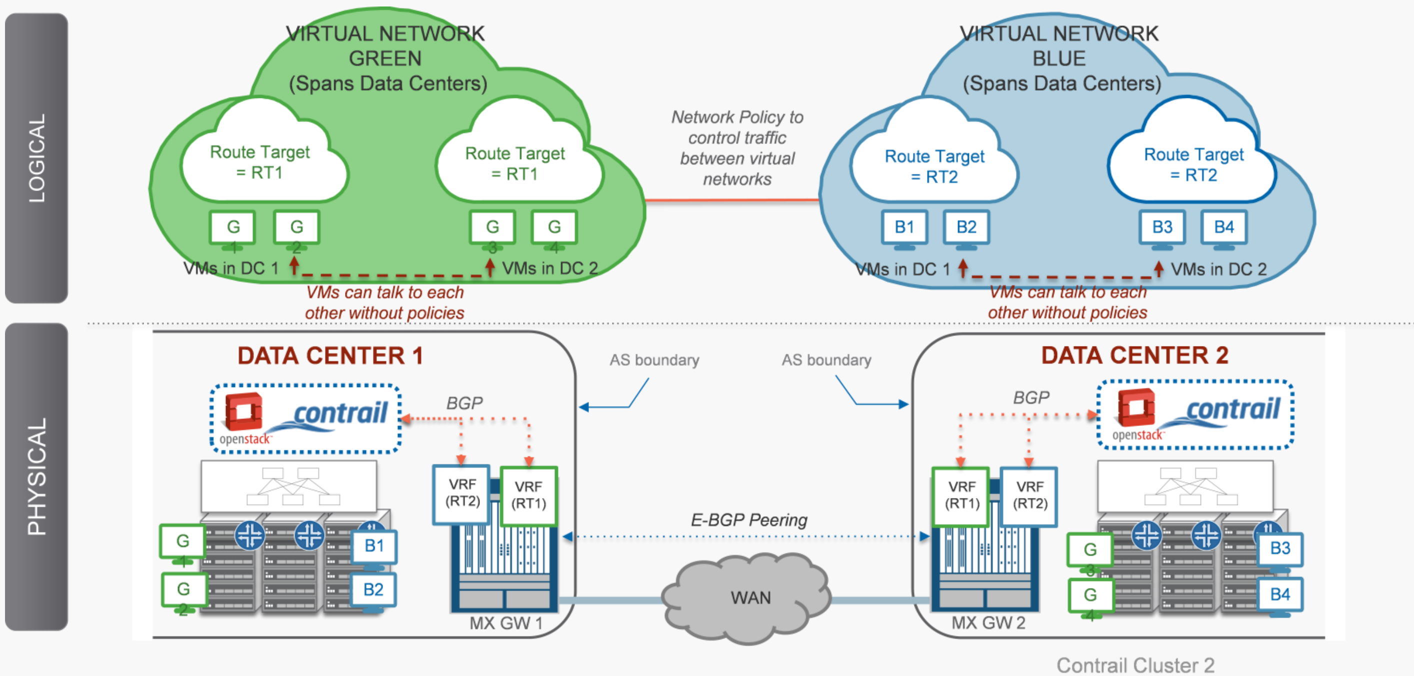




Source: http://conferences.sigcomm.org/sigcomm/2009/demos/sigcomm-pd-2009-final26.pdf

SDN for inter-DC traffic

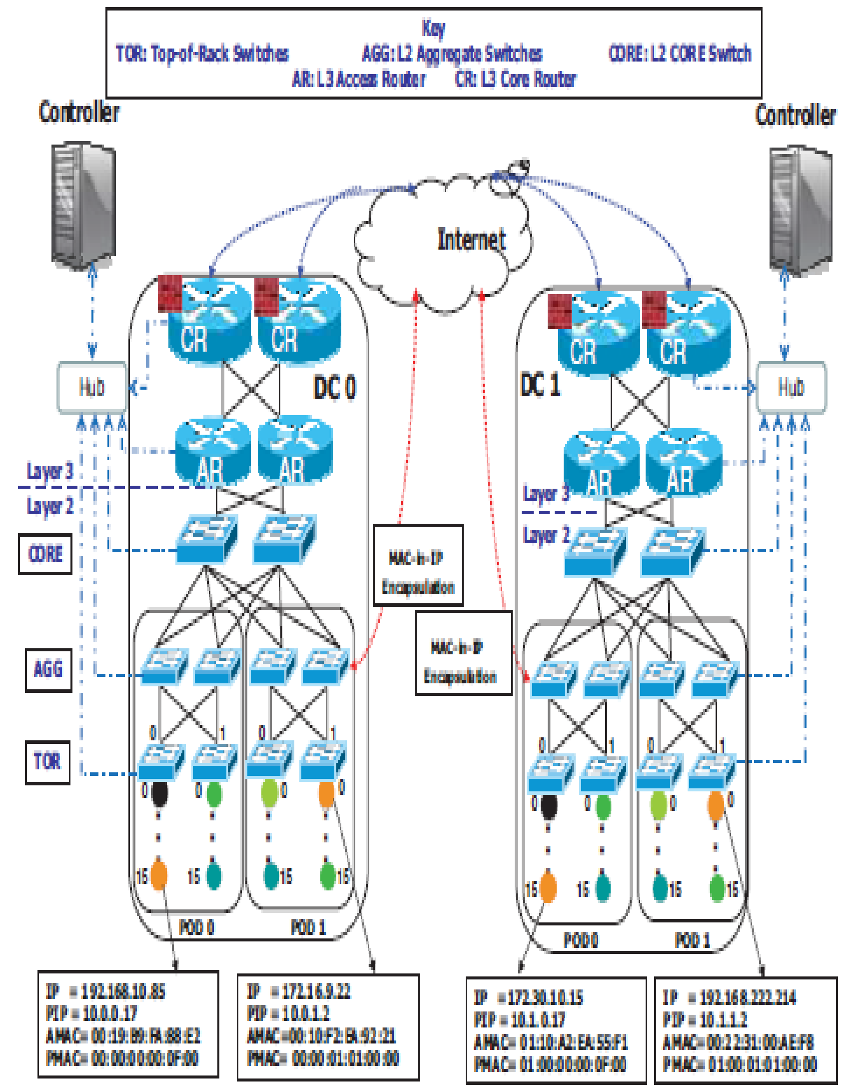
- Traffic
  - cloud bursting
  - geographical aspects in load balancing
- Tunnel provisioning with reactive operation
  - multipath
  - changes in paths = reprogramming packet headers on-the-fly



Source: http://www.opencontrail.org/how-to-setup-opencontrail-gateway-juniper-mx-cisco-asr-and-software-gw/

VM migration

- Reasons
  - maintenance, load balancing
  - VM consolidation (energy savings)
  - disaster recovery: migrating full application stacks
- Difficulties of migration to another subnet
  - hierarchical IP addressing
  - manual reconfiguration is not viable
  - without disrupting live TCP connections
- CrossRoads
  - locality independence: pseudo MAC (PMAC) and IP addresses (PIP)
  - SDN controlles manages the mapping



Source: Mann, V.; Vishnoi, A; Kannan, K.; Kalyanaraman, S., "CrossRoads: Seamless VM mobility across data centers through software defined networking," *Network Operations and Management Symposium (NOMS), 2012 IEEE , vol., no., pp.88,96, 16-20 April 2012*

SDN scalability

- A challenge for the control plane
  - number of VMs, tenant rules, SLAs, flows, etc.
- in multi domain environment: federation of controllers
  - information exchange
  - sharing states
  - easily extensible
- NEC tests from 2014
  - Trema OpenFlow controller
  - Layer 2 networks with VXLAN technology
  - controllers with load balancing
    - a controller manages 410 switches, scales linearly
  - running 16 000 virtual networks
    - 1024 switch, 128 VM on each
  - to provision a virtual network takes constant 4 sec

Deployments and Applications

- Amazon, Google, Facebook, Microsoft Azure
  - individual SDN solutions
- Google inter-datacenter WAN using SDN and OpenFlow
  - centralized traffic engineering
  - lowering network costs
- Data Centers provisioned by NEC
  - lowering network costs
- VMware
  - Nicira (SDN, network virtualization)
    - Network Virtualization Platform (NVP): overlay networking technology ⇨ VMware NSX

References

- Nick McKeown (Stanford University),"Software-defined Networking‟, Infocom Keynote Talk, April 2009, Rio de Janeiro, Brazil
- Srini Seetharaman, OpenFlow/SDN tutorial, Nov 2011
- Jennifer Rexford (Princeton University), Computer Science 461: Computer Networks, Software Defined Networking
- Matt Davy (Indiana University), Software Defined Networking & OpenFlow, GENI Workshop, July 7th, 2011
- Open Networking Foundation, https://www.opennetworking.org/
- http://www.openflow.org/
- CHIBA Yasunobu, SUGYOU Kazushi, „ OpenFlow Controller Architecture for Large-Scale SDN Networks”, NEC Technical Journal/Vol.8 No.2/Special Issue on SDN and Its Impact on Advanced ICT System, 2014