A Telecom Perspective on SDN and OpenFlow

Virgil Dobrota
Technical University of Cluj-Napoca, Romania
Virgil.Dobrota@com.utcluj.ro

Lecture at CUSO (Conférence Universitaire de Suisse Occidentale)
Doctoral School, University of Neuchâtel, Switzerland, 27 November 2015
http://informatique.cuso.ch/index.php?id=2283&no_cache=1&tx_displaycontroller

My background

Affiliation:

- IEEE Communications Society (since 1990)
- ACM SIGCOMM (since 1994)

Teaching:

- Switching & Routing Systems
- Internet Protocols
- Unified Communications

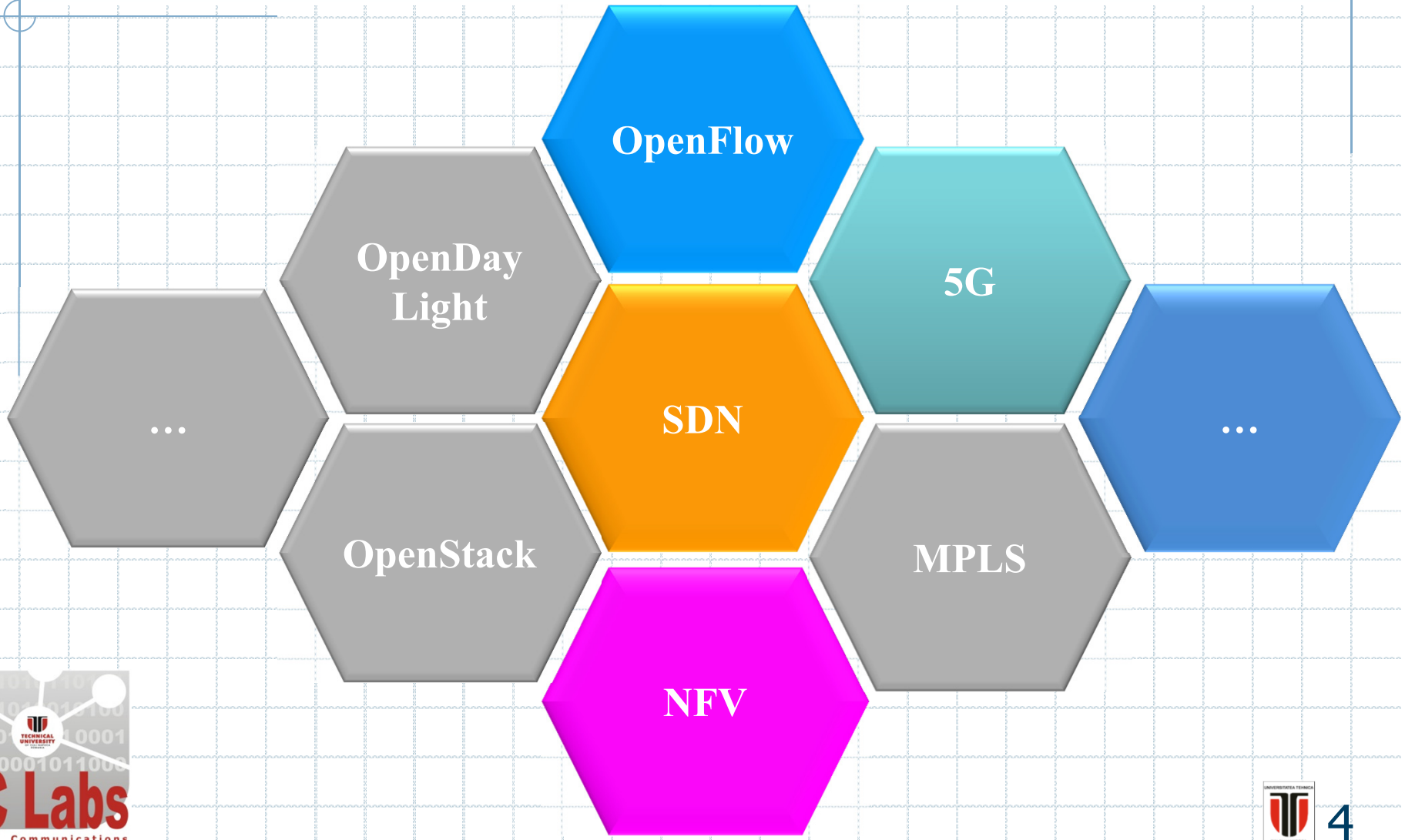
Research

- Cross-Layer QoS
- Seamless connectivity
- Passive/active measurements in Internet

Course Outline

1. Introduction
2. Legacy Networking Definitions
3. SDN
4. SDN and OpenFlow
5. SDN and OpenStack
6. SDN and NFV
7. SDN and OpenDayLight
8. SDN and MPLS
9. SDN and 5G
10. Practical Demonstration
11. Conclusions
12. References

SDN and OpenFlow, NFV, MPLS, etc.



2. Legacy Networking Definitions

Definitions 1/5

A network architecture has data, control and management planes.

Data Plane:

- Tasks: forwarding of the user traffic
- Devices: switches (usually Layer 2 devices), routers (Layer 3 devices)
- Examples of components:

Layer 2 and Layer 3 Ethernet switches with TCAM (Ternary Content-Addressable Memory) chips (dedicated ASIC/port or a single ASIC for all ports), a fast searching with 0,1 and X (Don't Care) matching conditions
TCAM=Source MAC (6 bytes)+Destination MAC (6 bytes)+Ethernet type (2 bytes)

IP routers with FIB (Forward Information Base), routing information from control plane organized for fast lookup (tree-based structure). Eventually FIB could be copied into TCAM
TCAM=Source IP (4 bytes)+Destination IP (4 bytes)+Protocol (2 bytes)+Type-of-Service (1 byte)

MPLS routers with LFIB (Label Forward Information Base)

Definitions 2/5

Control Plane:

- Tasks: signaling how to forward the user traffic. The routing functions could be distributed across multiple processors
- Examples:

IP routing protocol (OSPF) generates RIB (Routing Information Base), i.e. a routing table stored in RAM. For N routing entries, logical ANDs between destination IP address and netmask => O(N) complexity => inefficient search. FIB reorganizes RIB for faster lookup. *FIB and RIB occupy the same memory in software-defined routers…*

MPLS protocols LDP (Label Distribution Protocol), RSVP (Resource Reservation Protocol) and others generate LIB (Label Information Base)
MPLS Header=Label (20 bits)+Traffic Class (3bits)+Bottom of Stack (1bit)+Time-to-Live (8 bits)

Definitions 3/5

Management Plane:

- Tasks: methods of configuring the control plane
- Examples:

CLI (Command Line Interface)
GUI (Graphical User Interface)
SNMP (Simple Network Management Protocol)
NetFlow
XML
etc.

Definitions 4/5

Virtualization has been proposed long time ago. So nothing is new under the sun…

- **Virtual Circuit (VC):** in connection-oriented packet-switching technologies used in telecom (Frame Relay, ATM) for a better utilization of physical resources. Their addresses (DLCI, VCI) are mapped with IP addresses. MPLS on top of such Layer 2 technologies is mapping VCs with labels.
- **Virtual LAN (VLAN):** appeared as Layer2/Layer 3 technology in Ethernet switches to split broadcast domains into isolated local networks. A software grouping of hosts geographically distributed. It needs a router to interconnect LANs.
- **Virtual Private Network (VPN):** extends a private network across a public one, using tunneling in point-to-point topology. Modern versions based on **L2TP (Layer 2 Tunneling Protocol)** or **VPLS (Virtual Private LAN Service)** allow a complete emulation of LANs across the tunnel (including broadcast domains).

Definitions 5/5

Virtualization continued within the data centers…

- **Virtual Machine (VM):** is an abstraction of an operating system, network service, storage service etc. The behavior is theoretically similar to the physical IT resource. In practice the software-based solution is slower (up to several times) needing hardware acceleration for compensation.

- **Hypervisor:** is the virtualization software allowing multiple operating systems to run simultaneously on a single physical system.

- **I/O Virtualization (IOV):** storage, communications and network devices are virtualized into virtual components. Multiple virtual servers could offer services on separated I/O subsystems (although the physical machine has a single I/O subsystem).

3. SDN

SDN 1/12

Software-Defined Networking (SDN):

"The physical separation of the network control plane from the forwarding plane, and where a control plane controls several devices" [ONF15]

"SDN is new networking paradigm where control of the network is decoupled from the hardware allowing a logically centralized software program to control the behavior of an entire network" [6WIND15]

"SDN is an L2/L3 architecture in which a centralized controller controls the forwarding behavior of a set of distributed switches" [Mar15, Definition 1, p.2, *considered just a subset of Definition 2, see below*]

"SDN is a conceptual framework in which networks are treated as abstractions and are controlled programmatically, with minimal direct "touch" of individual network components" [Mar15, Definition 2, p.2]

SDN 2/12

Standardization Bodies

ONF (Open Networking Foundation): https://www.opennetworking.org/

IETF (Internet Engineering Task Force): https://www.ietf.org/

ETSI (European Telecommunications Standards Institute): http://www.etsi.org/

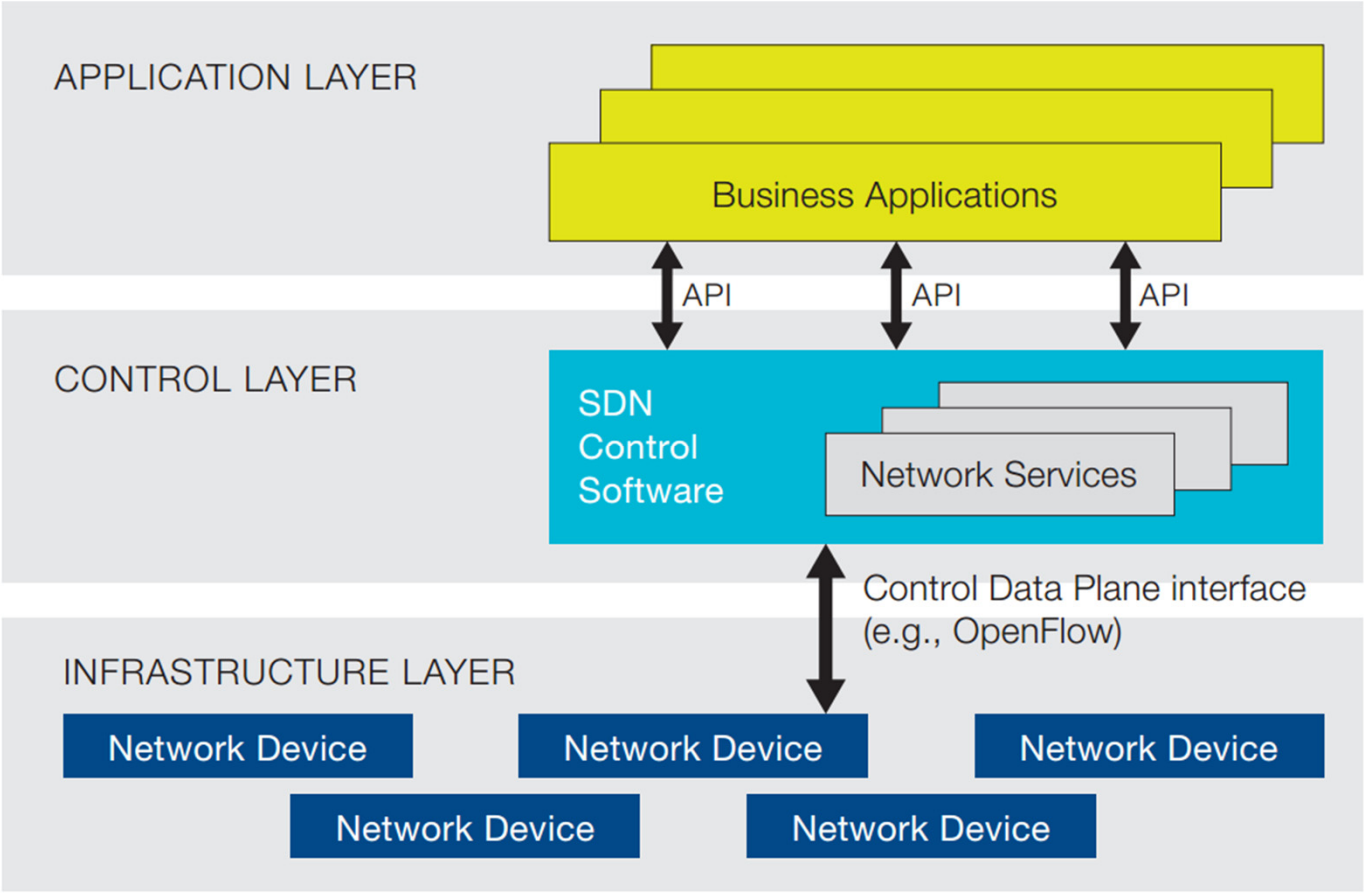
ITU-T (International Telecommunications Union - Telecommunication Standardization Sector): http://www.itu.int/en/Pages/default.aspx

IEEE Software Defined Networks: http://sdn.ieee.org/

5G Public-Private Partnership Association (5GPP): https://5g-ppp.eu/

SDN 3/12

ONF's SDN architecture



Source: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

SDN 4/12

IETF's SDN architecture

RFC 7426, January 2015

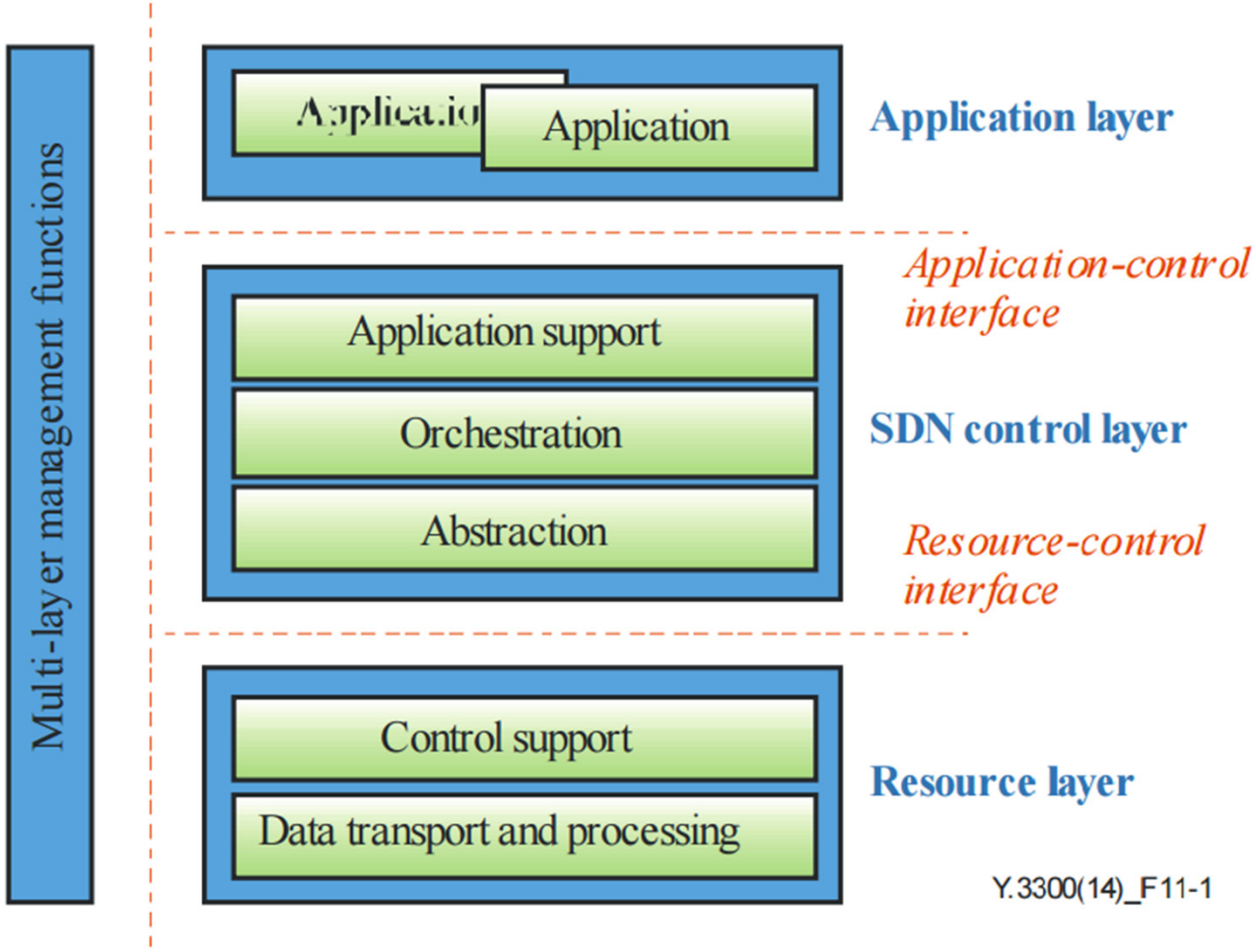
Source: https://tools.ietf.org/html/rfc7426

```
                 o-------------------------------o
                 |                               |
                 | +-------------+   +----------+ |
                 | | Application |   |  Service | |
                 | +-------------+   +----------+ |
                 |       Application Plane       |
                 o---------------Y---------------o
                                 |
*----------------------------Y-------------------------------*
|          Network Services Abstraction Layer (NSAL)         |
*------Y-----------------------------------------------Y-------*
       |                                               |
       |              Service Interface                |
       |                                               |
o------Y-----------------o       o--------------------Y------o
|      |    Control Plane |       | Management Plane    |      |
| +----Y----+   +-----+   |       |  +-----+       +----Y----+ |
| | Service |   | App |   |       |  | App |       | Service | |
| +----Y----+   +--Y--+   |       |  +--Y--+       +----Y----+ |
|      |          |      |       |     |              |      |
| *----Y----------Y----* |       | *---Y--------------Y----* |
| | Control Abstraction | |       | | Management Abstraction | |
| |     Layer (CAL)     | |       | |      Layer (MAL)       | |
| *----------Y----------* |       | *----------Y------------* |
|            |           |       |            |              |
o------------|-----------o       o------------|--------------o
             |                                |
             | CP                             | MP
             | Southbound                     | Southbound
             | Interface                      | Interface
             |                                |
*------------Y--------------------------------Y---------------*
|        Device and resource Abstraction Layer (DAL)          |
*------------Y--------------------------------Y---------------*
|            |                                |              |
|    o-------Y----------o   +-----+   o--------Y----------o   |
|    | Forwarding Plane |   | App |   | Operational Plane |   |
|    o------------------o   +-----+   o-------------------o   |
|                       Network Device                        |
+-------------------------------------------------------------+
```

SDN 5/12

ITU-T's SDN architecture

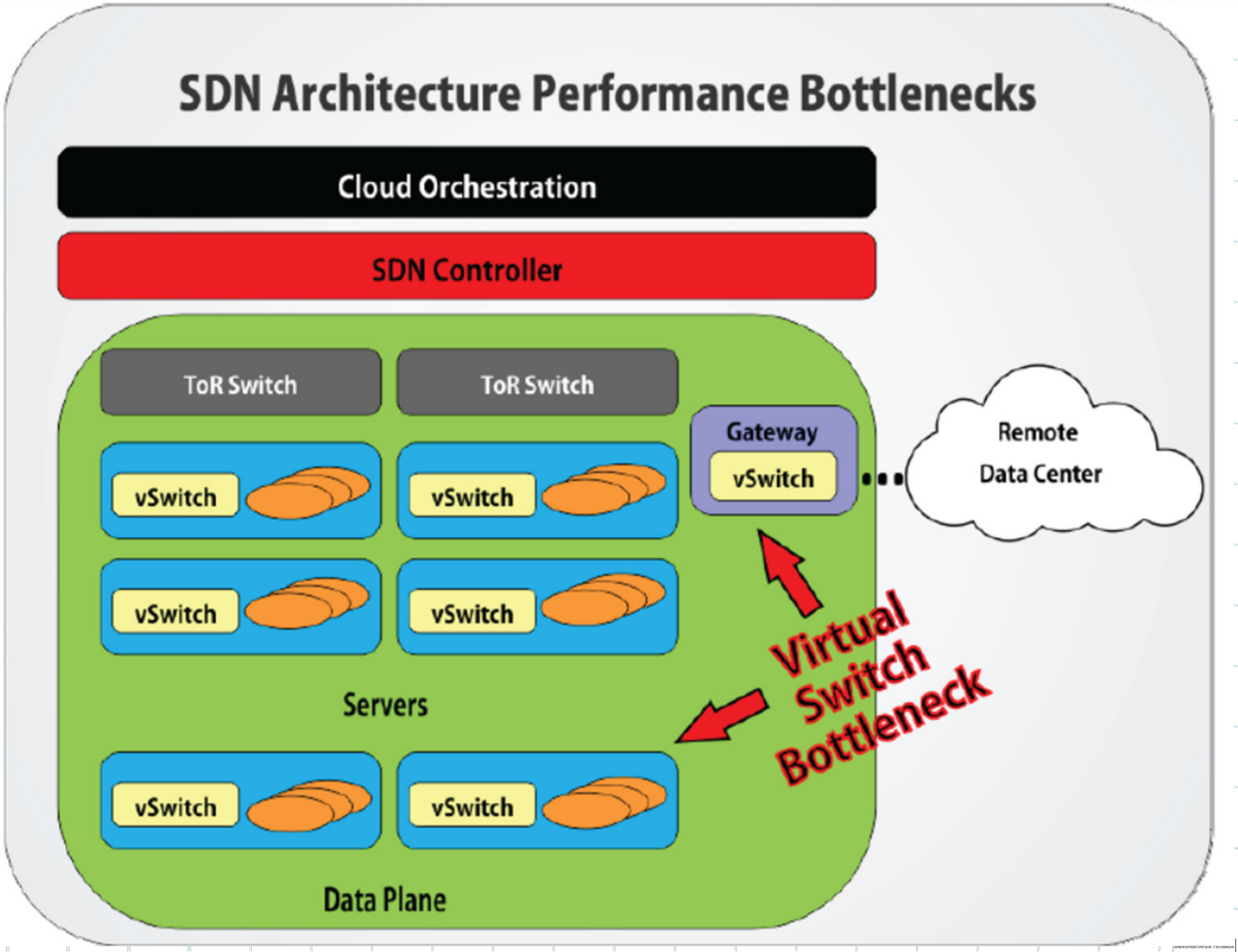
Y.3300,
July 2014



Source: http://www.itu.int/rec/T-REC-Y.3300-201406-I

SDN 6/12

6WIND's SDN architecture



Source: http://www.6wind.com/solutions/software-defined-networking/

SDN 7/12

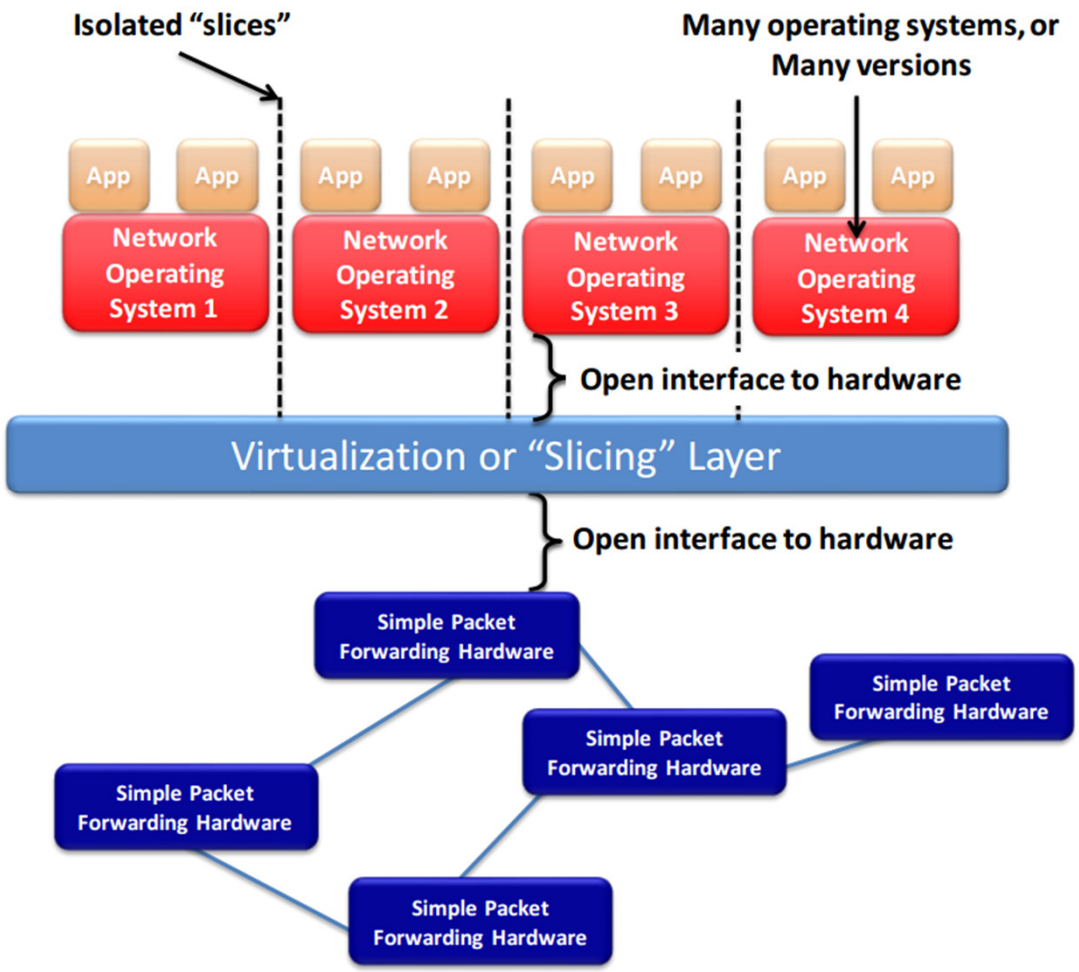
Open vSwitch: is an open-source virtual software switch acting in Data Plane. It forwards the user traffic between different virtual machines VMs on the same physical host, or between them and the physical network.

ToR Switch: is the Top-the-Rack switch in a data center (multiple I/O interfaces GbE/10GbE/40GbE etc.) acting at Layer 2 and Layer 3, performing bridging and Fibre Channel over Ethernet (FCoE) for an entire rack of servers.

SDN Controller: is used in Control Plane and exposes several APIs. Southbound API is used for logical centralized control of network resources (i.e. Open vSwitches) in Data Plane, whilst the Northbound one allows the network/security/business applications to perform a programmatic control of abstracted network resources. The Westbound and Eastbound APIs are used for horizontal interoperability (handover) with other SDN controllers. Note that it could run on a (virtual) server.

SDN 8/12

AT&T's SDN architecture



Source:
https://www.att.com/Common/about_us/pdf/AT&T%20Domain%202.0%20Vision%20White%20Paper.pdf

SDN 9/12

Question 1: Why should we physically separate the control plane from the data plane?

SDN 10/12

Answer to Question 1:

- Each node is trying to perform rerouting but from neighbors perspective. Thus the forwarding decision may not be optimized overall.
- There are a lot of middle-boxes in the network (NAT, firewall, load balancers) without re-routing capabilities.
- Network devices are provided by several vendors (with there own operating system, configuration rules). Thus the admin does not have to configure each switch individually.
- A centralized SDN controller has a complete vision about the network that allows a faster response to link or node failures.
- The routing loops are avoided.
- There is a single interface to Application Layer.

SDN 11/12

Question 2: Which are the disadvantages of a centralized SDN controller? What about countermeasures?

SDN 12/12

Answer to Question 2:

- A single point of failure
- A major target for the attackers
- FIB updates provided by the controller to the switches may generate inconsistency when the geographical distances do matter.

Countermeasures:

- Several SDN controllers may work in a cloud-based network
- Multiple controllers could be logically/physically distributed, hierarchical or hybrid.

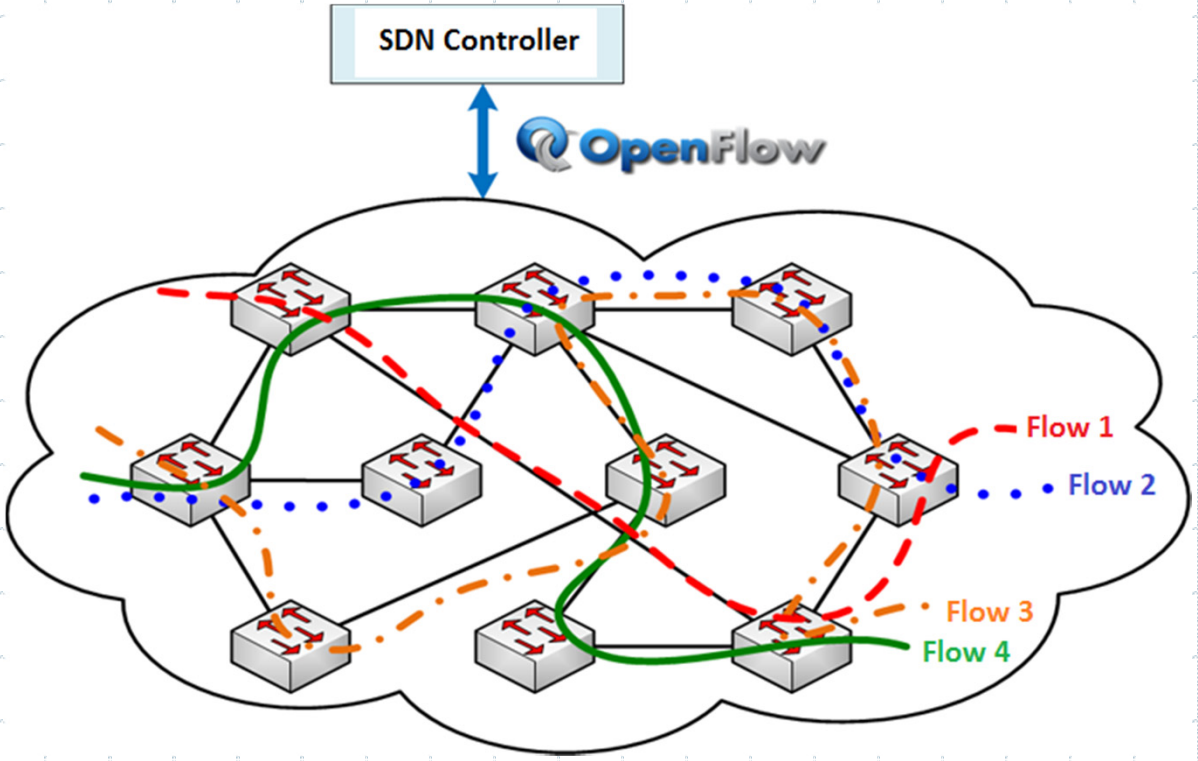
Advantages: traffic load balancing using ECMP (Equal-Cost Multi-Path) or other scheme [Aky14]

Disadvantages: this process has a latency

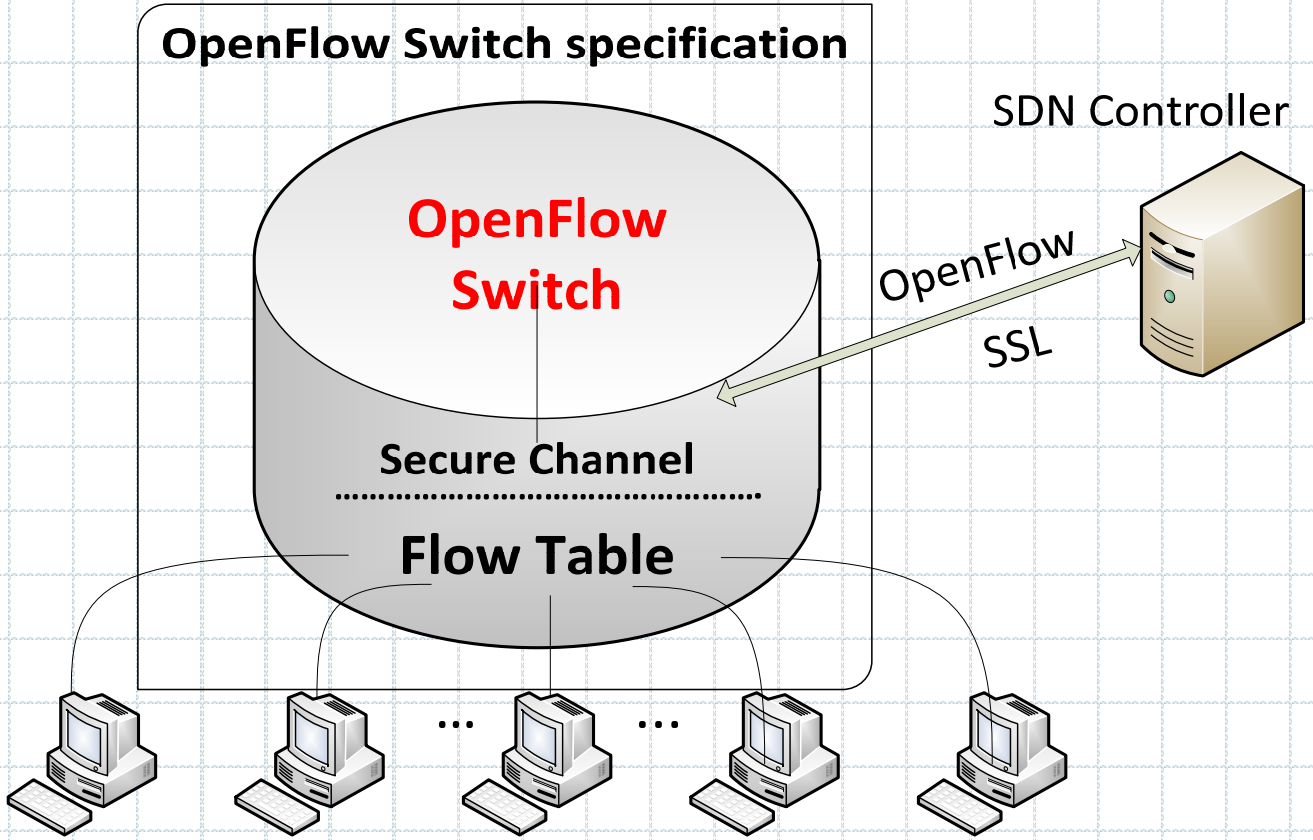
4. SDN and OpenFlow

SDN and OpenFlow 1/17

OpenFlow: is a communications protocol defined at Control Plane Southbound interface (see IETF's architecture) between the SDN controller and virtual/physical switches. It enables the controller to dynamically program the flow tables in the network devices thereby controlling the traffic through the Data Plane at flow level. ONF has the current responsibility for the OpenFlow standard.

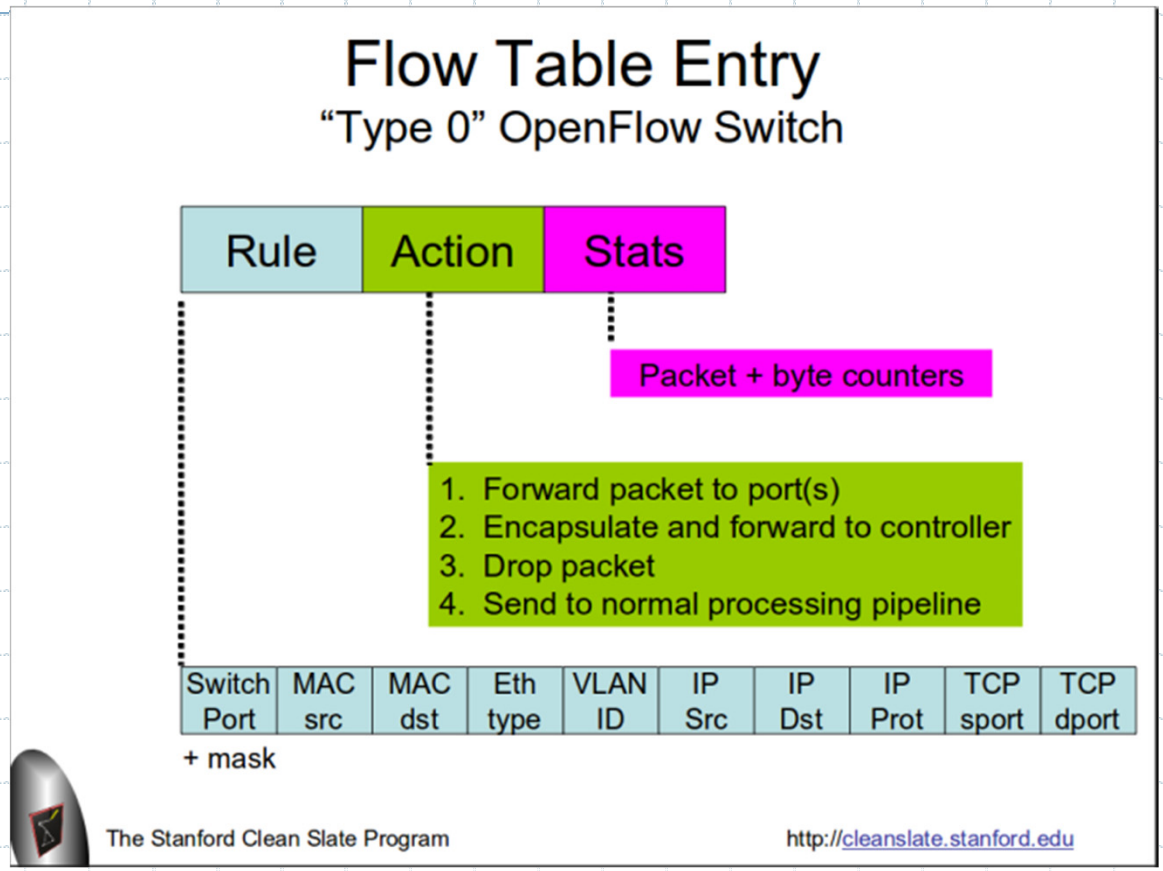


SDN and OpenFlow 2/17



- Although each vendor has a different flow table, the OpenFlow project has identified a common set of functions.
- Secure protocol between the controller and the switch

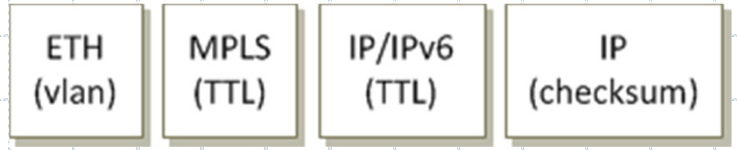
SDN and OpenFlow 3/17



Flow entries are added, updated, and deleted both reactively (in response to packets) and proactively

SDN and OpenFlow 4/17

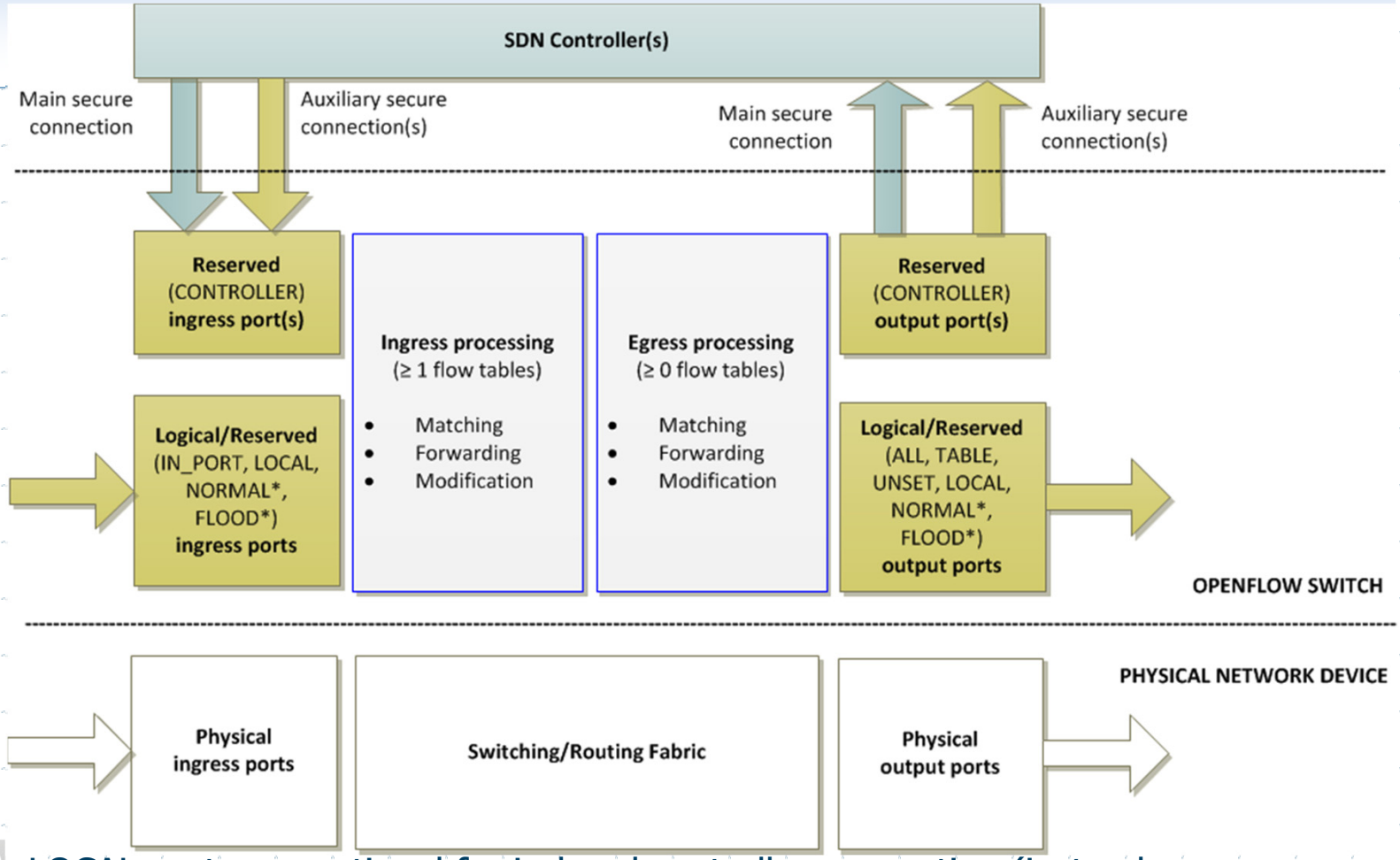
- Three groups of **fields** in a **flow table**:
  - **Match fields:** the first matching entry found allows to execute the instructions associated with that specific flow;



  - **Counters:** maintained for each table, flow, port, queue, group, and bucket
  - **Instructions:** packet forwarding, packet modification, group table processing, and pipeline processing. Note that pipelining allows packets to be sent to subsequent tables for further processing.

- The metadata-form information is communicated between tables.
- Pipelining → packets can be sent to subsequent tables for further processing

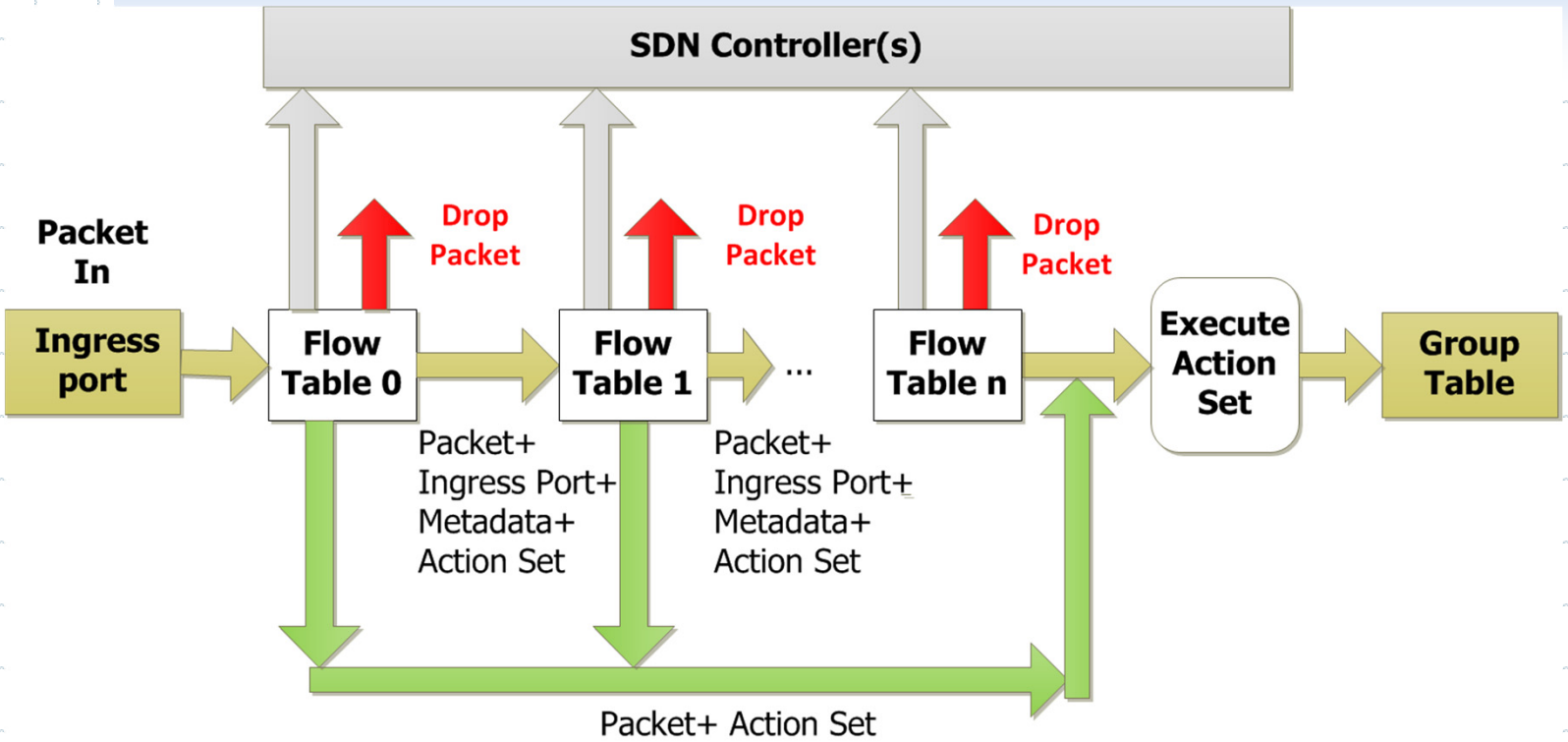
SDN and OpenFlow 5/17



LOCAL ports are optional for in-band controller connection (instead CONTROLLER)

* NORMAL and FLOOD for traditional non-OpenFlow forwarding and flooding are optional (for hybrid devices, i.e. OpenFlow+traditional)

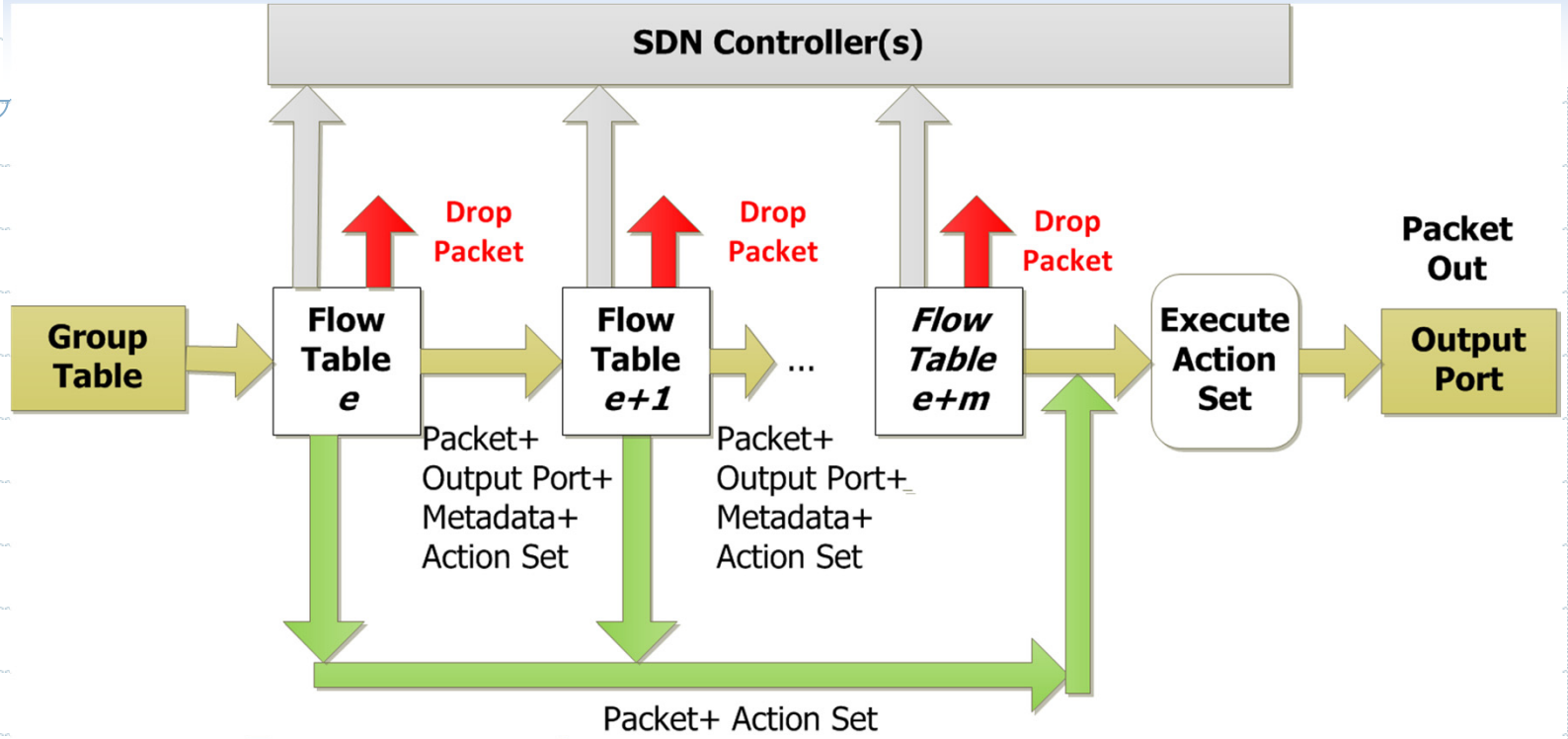
SDN and OpenFlow 6/17



Ingress processing:

- Find the flow entry with the highest-priority matching
- Apply instructions:
  - apply actions instruction: *modify packet & update matching field*
  - clear actions and/or write actions instructions: *update action set*
  - update metadata
- Send the packet + action set to next table or sent the first packet of the unknown flow to SDN controller or discard the packet.

SDN and OpenFlow 7/17



Egress processing:

- Separation in pipeline processing from ingress part is indicated by the first egress flow table *e*
- Note that: *Flow Tables 0…n* are for ingress only and *Flow Tables e…e+m* for egress only (e>n !!!)
- Egress processing is *optional*. If *Flow Table e* is not valid (or not configured)
  => packet goes to the output port

SDN and OpenFlow 8/17

◆ **Table-miss flow entry** within Flow Table: all matching fields omitted + lowest priority (0) =>

- send packet to SDN controller to ask for a new forwarding rule in TCAM/FIB
- drop packet
- send packet to next Flow Table

In case of missing, the packets are discarded *(similar to the case of missing the default route for destination IP networks not defined within the routing table).*

SDN and OpenFlow 9/17

OpenFlow-based open source projects [Azo13a]

Category	Project (organizations)
Switching (soft-switch)	Indigo (Big Switch Networks), LINC (infoBlox), OFSS (Ericsson), of13softswitch (CPqD), **OpenvSwitch (Nicira/VMWare)*,** Pantou/OpenWRT (Stanford University), Pica8/XORPlus (Pica8), Snabb (Snabb), OF13SoftSwitch (CPqD)
OpenFlow controllers	**Beacon (Stanford University)*,** Floodlight (Big Switch Networks), FlowER (Travelping GmbH), Jaxon (University of Tsukuba), NodeFlow (Cisco), NOX (ICSI), POX (ICSI), Ryu NOS (NTT Communications), Trema (NEC), Open-Daylight (Linux Foundation collaborative projects)
Routing	FlowScale (Indiana University), Quagga (Quagga Routing project), Resonance (Georgia Tech), RouteFlow (CPqD)
Orchestration/slicing	FlowVisor (On.Lab), Maestro (Rice University), NDDI OE SS (Internet 2), **OpenStack Quantum (OpenStack Foundation)*,** NetL2API (Locaweb)
Security	FortNOX (SRI International), FRESCO (SRI International)
Simulation, testing and tools	Cbench (Stanford University), MiniNet (Stanford University) NICE (INET), OFDissector (CPqD), OFLOPS (Stanford University), OFTest (Stanford University), Twister (Luxoft), Frenetic (Princeton University), PerfSONAR (ESnet), OFELIA (EICT)
Software libraries	Nettle (Yale University), OFLib-Node (Ericsson), Open Faucet (Midokura), OpenFlowJ (Stanford University)
SDN applications	Avior (Marist College), OSCARS (ESnet), The BIRD (CIXP), FlowScale (Indiana University InCNTRE), Resonance (Georgia Tech)

** The practical demonstration at the end of this lecture is based on these technologies .*

SDN and OpenFlow 10/17

OpenFlow compliant switches [Nun14] [Lar14] [Aky14]

Vendor	Product	Latency (OpenFlow 1.3)
Arista	7124FX	
Broadcom	Strata XGS	N/A
Brocade	MLXe/CER/CES Series	N/A
Ciena	Coredirector firmware v6.1.1	
Cisco	cat6k, catalyst 3750, 6500 series	
Dell	Z9000 and S4810	
HP	5900	1.5 µs
Huawei	SN-640	300-400 ms
IBM	RackSwitch G8264	880 ms
Juniper	MX-240, T-640	
NEC	IP8800, PF5240 PF5820	N/A
Pica8	P-3290, P-3295, P-3780 and P-3920	1 µs
Pronto	3240, 3290, 3780	
Quanta	LB4G	
Toroki	Lightswitch 4810	

SDN and OpenFlow 11/17

Current software switch implementations compliant with the OpenFlow standard [Nun14]

Software Switch	Imple-mentation	Overview	URL
OpenvSwitch*	C/ Python	Open source software switch that aims to implement a switch platform in virtualized server environments. Supports standard management interfaces and enables programmatic extension and control of the forwarding functions. Can be ported into ASIC switches.	http://openvswitch.org/
Pantou/ OpenWRT	C	Turns a commercial wireless router or Access Point into an OpenFlow-enabled switch.	https://www.sdxcentral.com/ projects/pantou-openwrt/
ofsoftswitch13	C/C++	OpenFlow 1.3 compatible user-space software switch implementation.	https://www.openhub.net/p/ ofsoftswitch13
Indigo	C	Open source OpenFlow implementation that runs on physical switches and uses the hardware features of Ethernet switch ASICs to run OpenFlow.	http://www.projectfloodlight. org/indigo/

** The practical demonstration at the end of this lecture is based on these technologies .*

SDN and OpenFlow 12/17

Current controller implementations compliant with the OpenFlow standard [Nun14] 1/3

Controller	Implementation	Open Source	Developer	Overview
POX	Python	Yes	Nicira	General, open-source SDN controller written in Python.
NOX	Python/ C++	Yes	Nicira	The first OpenFlow controller written in Python and C++.
MUL	C	Yes	Kulcloud	OpenFlow controller that has a C-based multi-threaded infrastructure at its core. It supports a multi-level north-bound interface (see Section III-E) for application development.
Maestro	Java	Yes	Rice University	A network operating system based on Java; it provides interfaces for implementing modular network control applications and for them to access and modify network state.
Trema	Ruby/ C	Yes	NEC	A framework for developing OpenFlow controllers written in Ruby and C.

SDN and OpenFlow 13/17

Current controller implementations compliant with the OpenFlow standard [Nun14] 2/3

Controller	Implementation	Open Source	Developer	Overview
Beacon*	Java	Yes	Stanford	A cross-platform, modular, Java-based OpenFlow controller that supports event-based and threaded operations.
Jaxon	Java	Yes	Independent Developers	A Java-based OpenFlow controller based on NOX.
Helios	C	No	NEC	An extensible C-based OpenFlow controller that provides a programmatic shell for performing integrated experiments.
Floodlight	Java	Yes	BigSwitch	A Java-based OpenFlow controller (supports v1.3), based on the Beacon implementation, that works with physical/virtual-OpenFlow switches.
SNAC	C++	No	Nicira	An OpenFlow controller based on NOX-0.4, which uses a web-based, user-friendly policy manager to manage the network, configure devices, and monitor events.

** The practical demonstration at the end of this lecture is based on these technologies .*

SDN and OpenFlow 14/17

Current controller implementations compliant with the OpenFlow standard [Nun14] 3/3

Controller	Implementation	Open Source	Developer	Overview
Ryu	Python	Yes	NTT, OSRG group	An SDN operating system that aims to provide logically centralized control and APIs to create new network management and control applications. Ryu fully supports OpenFlow v1.0, v1.2, v1.3, and the Nicira Extensions.
NodeFlow	JavaScript	Yes	Independent Developers	An OpenFlow controller written in JavaScript for Node.JS [65].
ovs-controller	C	Yes	Independent Developers	A simple OpenFlow controller reference implementation with Open vSwitch for managing any number of remote switches through the OpenFlow protocol; as a result the switches function as L2 MAC-learning switches or hubs.
Flowvisor	C	Yes	Stanford/ Nicira	Special purpose controller implementation.
RouteFlow	C++	Yes	CPqD	Special purpose controller implementation.

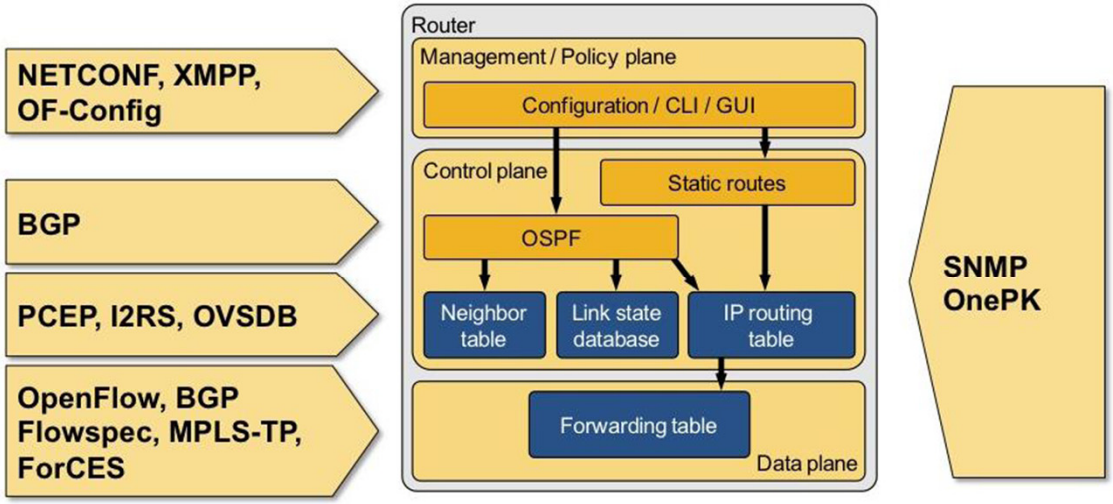
SDN and OpenFlow 15/17

Question 3: Is it possible to implement SDN without OpenFlow?

Answer to Question 3:

- Yes, the competitors are BGP Border Gateway Protocol), XMPP (Extensible Messaging and Presence Protocol), OVSDB, NETCONF (for management, see RFC 6241) etc.

The Many Protocols of SDN



- Vendor APIs: Cisco, Juniper, Arista …
- Scripting: Cisco, Juniper, Arista, Dell, F5 ...

[Pep14b] Ivan Pepelnjak, SDN and OpenFlow -The Hype and the Harsh Reality, ipSpace.net AG, 2014, pp.7-8

SDN and OpenFlow 17/17

Answer to Question 3 (to be continued)

- Major vendors are currently promoting **hybrid SDN**. For instance **Cisco** is offering **APIC-EM** *(Application Policy Infrastructure Controller- Enterprise Module)* for delivering SDN to the Enterprise to the WAN, Campus and Access networks. The distributed solution is highly programmable through open APIs (representational state transfer [REST]).

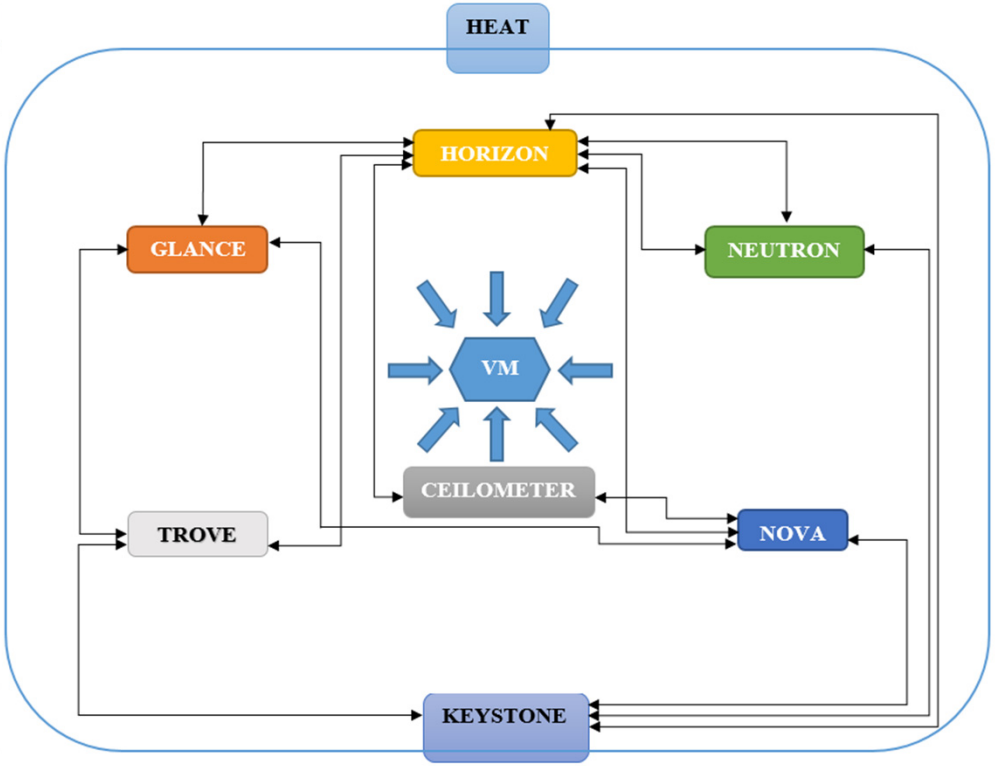
- The trend is to replace the **imperative languages** (telling the "machine" how to do something, and as a result what you want to happen will happen, e.g. C, Java etc.) by **declarative languages** (describing what you want to do, and not how you want to do it, e.g. Haskell, Erlang etc.).

- Cisco claims that APIC-EM can enable independent software developers to create innovative network services and applications on top of a vendor-independent network device. See **https://developer.cisco.com/site/networking/DevNetZone/**

- In November 2015 Cisco and Ericsson announced their agreement to build networks of future based on SDN/NFV.

5. SDN and OpenStack

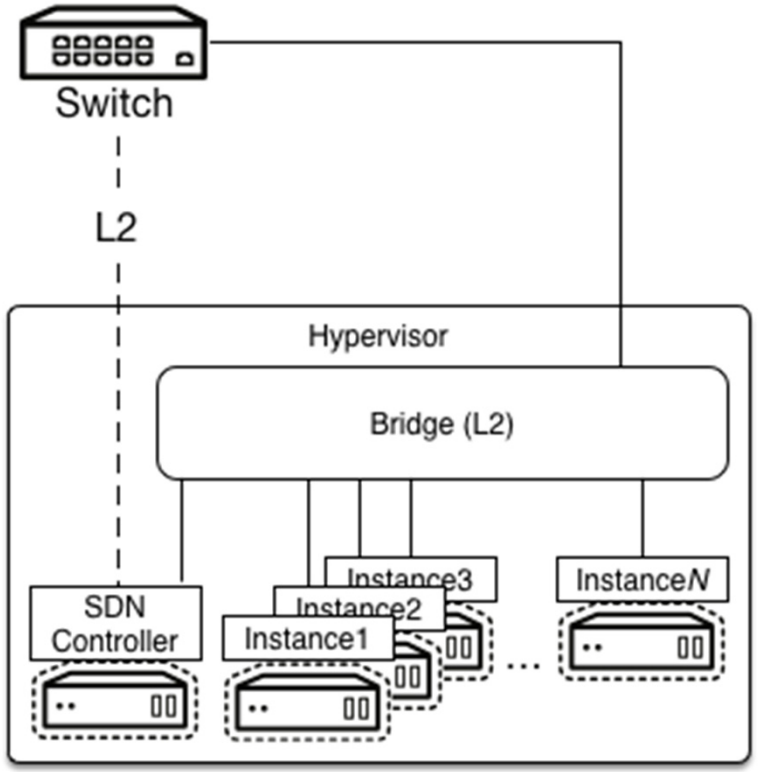
SDN and OpenStack 1/2



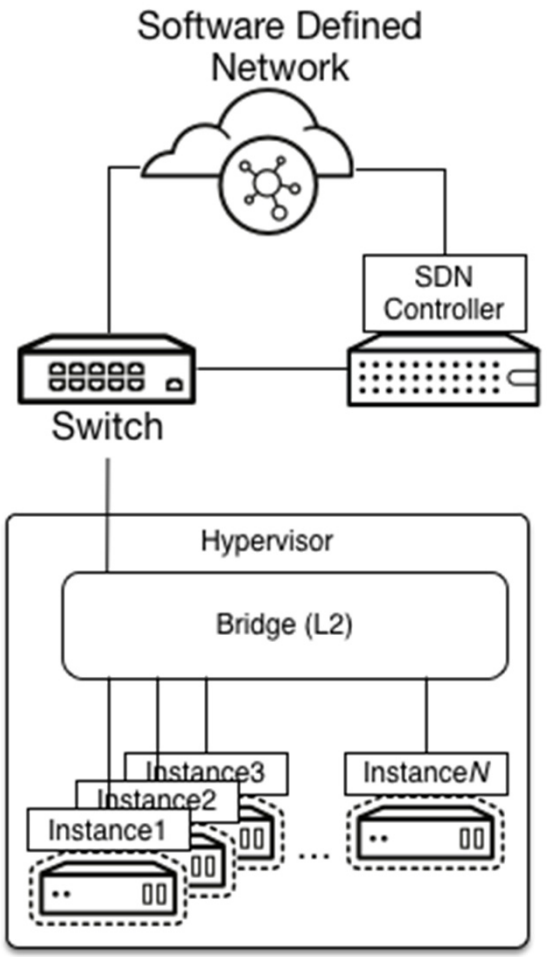
OpenStack components:
GLANCE: stores and retrieves virtual machine disk images
HORIZON: provides a web-based self-service portal
KEYSTONE: responsible for the authentication and authorization service
NOVA: manages the lifecycle of compute instances
NEUTRON: enables network connectivity as a service for other OpenStack services (such as OpenStack Compute)
HEAT: the orchestrator of the cloud CEILOMETER: monitors and meters services within the cloud
TROVE: provides a scalable and reliable database.

OpenStack: Open source project developed by OSF (OpenStack Foundation) as a cloud operating system for computing, networking and storage.

SDN and OpenStack 2/2



OpenStack hosted SDN controller



OpenStack participating in an SDN controller network

Source: OpenStack Architecture Design Guide, current (2015-11-20), Copyright © 2014, 2015 OpenStack Foundation

6. SDN and NFV

SDN and NFV 1/3

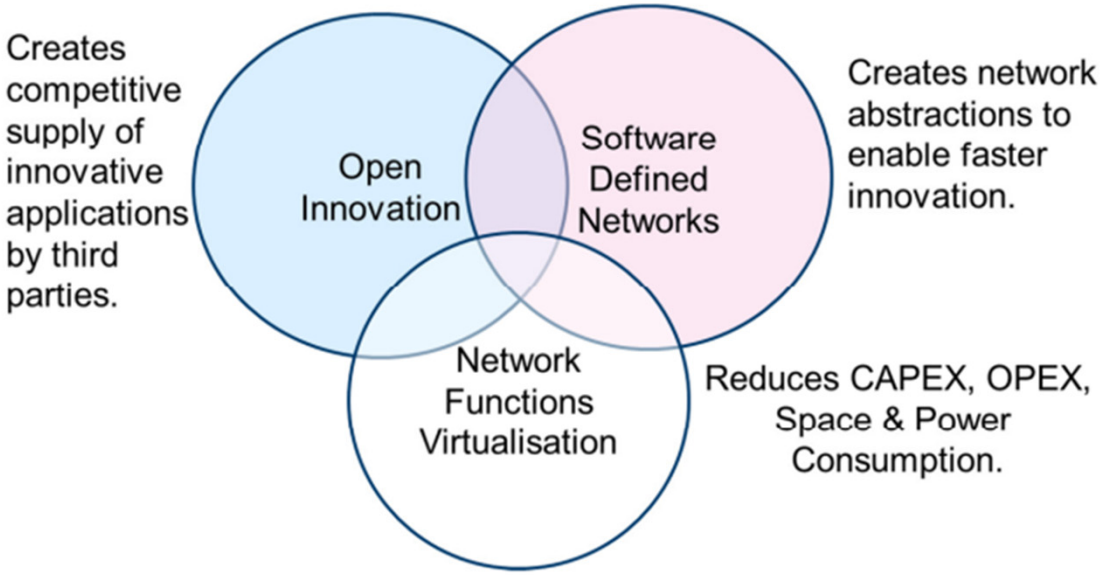
Network Functions Virtualization (NFV): is an initiative of the ETSI Industry Specification Group to virtualize network functions previously performed by proprietary dedicated hardware.

Relationship between SDN and NFV:

Category	SDN	NFV
Reason for Being	Separation of control and data, centralization of control and programmability of network	Relocation of network functions from dedicated appliances to generic servers
Target Location	Campus, data center / cloud	Service provider network
Target Devices	Commodity servers and switches	Commodity servers and switches
Initial Applications	Cloud orchestration and networking	Routers, firewalls, gateways, CDN, WAN accelerators, SLA assurance
New Protocols	OpenFlow	None yet
Standardization Body	Open Networking Forum (ONF)	ETSI NFV Working Group

Source: https://www.sdxcentral.com/articles/contributed/nfv-and-sdn-whats-the-difference/2013/03/

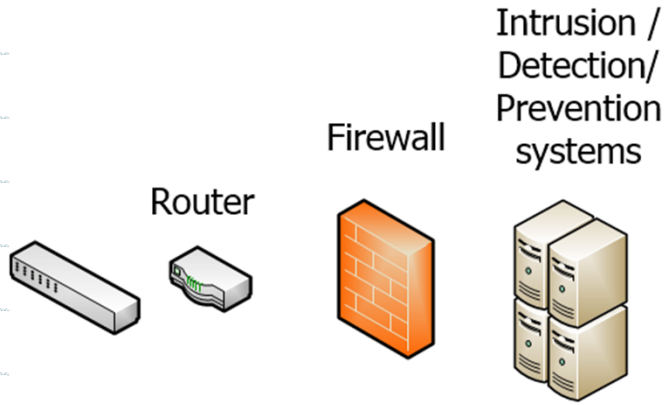
SDN and NFV 2/3



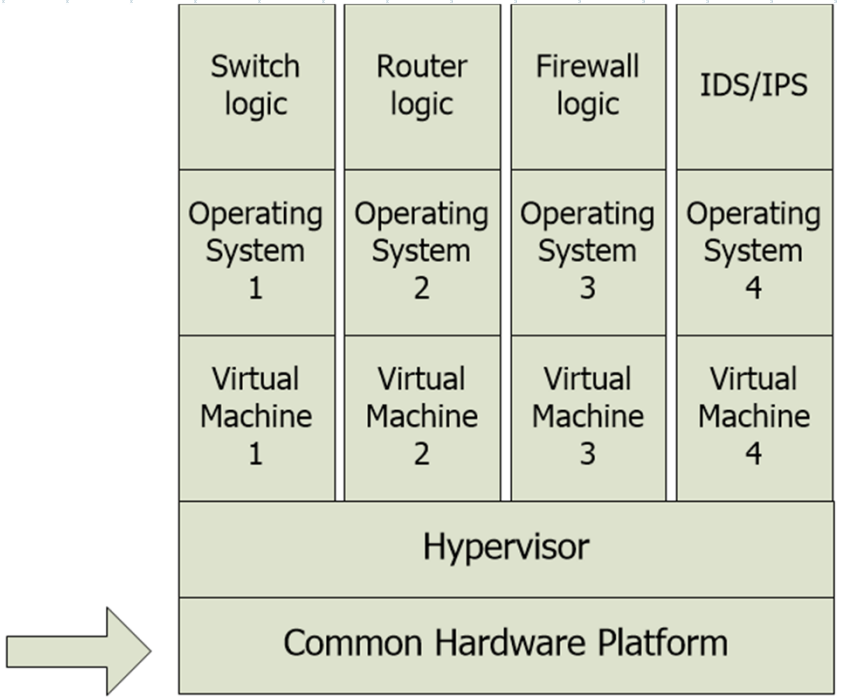
Source: https://portal.etsi.org/NFV/NFV_White_Paper.pdf

- There is a pressure from telecom operators and vendors to have **OpenNFV.**
- **Platinum members:** AT&T, Brocade, China Mobile, Cisco, Dell, Ericsson, HP, Huawei, IBM, Intel, Juniper Networks, NEC, Nokia Networks, NTT DOCOMO, Red Hat, Telecom Italia and Vodafone.
- **Silver members**: 6WIND, Alcatel-Lucent, ARM, CableLabs, Cavium, CenturyLink, Ciena, ClearPath, ConteXtream, Coriant, Cyan, Dorado Software, Ixia, Metaswitch Networks, Mirantis, Orange, Sandvine, Sprint and Wind River.

SDN and NFV 3/3



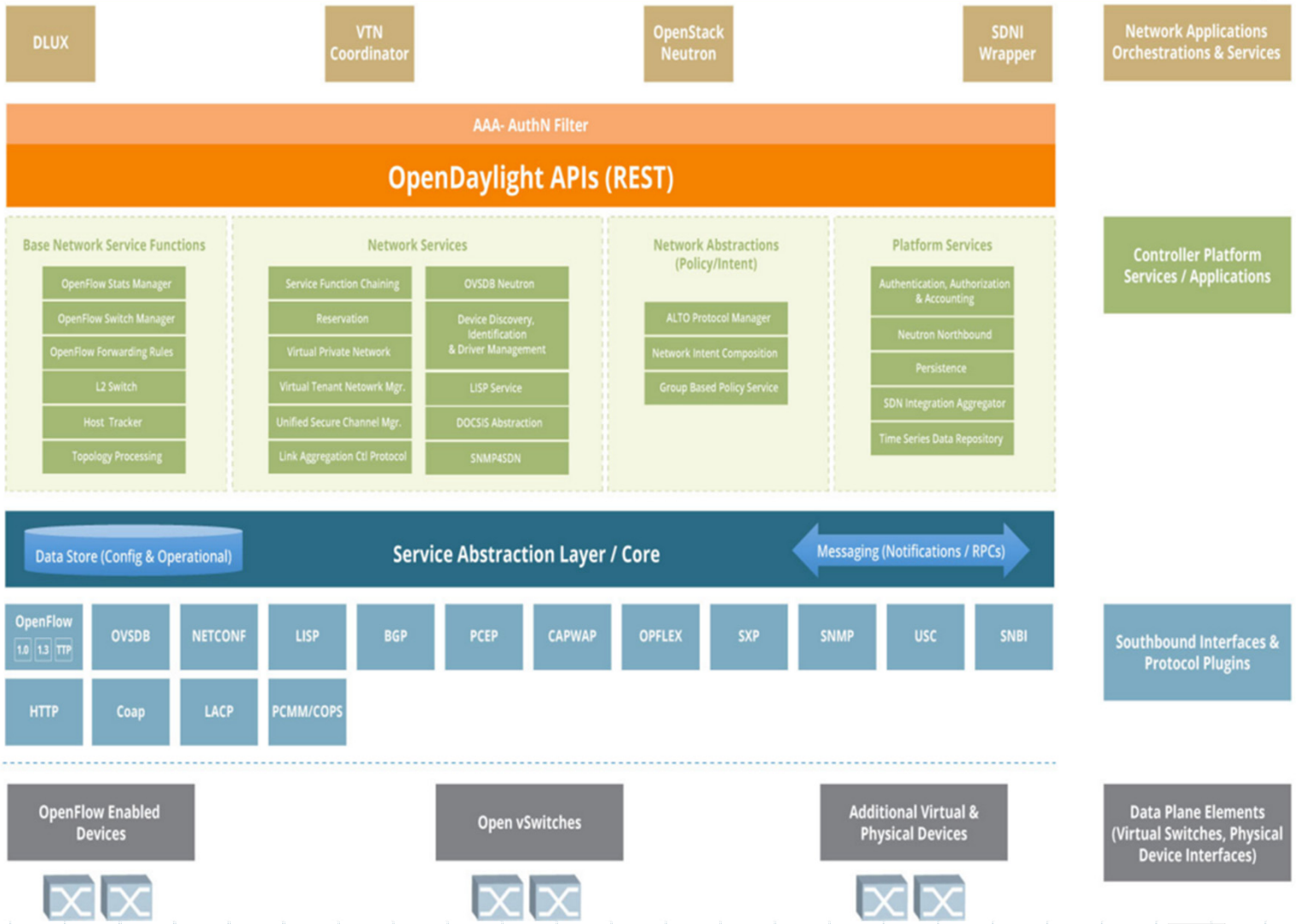
Network functions running on separate network devices



Network functions virtualization based on a virtualized platform

7. SDN and OpenDayLight

SDN and OpenDayLight

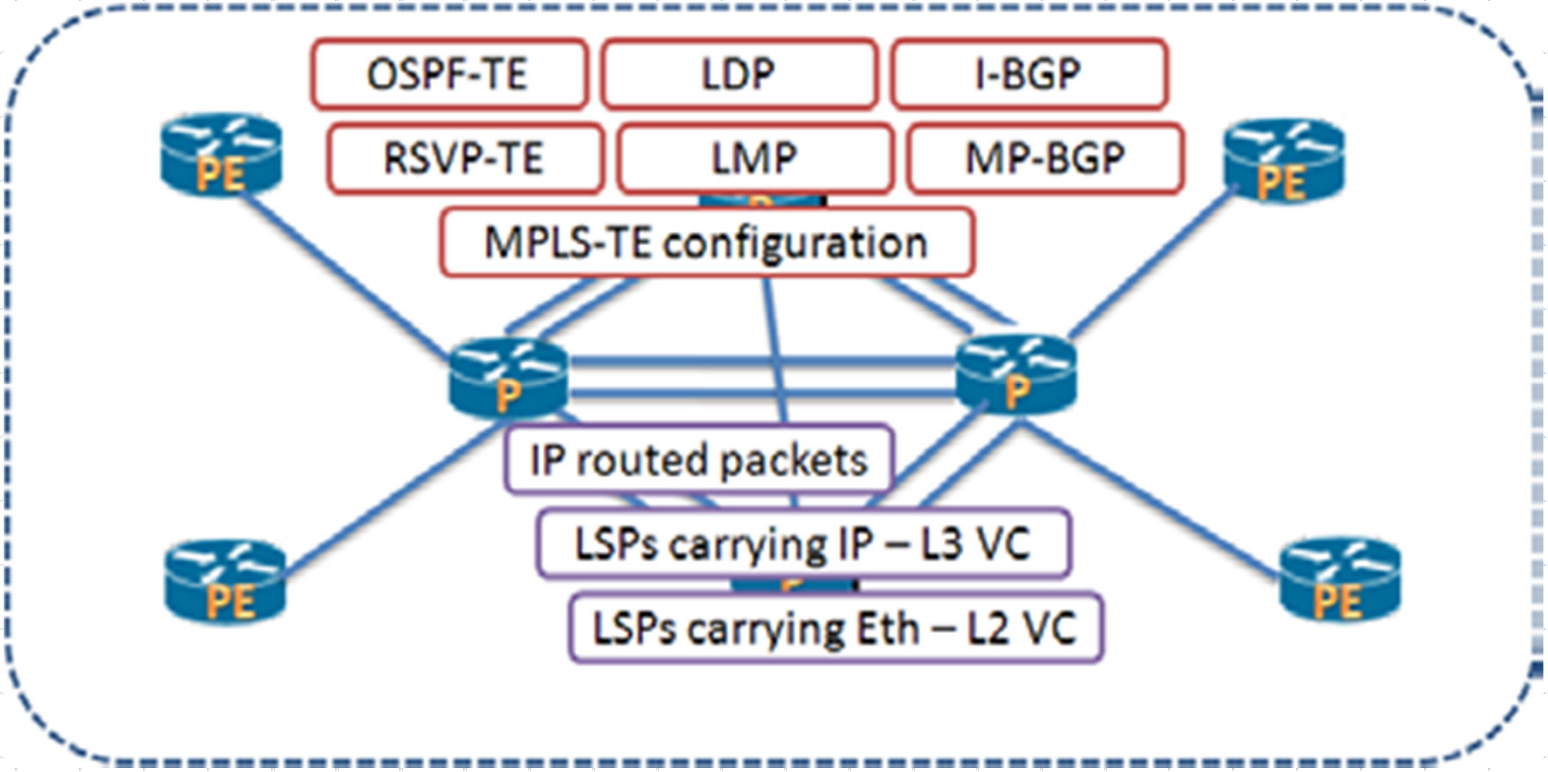


Source: https://www.opendaylight.org/lithium

8. SDN and MPLS

SDN and MPLS 1/2

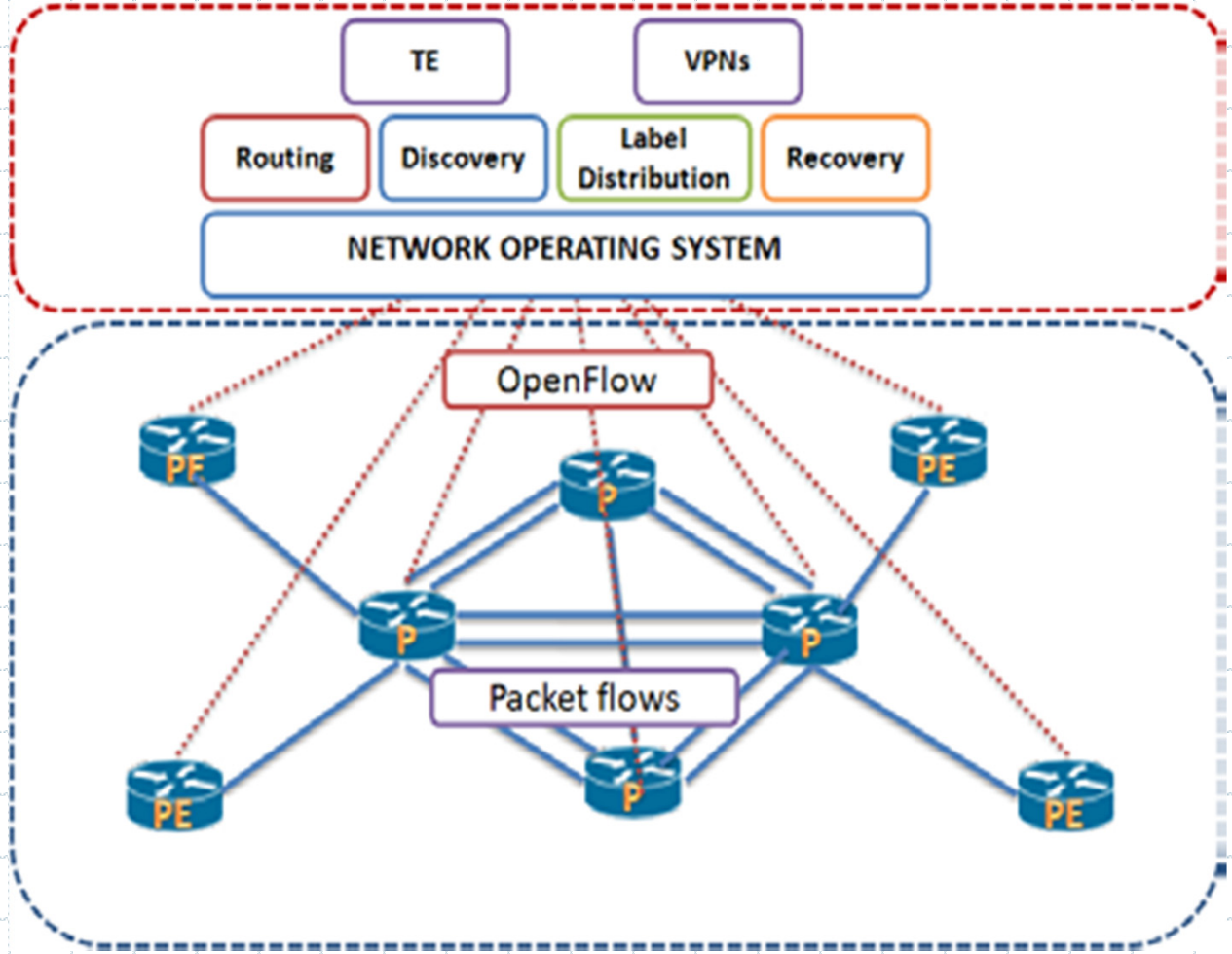
Today's IP/MPLS networks



Source: http://archive.openflow.org/wk/index.php/MPLS_with_OpenFlow/SDN

SDN and MPLS 2/2

MPLS with the Map-abstraction



Source: http://archive.openflow.org/wk/index.php/MPLS_with_OpenFlow/SDN

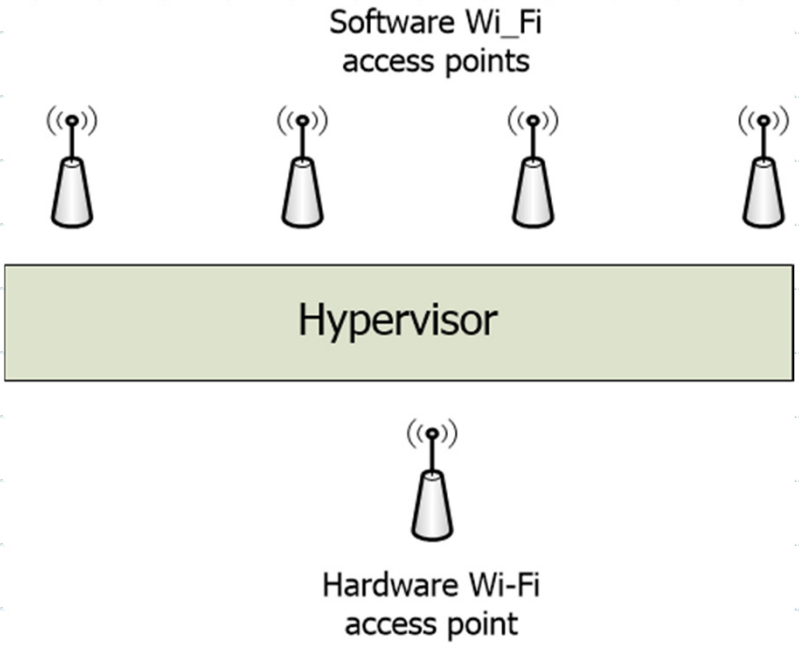
9. SDN and 5G

SDN and 5G 1/6

5G: is defined by Next Generation Mobile Networks (NGMN) Alliance as "end-to-end system environment to enable a fully mobile and connected society". Its architecture will be a native SDN/NFV-based one.

Characteristics:

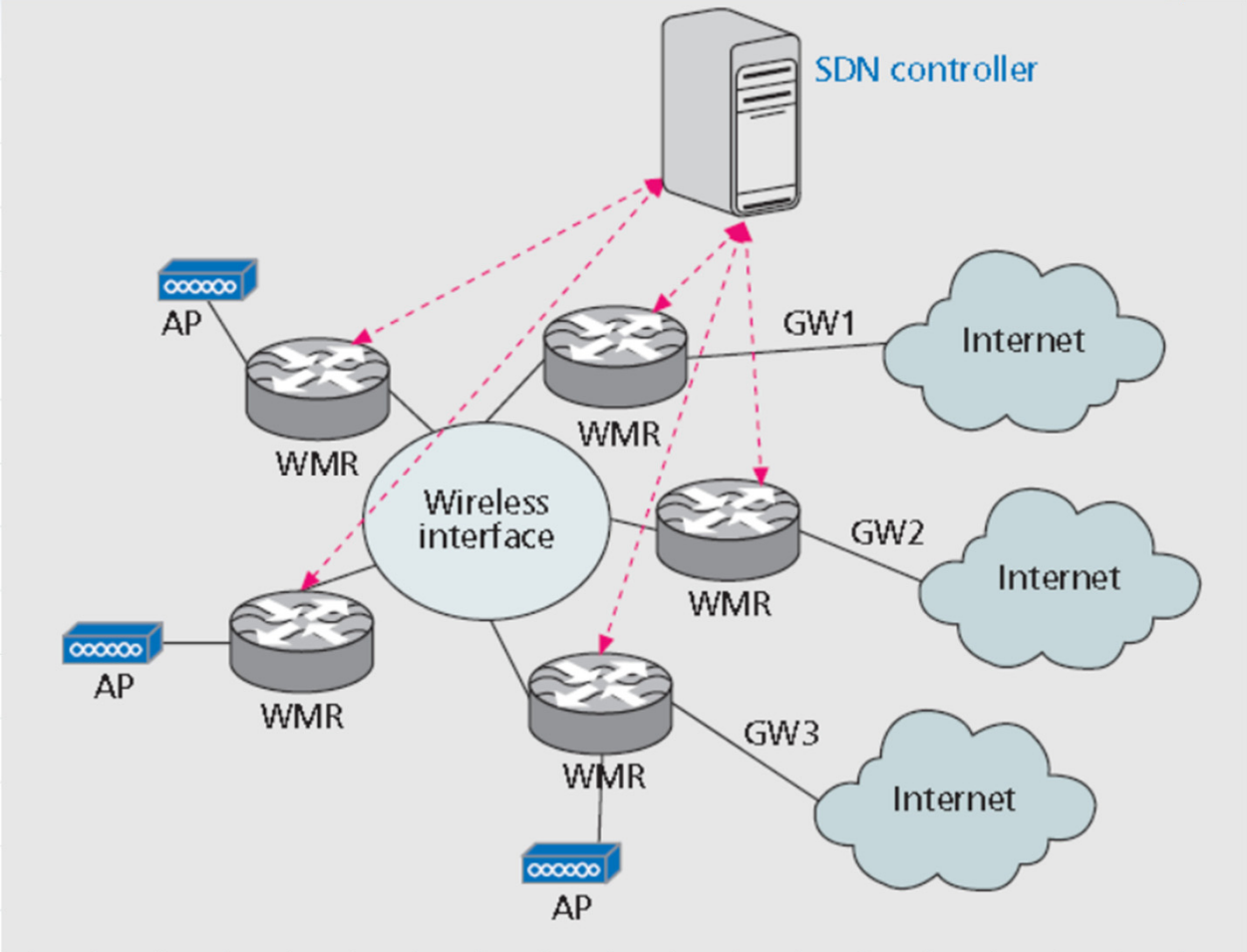
- Transfer rates > 1 Gbps
- Ultra Wide Band (UWB)
- Virtualization of HLRs, VLRs, eNodeB, RANs
- Software-based approach (Software-defined radio, SDN, seamless access)
- Smart antennas, smart cells
- Multi-access, multi-homing, Device to Device (D2D) etc.
- Multi-tenancy (a single instance of a software application serves multiple customers (called atenants).



SDN and 5G 2/6

SDN over Wireless Mesh Networks

- Wireless Mesh Routers (WMR) dynamically selects end-to-end paths through the mesh based on maximizing client-server throughput and minimizing latency.



Source: [Gra15]

- Optimized Link State Protocol (OLSR) is used to route the OpenFlow control/data traffic

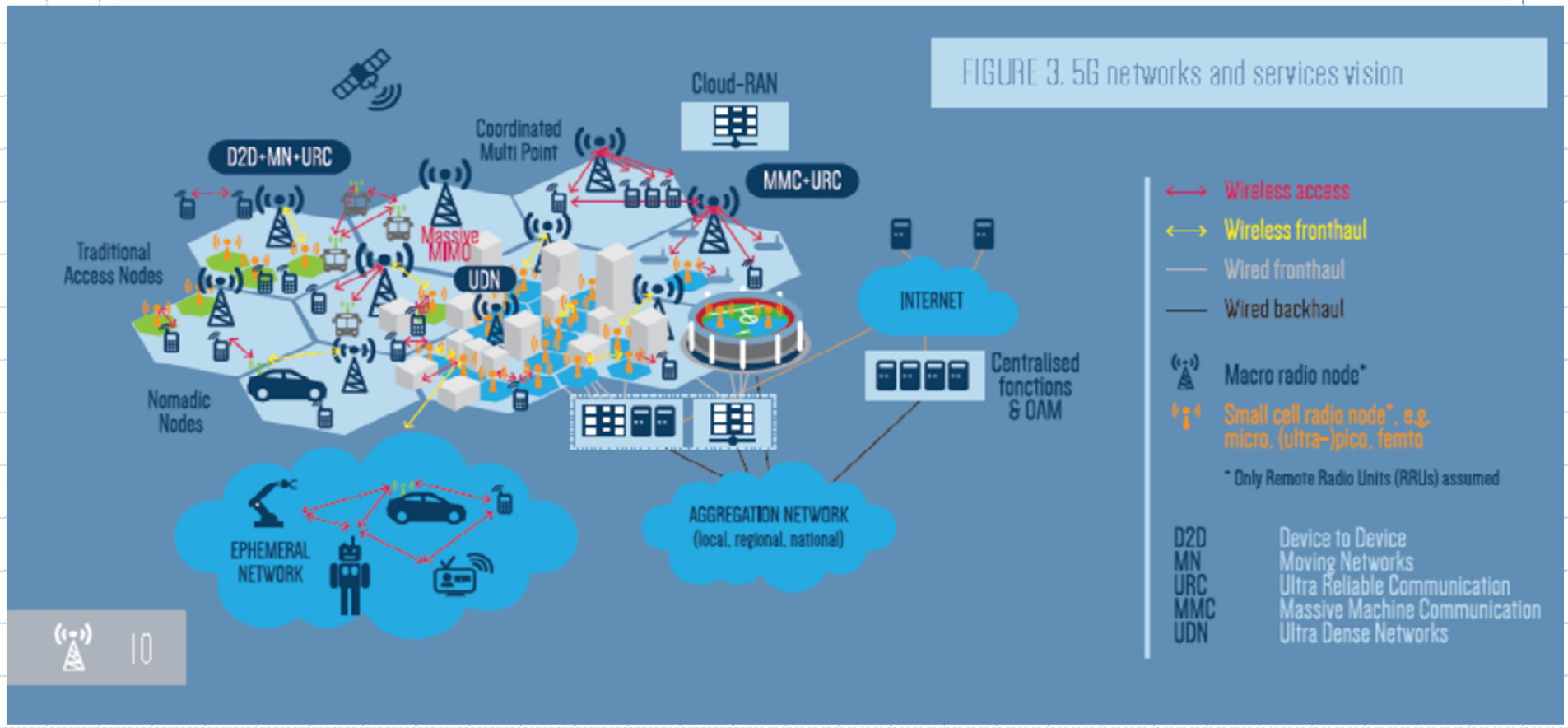
SDN and 5G 3/6

Source: https://5g-ppp.eu/wp-content/uploads/2015/02/5G-Vision-Brochure-v1.pdf

5G Public-Private Partnership Association (5GPP) said that ***"5G** is more than an evolution of mobile broadband. It will be a key enabler of the future digital world, the next generation of ubiquitous ultra-high broadband infrastructure that will support the transformation of processes in all economic sectors and the growing consumer market demand".*

UC Labs
Unified Communications Laboratories

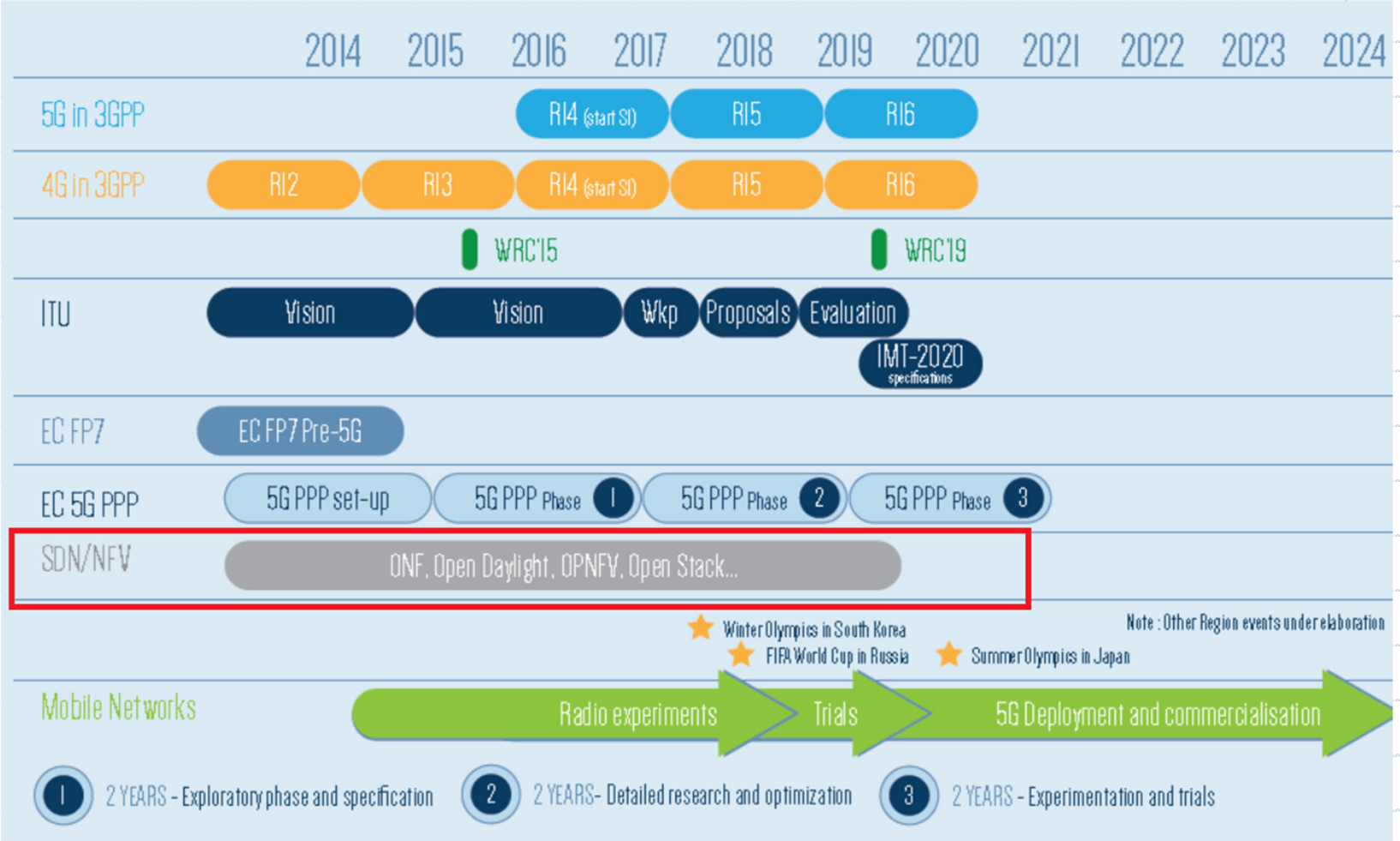
SDN and 5G 4/6



Source: https://5g-ppp.eu/wp-content/uploads/2015/02/5G-Vision-Brochure-v1.pdf

5GPP's vision: "unified control framework through virtualization and programmability of multi-tenant networks and services".

SDN and 5G 5/6



Source: https://5g-ppp.eu/wp-content/uploads/2015/02/5G-Vision-Brochure-v1.pdf

SDN and 5G 6/6

MILESTONES	TABLE I : FROM 5G EXPLORATION TO 5G DEPLOYMENT
2014-2015	Exploratory phase to understand detailed requirements on 5G future systems and identify most promising functional architectures and technology options which will meet the requirements. These activities will build on previous research work in industry and research framework programmes as well as global activities in other regions and standards bodies.
2015-2017	Detailed system research and development for all access means, backbone and core networks (including SDN, NFV, cloud systems, undedicated programmable hardware...) by taking into account economic conditions for future deployment.
2016-2018	Detailed system optimisation by taking into account all identified requirements and constraints. Identification and analysis of frequency bands envisaged for all 5G communications (also taking into account the result of WRC15) and final system definition and optimisation by means of simulations, validation of concepts and early trials. Contributions to initial global standardisation activities e.g. in 3GPP. Preparation of WRC19. Support of regulatory bodies for the allocation of newly identified frequency bands for the deployment of new systems. New frequency bands should be available around 2020.
2017-2018	Investigation, prototypes, technology demos and pilots of network management and operation, cloud-based distributed computing and big data for network operation. Extension of pilots and trials to non ICT stakeholders to evaluate the technical solutions and the impact in the real economy. Detailed standardisation process based on validated system concepts by means of simulations and close to real world trials.
2018-2020	Demonstrations, trials and scalability testing of different complexity depending on standard readiness and component availability.
2020	New frequency bands available for trial network deployment and initial commercial deployment of new systems. Close to commercial systems deployment under real world conditions with selected customers to prepare economic exploitation on global basis.

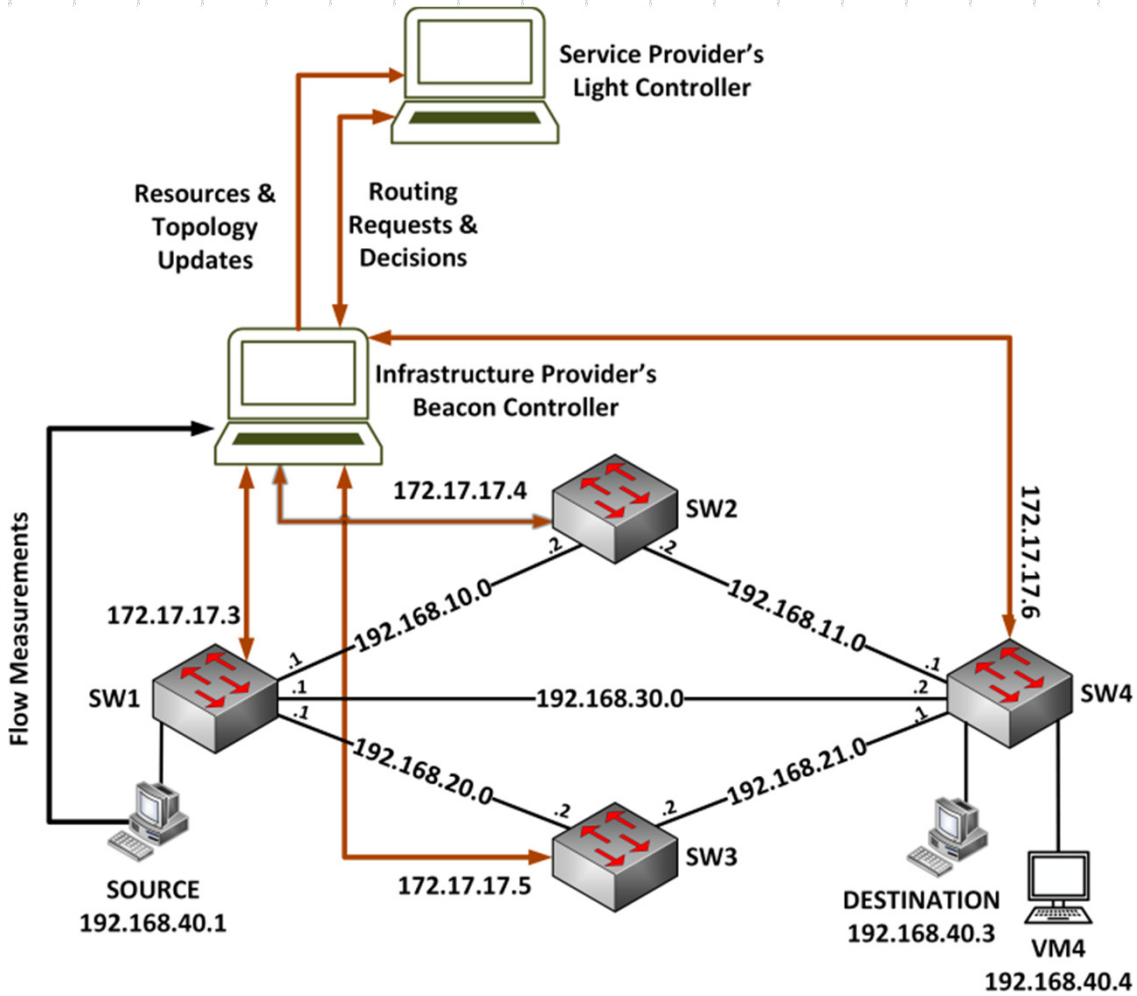
SDN, NFV, cloud systems are milestones in 2015-2017 for 5G deployment after 2020...

Source: https://5g-ppp.eu/wp-content/uploads/2015/02/5G-Vision-Brochure-v1.pdf

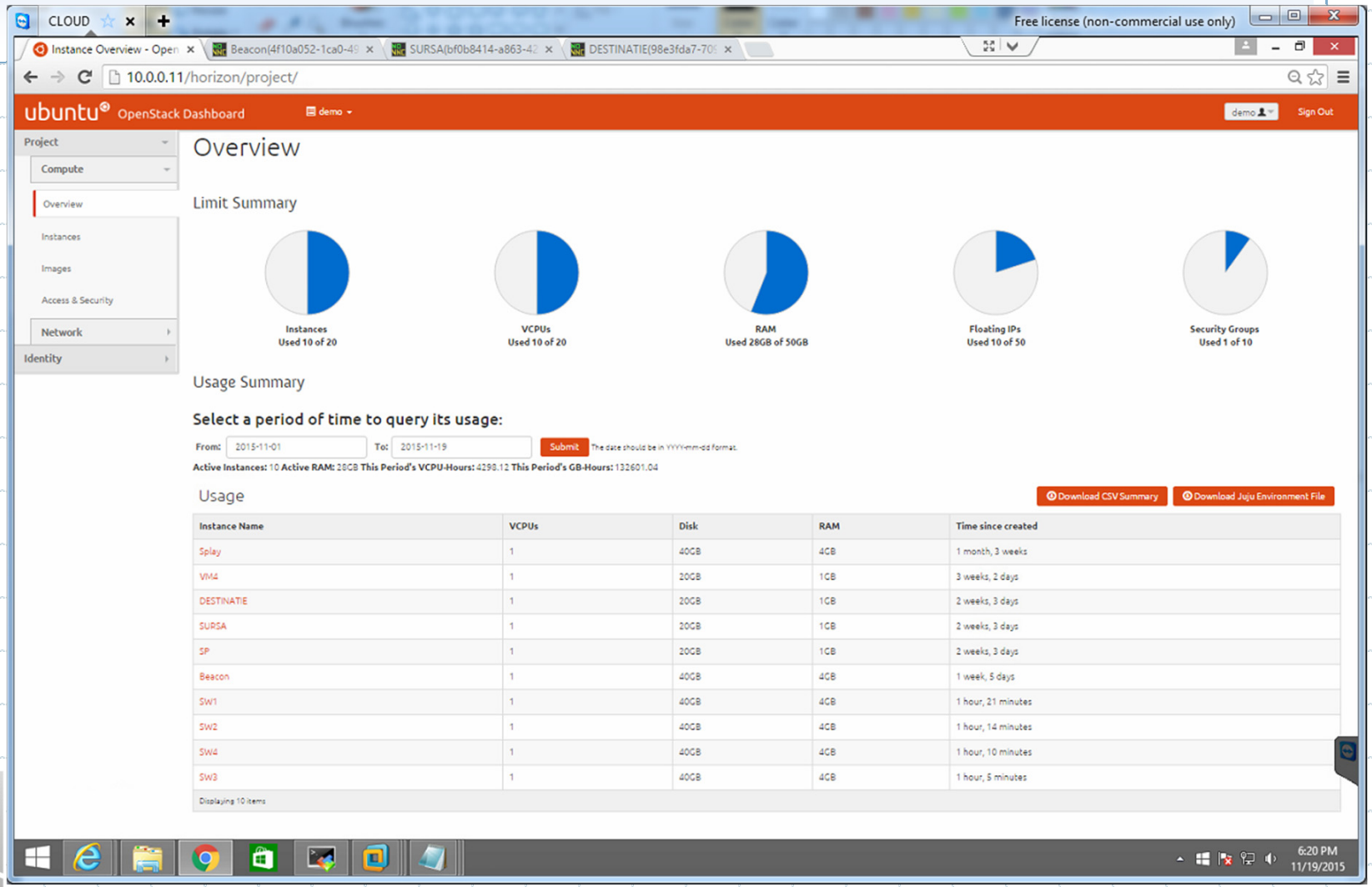
10. Practical Demonstration

Practical Demonstration 1/8

Real implementation of an SDN-based testbed migrated into a private cloud: An OpenStack Approach [Uli14]



Practical Demonstration 2/8



UC Labs
Unified Communications Laboratories

Practical Demonstration 3/8

CLOUD

Free license (non-commercial use only)

Instances - OpenStack Da | Beacon(4f10a052-1ca0-49 | SURSA(bf0b8414-a863-42 | DESTINATIE(98e3fda7-70

10.0.0.11/horizon/project/instances/

ubuntu OpenStack Dashboard | demo | demo | Sign Out

Project
- Compute
  - Overview
  - Instances
  - Images
  - Access & Security
- Network
- Identity

Instances

Instances

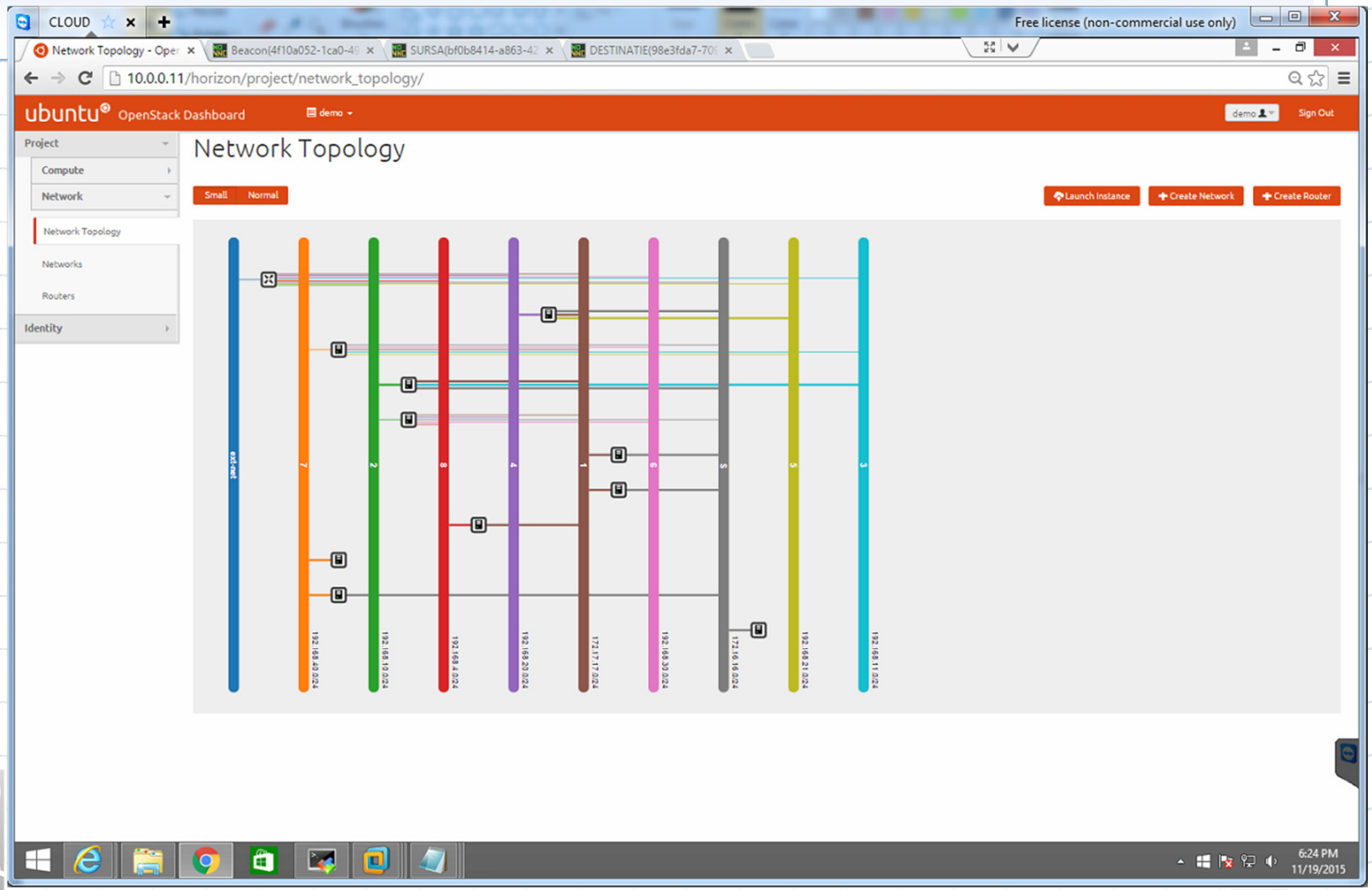
Instance Name | Filter | Filter | Launch Instance | Soft Reboot Instances | Terminate Instances

Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
SW3	SW3B-08.10.15	1 172.17.17.5 S 172.16.16.4 192.168.100.106 5 192.168.21.2 4 192.168.20.2	m1.medium	demo-key	Active	nova	None	Running	3 hours, 47 minutes	Create Snapshot
SW4	SW4B-08.10.15	1 172.17.17.6 3 192.168.11.1 5 192.168.21.1 7 192.168.40.10 6 192.168.30.2 S 172.16.16.5 192.168.100.107	m1.medium	demo-key	Active	nova	None	Running	3 hours, 53 minutes	Create Snapshot
SW2	SW2B-08.10.15	1 172.17.17.4 3 192.168.11.2 2 192.168.10.2 S 172.16.16.3	m1.medium	demo-key	Active	nova	None	Running	4 hours	Create Snapshot

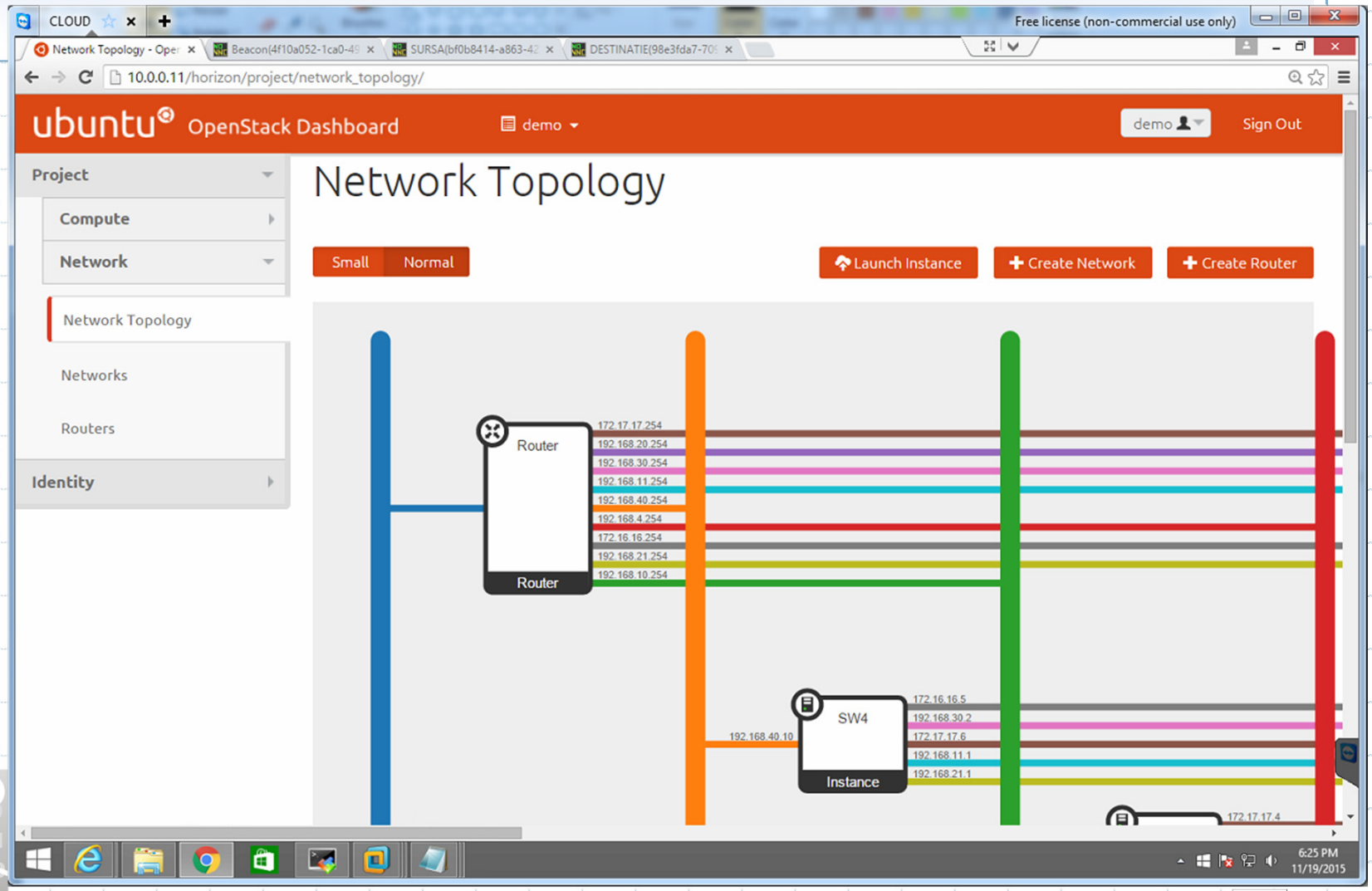
6:20 PM
11/19/2015

UC Labs
Unified Communications Laboratories

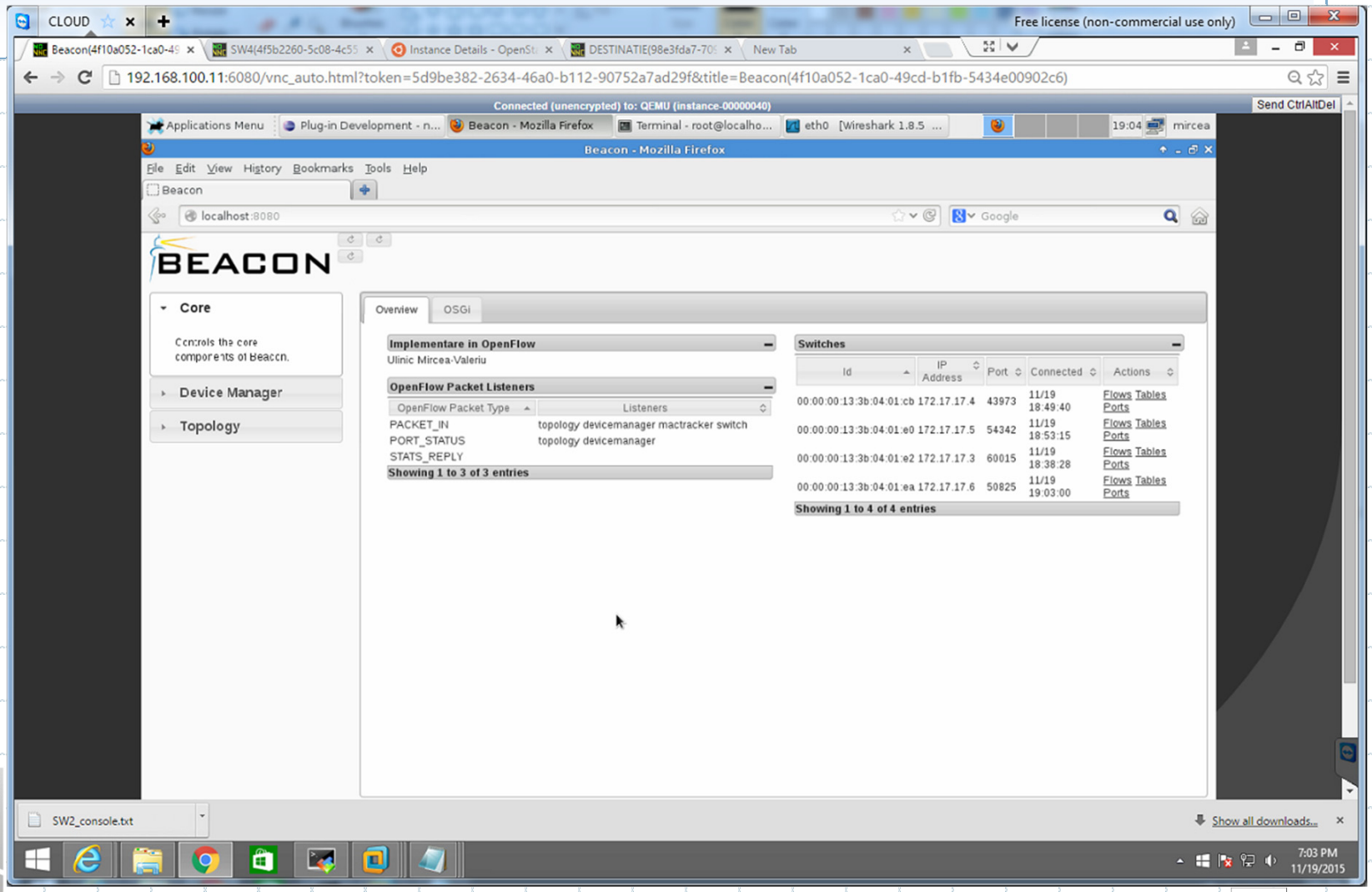
Practical Demonstration 4/8



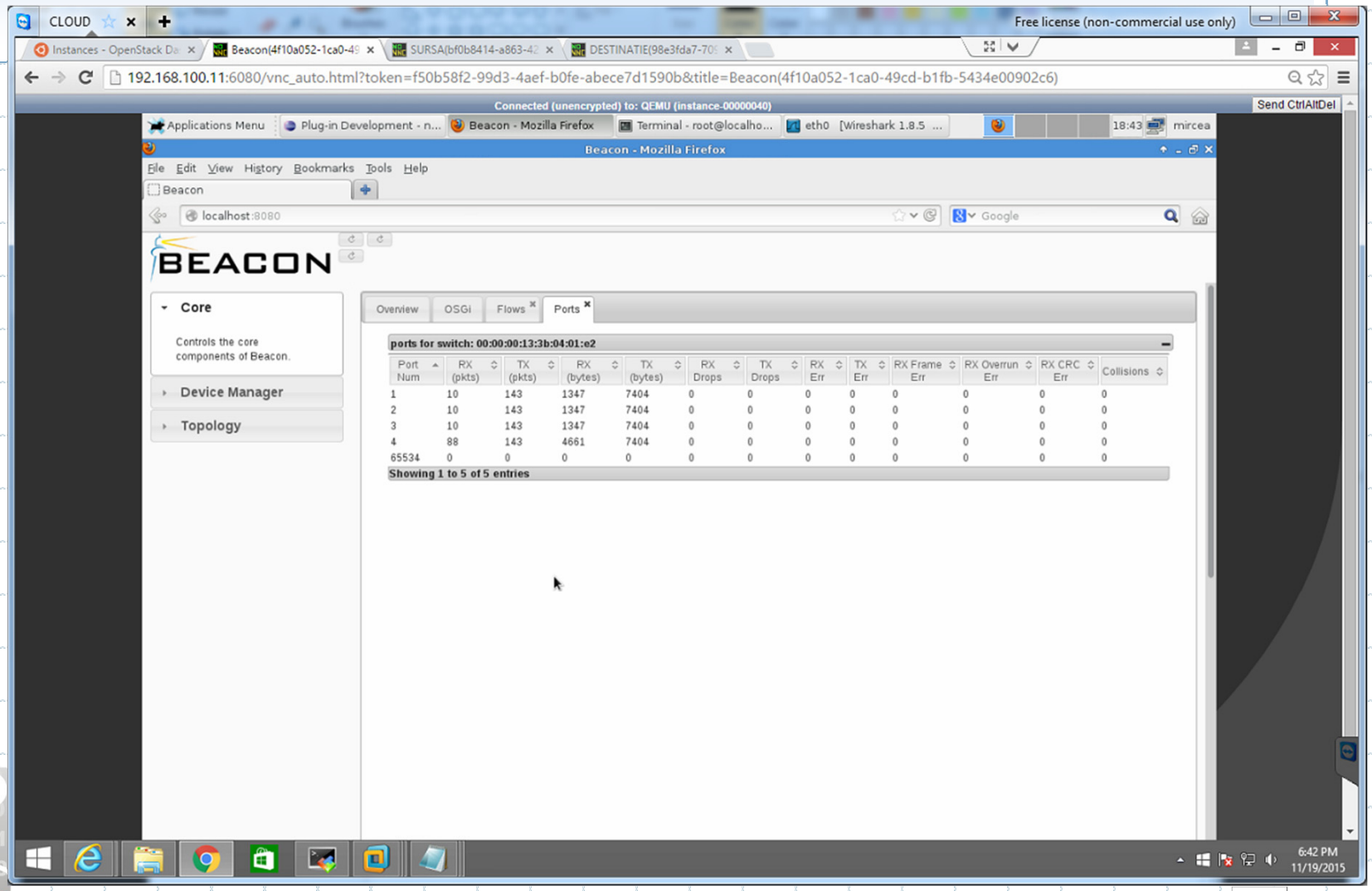
Practical Demonstration 5/8



Practical Demonstration 6/8

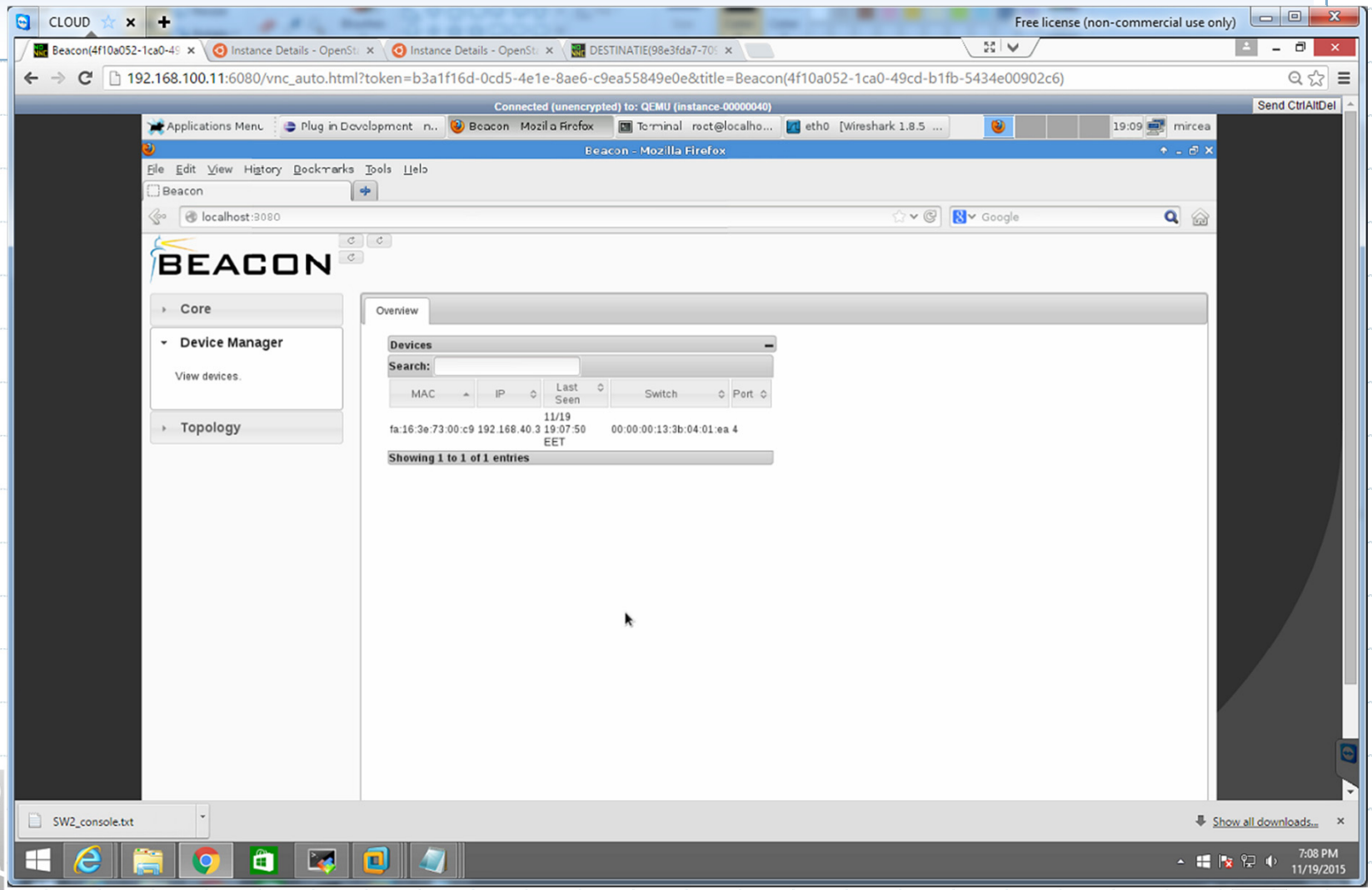


Practical Demonstration 7/8



UC Labs
Unified Communications Laboratories

Practical Demonstration 8/8



UC Labs
Unified Communications Laboratories

11. Conclusions

Conclusions

- SDN, NFV are of major importance from telecom perspective.
- OpenFlow is the most popular protocol for Southbound interface between control plane and forwarding plane.
- Almost all standardization bodies are involved in SDN
- A large consortium of telecom operators and manufacturers are promoting SDN solutions driven by software and being hardware-independent.
- OpenNFV is urgently requested by the industry

12. References

References: Books

- **[Gor14]** Paul Goransson, Chuck Black, *Software Defined Networks: A Comprehensive Approach, 1st Edition,* Elsevier Inc., 2014
- **[Liy15]** Madhusanka Liyanage, Andrei Gurtov, Mika Ylianttila (Editors), *Software Defined Mobile Networks (SDMN): Beyond LTE Network Architecture (Wiley Series on Communications Networking & Distributed Systems), 1st Edition,* John Wiley & Sons, Ltd., 2015
- **[Mar15]** Doug Marschke, Jeff Doyle, Pete Moyer, *Software Defined Networking (SDN): Anatomy of OpenFlow Volume I,* Lulu.com, 2015
- **[Pep14a]** Ivan Pepelnjak, *Overlay Virtual Networks in Software-Defined Data Centers: Architectures, Technical Details and Products,* ipSpace.net AG, 2014
- **[Pep14b]** Ivan Pepelnjak, *SDN and OpenFlow -The Hype and the Harsh Reality*, ipSpace.net AG, 2014
- **[Pep15]** Ivan Pepelnjak, *Data Center Design Case Studies from DMVPN and WAN Edge to Server Connectivity and Virtual Appliances,* ipSpace.net AG, 2015
- **[Puj15]** Guy Pujolle, *Software Networks: Virtualization, SDN, 5G, Security (Networks & Telecommunication: Advanced Networks), 1st Edition,* ISTE Ltd. and John Wiley & Sons, Inc., 2015
- **[Sta16]** William Stallings, *Foundations of Modern Networking: SDN, NFV, QoE, IoT, and Cloud, 1st Edition,* Pearson Education, Inc. , 2016

References: Papers 1/2

- **[Aky14]** I.F. Akyildiz, A.Lee, P.Wang, M.Luo, W.Chou, "A roadmap for traffic engineering in SDN-OpenFlow networks", *Computer Networks*, 71 (2014), pp.1–30
- **[Azo13]** S. Azodolmolky, P. Wieder, and R. Yahyapour, "Cloud computing networking: challenges and opportunities for innovations", *IEEE Communications Magazine,* 51(7), 2013, pp.54-62
- **[Fur13]** A.G. Furculita, M.V. Ulinic, A.B. Rus, and V. Dobrota, "Implementation Issues for Modified Dijkstra's and Floyd-Warshall Algorithms in OpenFlow", *12th IEEE RoEduNet International Conference,* Constanta, Romania, 2013, pp.50-55, DOI:10.1109/RoEduNet.2013.6714208
- **[Gra15]** F.Granelli, et al., "Software defined and virtualized wireless access in future wireless networks: Scenarios and standards", *IEEE Communications Magazine - Communications Standards Supplement*, June 2015, pp.26-34
- **[Kre15]** D. Kreutz, F.M.V. Ramos, P. Esteves Verissimo, C. Esteve Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey", *Proceedings of the IEEE,* Vol. 103, No.1, 2015, pp.14-76

References: Papers 2/2

- **[Lar14]** A. Lara, A. Kolasani, and B. Ramamurthy, "Network innovation using openflow: A survey". *IEEE Communications Surveys & Tutorials,* 16(1), 2014, pp. 493-512.
- **[Lin15]** P. Lin, J. Bi, S. Wolff, Y. Wang, A. Xu, Z. Chen, H. Hu and Y. Lin, "A West-East Bridge Based SDN Inter-Domain Testbed", *IEEE Communications Magazine*, Vol.53, No.2, 2015, pp. 190-197
- **[McK08]** N. McKeown, et al., OpenFlow: enabling innovation in campus networks: ACM SIGCOMM Computer Communication Review, 38 (2), April 2008, pp. 69-74
- **[Nun14]** B. Nunes, M. Mendonca, X. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks", *HAL Id: hal-00825087*, 2014 https://hal.inria.fr/hal-00825087v2
- **[Uli14]** M.V. Ulinic, A.B. Rus, and V. Dobrota, "OpenFlow-Based Implementation of a Gearbox-Like Routing Algorithm Selection in Runtime", *ACTA TECHNICA NAPOCENSIS, Electronics and Telecommunications*, ISSN 1221-6542, Vol.55, No.2, 2014, pp.23-32

Acknowledgments

I acknowledge the support from

- Current UC Labs members: Iustin Ivanciu, Eduard Luchian, Bogdan Rus, Gabriel Lazar, Daniel Zinca
- Former UC Labs members: Mircea Ulinic, Adrian Furculita
- „CHIST-ERA DIONASYS "Declarative and Interoperable Overlay Networks, Applications to Systems of Systems" Project, 2015-2017