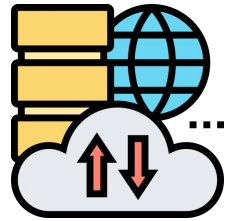
COEN 241
Introduction to Cloud Computing

Lecture 13 - Software Defined Networking II



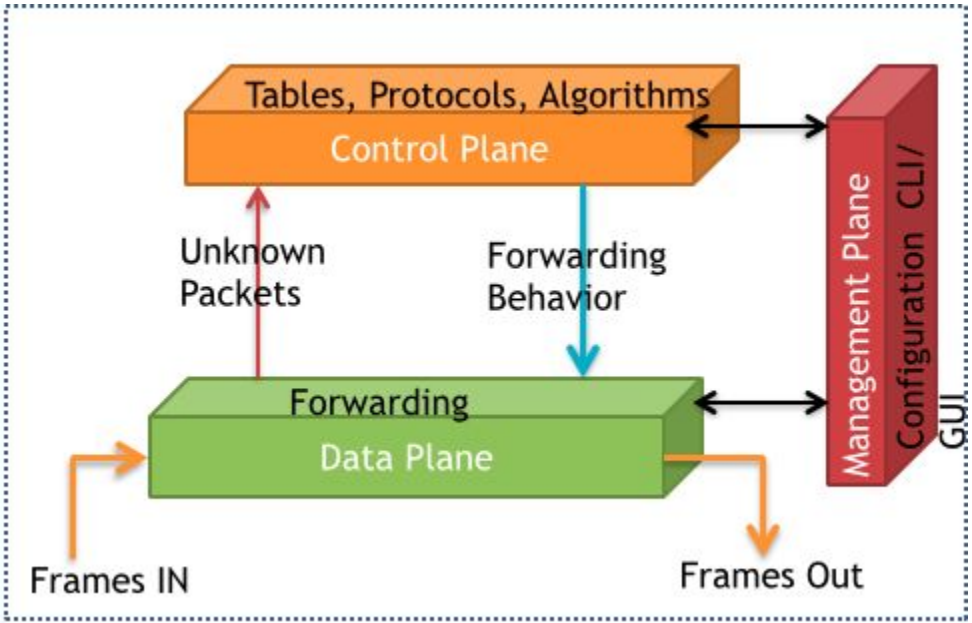
Lecture 12 Recap

- Software Defined Networking
- OpenFlow
- Readings
  - Recommended: http://yuba.stanford.edu/~casado/ethane-sigcomm07.pdf
  - Optional:
    - http://ccr.sigcomm.org/online/files/p69-v38n2n-mckeown.pdf

The Networking "Planes"

- **Data plane:** processing and delivery of packets with local forwarding state
  - Forwarding state + packet header with forwarding decision
  - Tasks: Filtering, buffering, scheduling
- **Control plane:** computing the forwarding state in routers
  - Determines how and where packets are forwarded
  - Tasks: Routing, (automatic) traffic engineering, failure detection/recovery, …
- **Management plane:** configuring and tuning the network
  - Task: (Manual) traffic engineering, ACL config, device provisioning, …

The Networking "Planes"



Timescale & Location of Each "Planes"

	Data Plane	Control Plane	Management Plane
Time-scale	Packet (nsec)	Event (10 msec to sec)	Human (min to hours)
Location	Linecard hardware	Router software	Humans or software scripts

Challenges in Networking

- Many task-specific protocols and control mechanisms
  - No modularity, limited functionality
- Indirect control mechanisms
  - Must invert protocol behavior, “coax” it to do what you want
  - E.g. Changing weights instead of the actual paths for traffic engineering
- Uncoordinated control mechanisms due to distributed nature
  - Cannot control which router updates first
- Complex interactions between protocols and mechanisms
  - Routing, addressing, access control, QoS

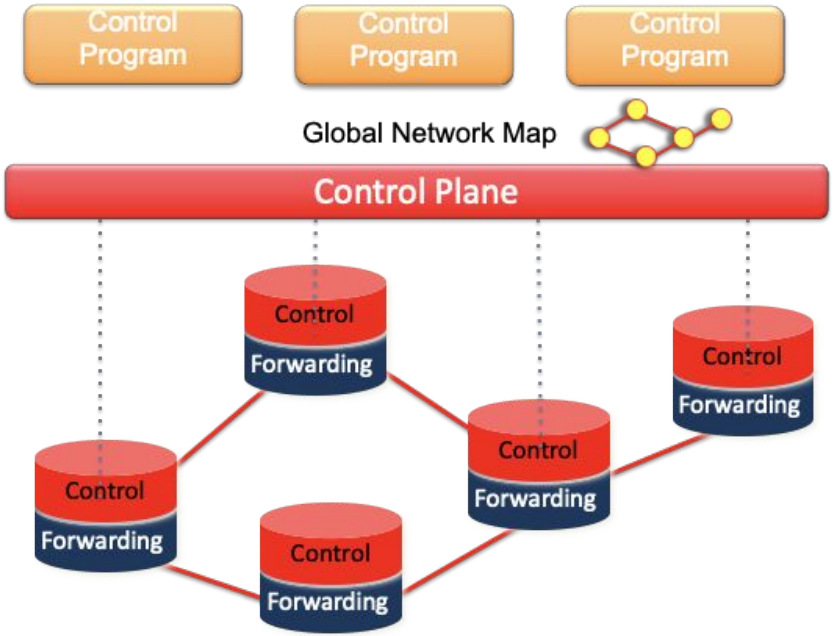
Challenges in Networking

- Therefore, Computer networks are:
  1. Hard to reason about
  2. Hard to evolve as a whole
  3. Expensive to build, operate and manage

Software Defined Networking Definition

- A network in which the **control plane is physically separate from the data plane** and **a single (logically centralized) control plane** controls several forwarding devices (data planes).
- In a simple way, having a central, physically separate controller for many routers and switches in the network.

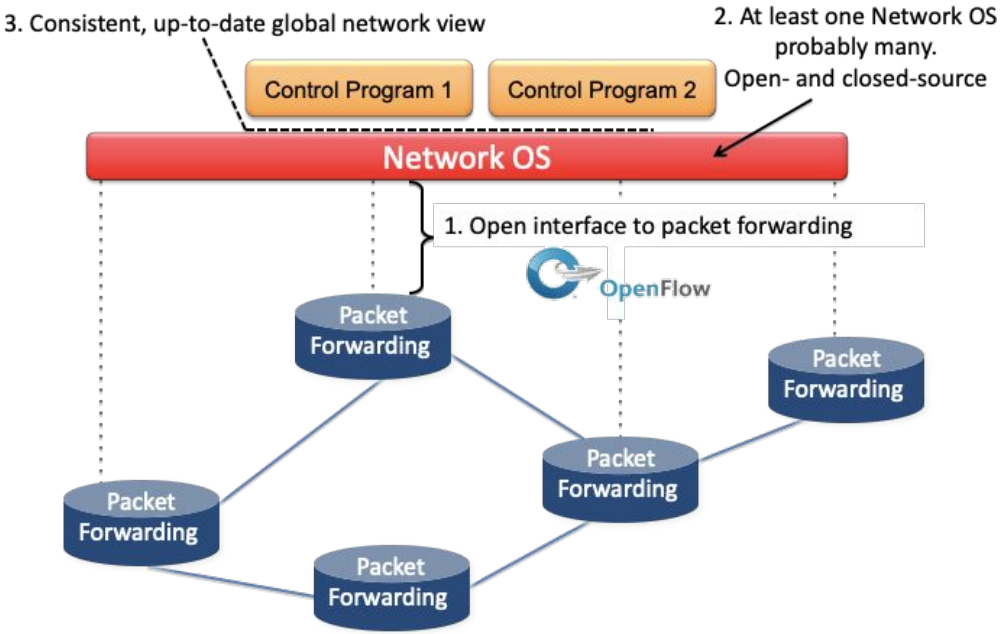
Software Defined Networking Definition



Why is this Better?

- Simpler management
  - No need to “invert” control-plane operations
- Faster pace of innovation
  - Less dependence on vendors and standards
- Easier interoperability
  - Compatibility only in “wire” protocols
- Simpler, cheaper equipment
  - Minimal software
  - Minimal hardware

Components of SDN



OpenFlow

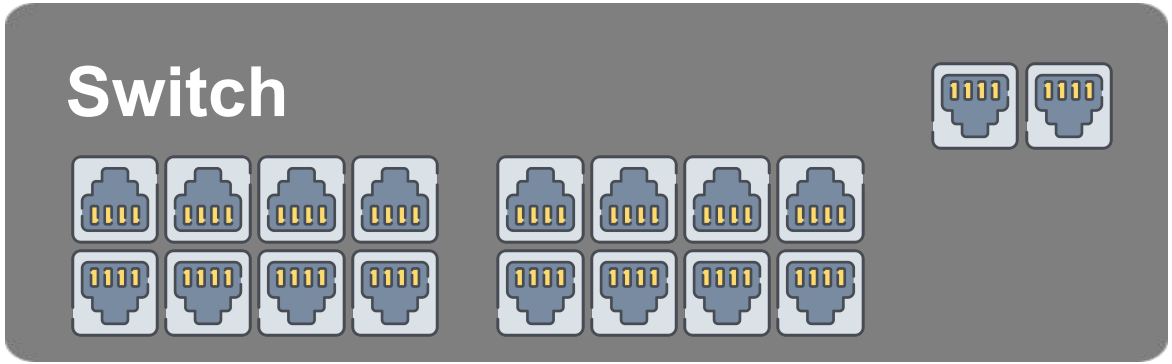
- OpenFlow allows remote management of switch rules
  - OF switches use dedicated network link to a controller
    - Controller is often a ‘normal’ server, e.g., running Linux
  - Typically for first-time packet forwarding, call out to controller
  - Controller provides resulting packet matching rules & actions
  - Establishes flow to potentially be used for subsequent packets
- OF can be easily implemented within existing switches
  - OF controller can co-exist well with existing control programs

Agenda for Today

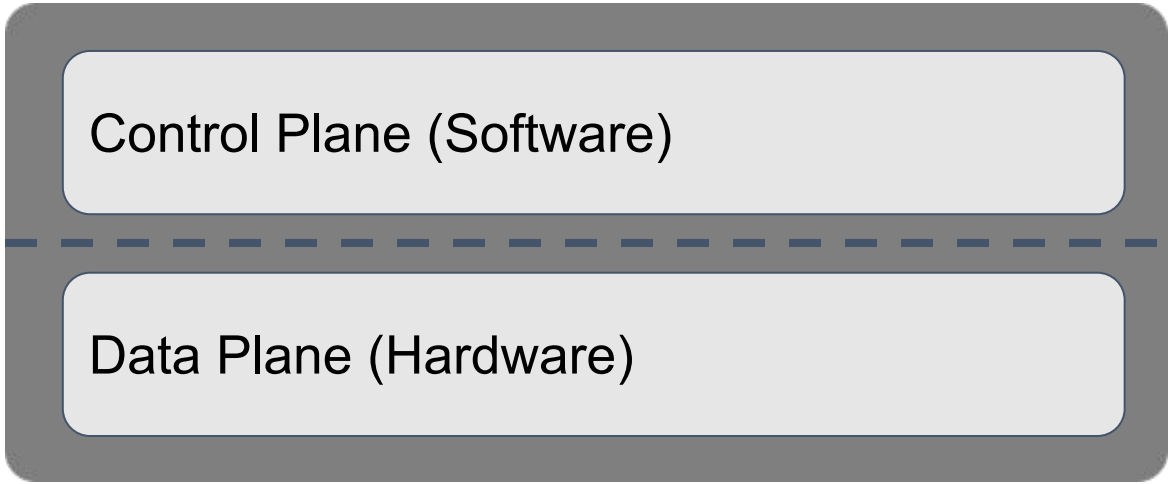
- OpenFlow Part II
- Mininet
- NFV
  - Openstack
- Readings
  - Recommended: http://conferences.sigcomm.org/hotnets/2010/papers/a19-lantz.pdf
  - Optional: https://queue.acm.org/detail.cfm?id=2560327

How does OpenFlow Work?

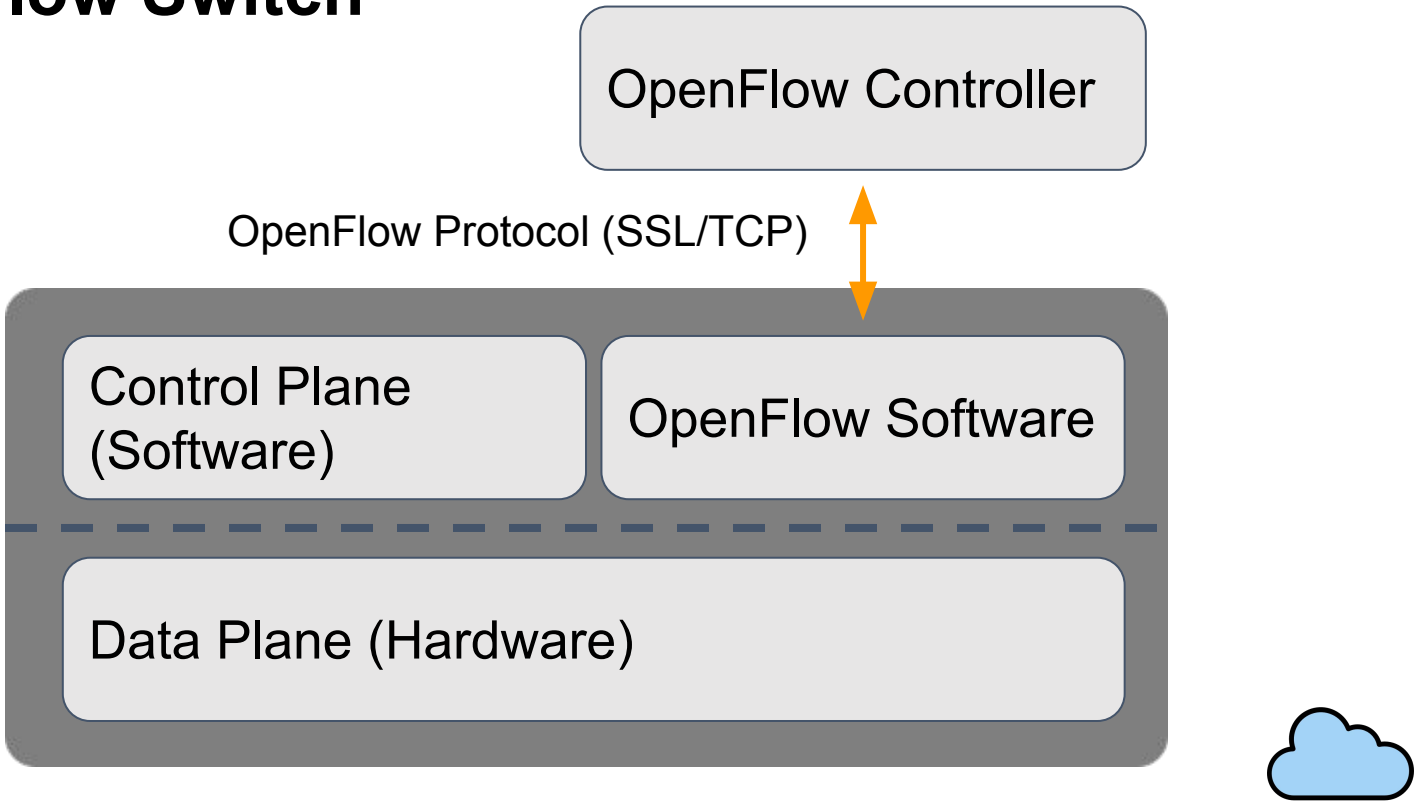
Consider a Switch



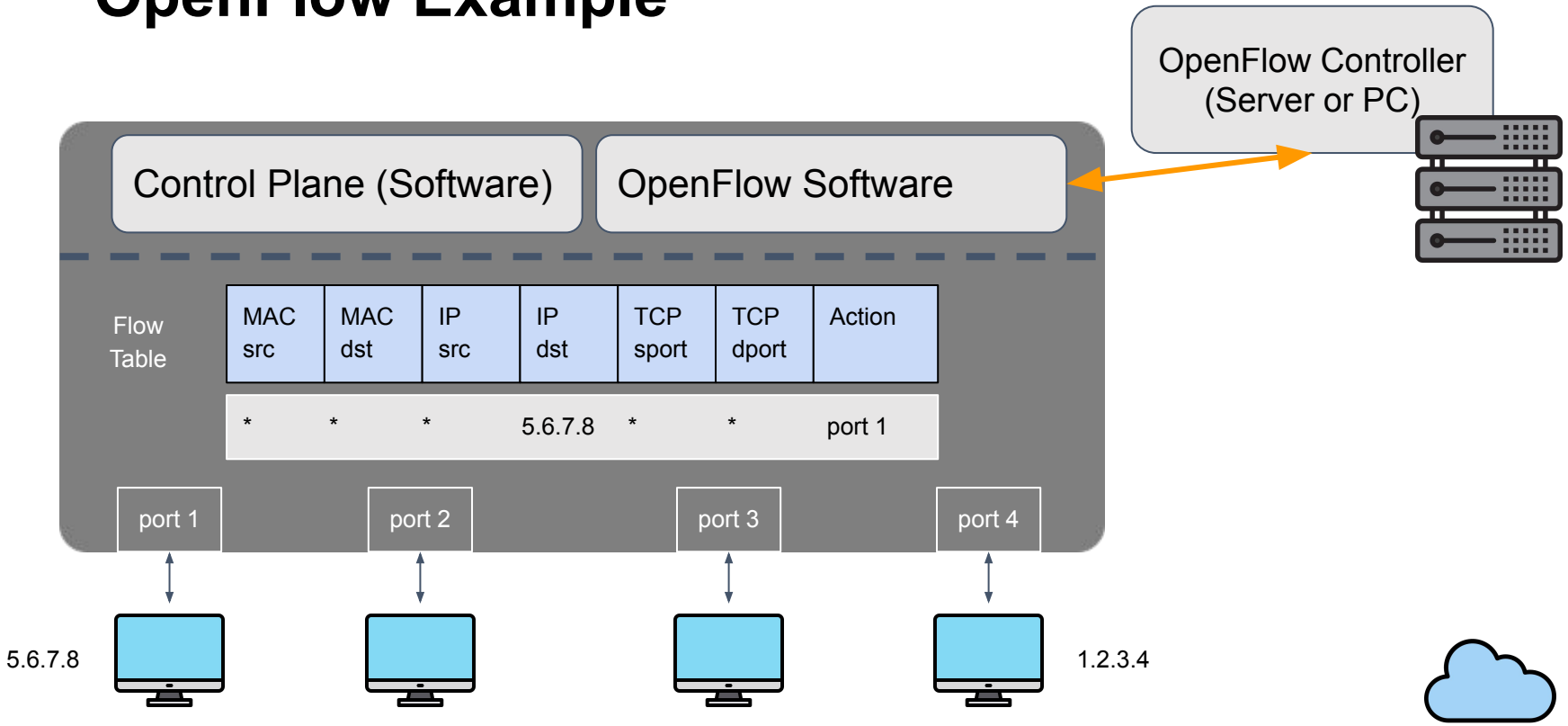
Traditional Switch



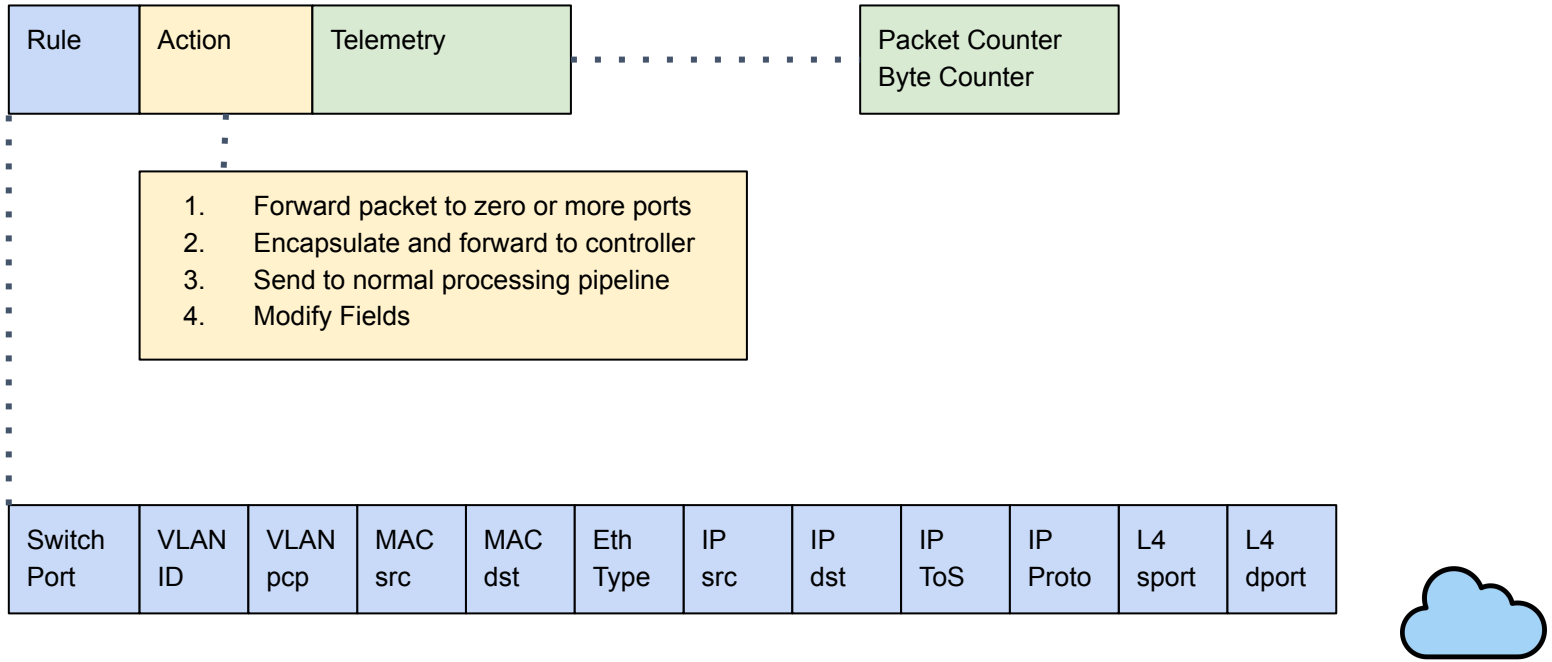
OpenFlow Switch



OpenFlow Example



OpenFlow Basics



Examples

- Switching

Switch Port	VLAN ID	VLAN pcp	MAC src	MAC dst	Eth Type	IP src	IP dst	IP ToS	IP Proto	L4 sport	L4 dport	Action
*	*	*	*	00:1f:..	*	*	*	*	*	*	*	port 4

- Firewall

Switch Port	VLAN ID	VLAN pcp	MAC src	MAC dst	Eth Type	IP src	IP dst	IP ToS	IP Proto	L4 sport	L4 dport	Action
*	*	*	*	*	*	*	*	*	*	*	22	drop

Examples

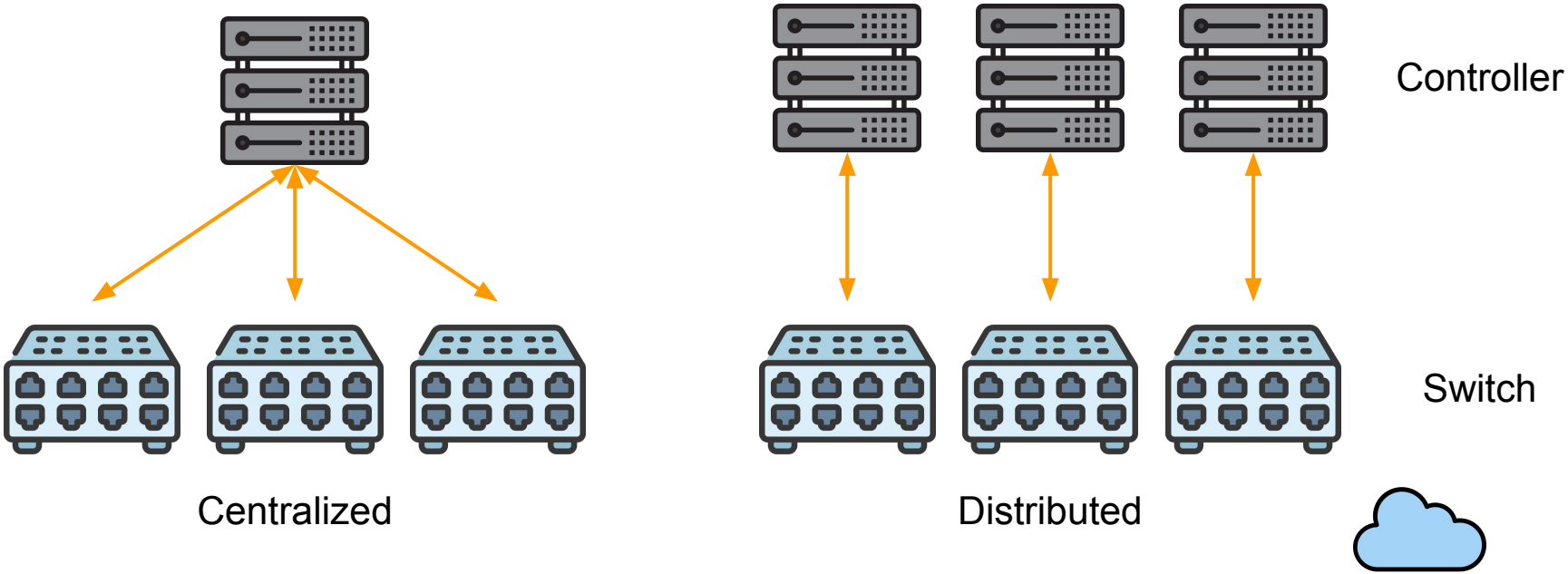
- Routing

Switch Port	VLAN ID	VLAN pcp	MAC src	MAC dst	Eth Type	IP src	IP dst	IP ToS	IP Proto	L4 sport	L4 dport	Action
*	*	*	*	*	*	*	5.6.7.8	*	*	*	*	port 1

- VLAN Switching

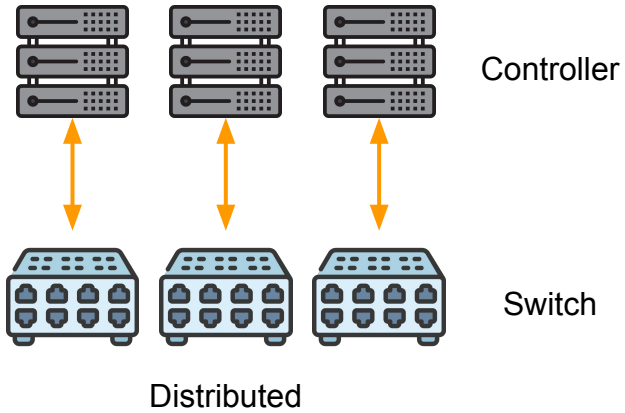
Switch Port	VLAN ID	VLAN pcp	MAC src	MAC dst	Eth Type	IP src	IP dst	IP ToS	IP Proto	L4 sport	L4 dport	Action
*	vlan 1	*	*	00:1f..	*	*	*	*	*	*	*	port 2

Centralized vs Distributed Controller



Centralized vs Distributed Controller

- Distributed Controller improves scalability
  - E.g., Many packets to capture
- Still requires lot of packets through the control channels
- How to solve this issue?
  - Modify the data plane
  - At the loss of visibility



Types of Flow Rules

- Flow-based
  - Every flow is individually set up by controller
  - Exact-match flow entries
  - Flow table contains one entry per flow
  - Good for fine grain control, e.g. campus networks

- Aggregated
  - One flow entry covers large groups of flows
  - Wildcard flow entries
  - Flow table contains one entry per category of flows
  - Good for large number of flows, e.g. backbone

Flow Rule Installation Methods

- Reactive
  - First packet of flow triggers controller to insert flow entries
  - Efficient use of flow table
  - Every flow incurs small additional flow setup time
  - If control connection lost, switch has limited utility

- Proactive
  - Controller pre-populates flow table in switch
  - Zero additional flow setup time
  - Loss of control connection does not disrupt traffic
  - Essentially requires aggregated (wildcard) rules

Mininet Overview

So now what?

- We have learned many interesting concepts regarding SDN
- But SDN is a concept that applies within a cloud
- So, how can we even test and try new technologies in SDN?
- **Mininet to the rescue!**
  - We will try mininet out for HW 3 as well

What is Mininet?

- Software for development and testing of network tools and protocols
- Creates virtual network on any type of machine (VM or native)
- Enables the following features:
  - Fast prototyping for new networking protocols
  - Simplified testing for complex topologies without buying expensive hardware
  - Realistic execution as it runs real code on the Unix and Linux kernels
  - Large open source community
- Designed for experiment in SDN

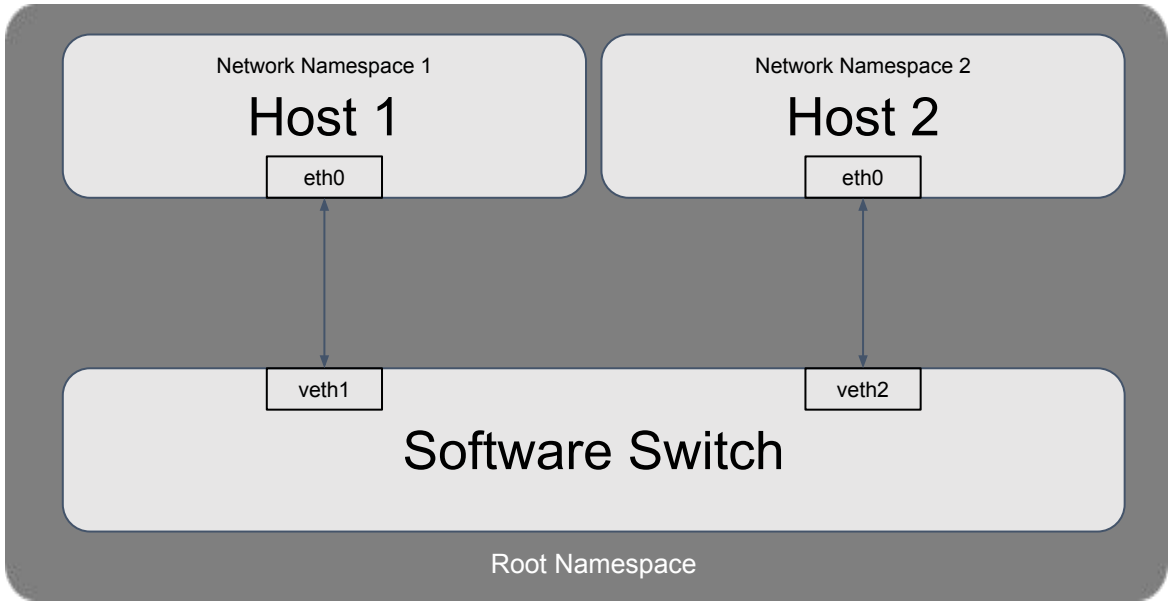
What is Mininet?

- Mininet provides network **emulation** opposed to **simulation**, allowing all network software at any layer to be simply run as is
- Mininet's logical nodes can be connected into networks
- Nodes are also called containers or network namespaces
- Nodes consume sufficiently few resources
  - Networks of over a thousand nodes have been created, running on a laptop

Mininet Nodes

- A node is a process (or group of processes) that no longer has access to all the host system's native network interfaces
- Nodes are then assigned virtual Ethernet interfaces, which are connected to other containers through a **virtual switch**
- Mininet connects a host and a switch using a virtual Ethernet (veth) link
- The veth link is analogous to a wire connecting two virtual interfaces

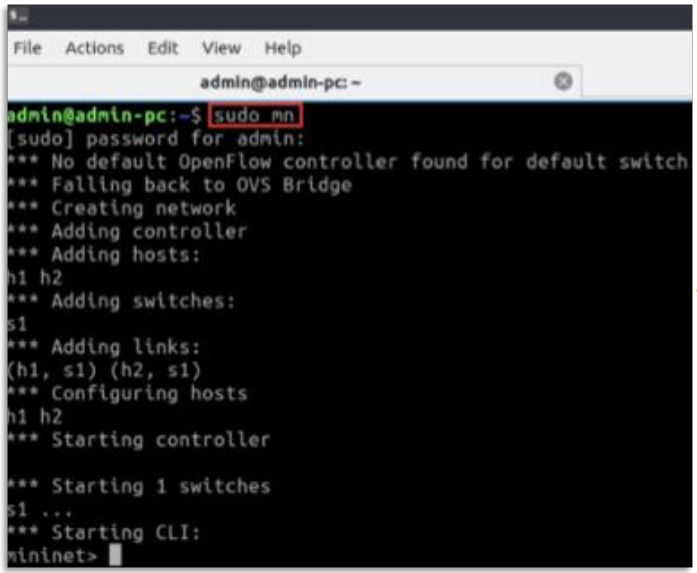
Mininet Nodes

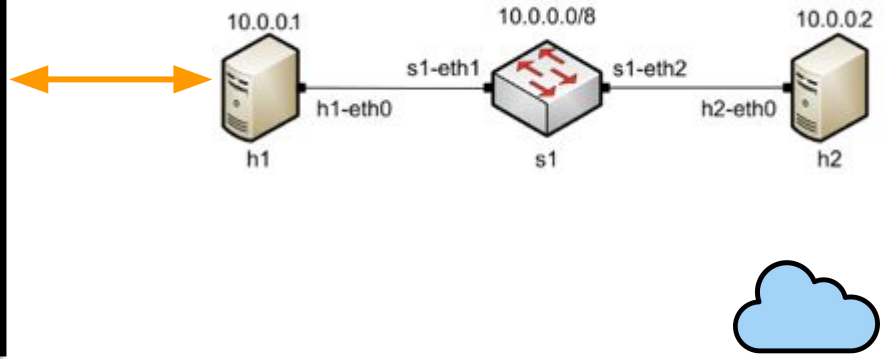


How to Use Mininet

Starting Mininet

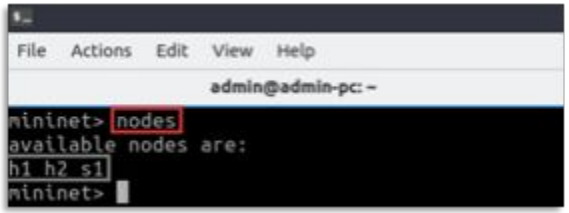
- To start a minimal topology, enter the command `sudo mn`





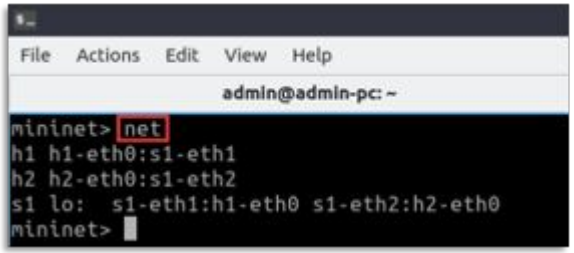
Mininet Commands

- To display the available nodes, enter the command `nodes`



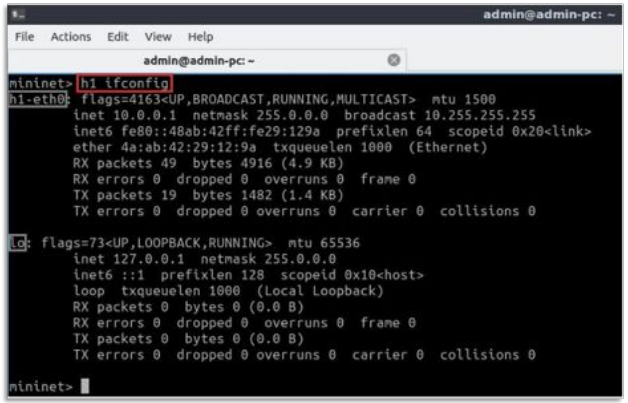
Mininet Commands

- To display the links between the devices, type `net`



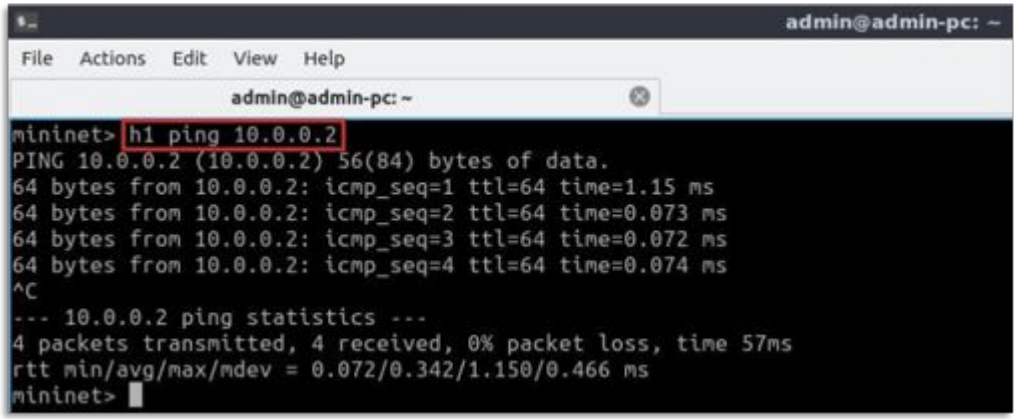
Mininet Commands

- To execute commands from a specific device, type the device first, followed by the command
- E.g., execute the command ifconfig on host h1, type `h1 ifconfig`



Mininet Commands

- To test connectivity between end-hosts, use the ping command.
- Type `h1 ping 10.0.0.2` to test the connectivity between host h1and host h2 (10.0.0.2)



Mininet Topology

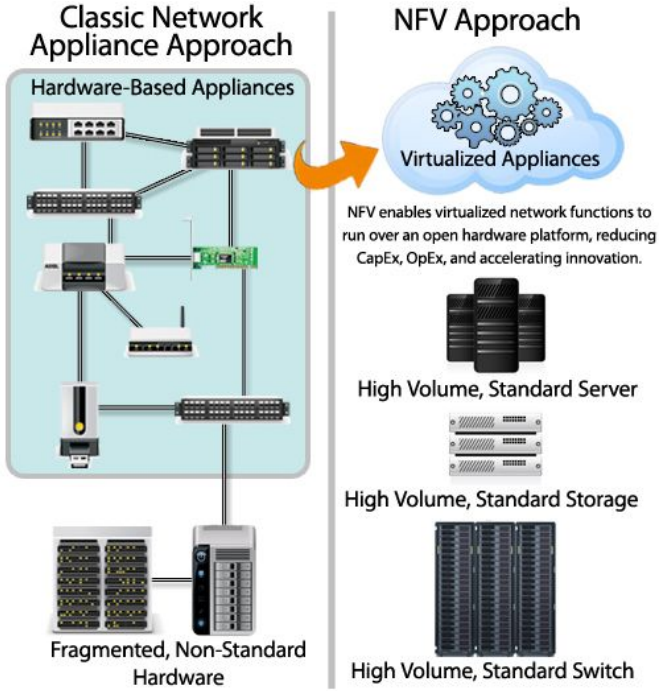
- Default topology is two hosts and one switch
- `sudo mn --topo single,3` gives single switch and three hosts
- `sudo mn --topo linear,3` gives three switch and three hosts
- `sudo mn --topo tree,fanout=2,depth=2` gives a tree topology with depth of two and fanout of two
- You can also pass in a custom topology
  - sudo mn --custom <file_name> --topo=<topology_name>
  - http://mininet.org/walkthrough/#custom-topologies

Network Function Virtualization

What is Network Function Virtualization (NFV)?

- Move network control functions from switch firmware into software
  - I.e., **Virtualize** the network function or network services
- Relies on server virtualization
  - OpenStack & VMWare are primarily used
  - Functions / Services are packaged as VMs and run on **commodity hardware**
- Widely used in Telcos
- Example of network function virtualization include:
  - **DHCP**: dynamic host configuration protocol (assign IPs)
  - **Firewalls**: filter and modify traffic to secure networks
  - **DPI**: deep packet inspection: scans packet data
  - **IDS**: intrusion detection systems scan network for attacks
  - **NTP**: network time protocol
- Not NFT

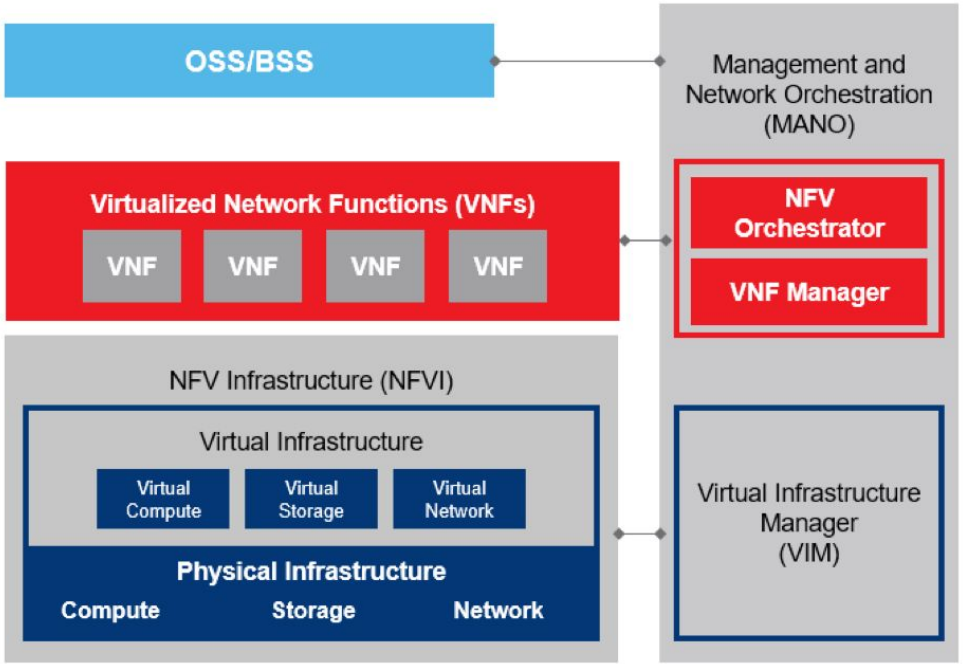
What is Network Function Virtualization?



NFV Frameworks

- There are three components to NFV framework
- Network functions virtualization infrastructure (NFVi) is all the hardware and software components that build the environment where NFVs are deployed
  - Can span several locations like a distributed system
- Virtualized network functions (VNFs) are software implementations of network functions deployed to NFVI
- **M**anagement, **a**utomation and **n**etwork **o**rchestration (MANO) is an architectural framework for managing and orchestrating VNFs and other software components

NFV Frameworks



OSS/BSS = Operational Support System / Business Support System

Pros & Cons of NFV

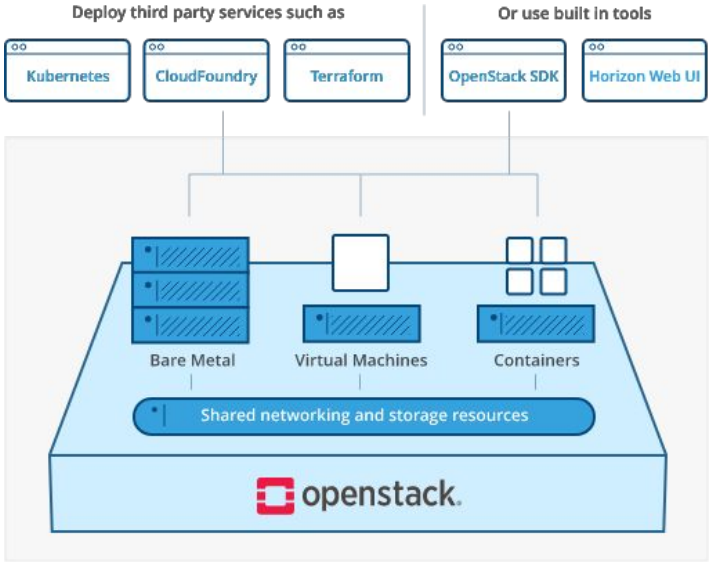
- Advantages
  - Reduced space needed for network hardware
  - Reduce network power consumption
  - Reduced network maintenance costs
  - Easier network upgrades
  - Longer life cycles for network hardware
  - Reduced maintenance and hardware costs
  - Easy to scale

Pros & Cons of NFV

- Disadvantages
  - NFV also demands a process realignment so that traditional and virtual infrastructure can be managed simultaneously
  - NFV requires managing IT in the abstract sense
  - NFV environments are more dynamic than traditional ones, which might require scaling up with additional features to cope with the speed of upgrade

Want to try out NFV?

- Way to create your own private cloud
  - https://www.openstack.org/
  - Basically having your own AWS



Want to try out NFV?

- Openstack is often used for NFV Deployment, especially in Telcos
  - Verizon: https://www.verizon.com/about/news/verizon-launches-industry-leading-large-openstack-nfv-deployment
  - Tutorial: https://www.udemy.com/course/openstack-telcocloud-asad/
- We may revisit this in Advanced Cloud Computing class

SDN & NFV

- NFV and SDN are not dependent on each other, but have similarities and differences
- NFV and SDN can be used together
- Similarities
  - Both rely on virtualization and use network abstraction, but how they separate functions and abstract resources is different.
  - Both use commodity hardware
- Differences
  - SDN separates network forwarding functions from network control functions with the goal of creating a network that is centrally manageable and programmable.
  - NFV abstracts network functions from hardware. NFV supports SDN by providing the infrastructure on which SDN software can run.

TODOs!

- HW 3
- Quiz 3

Agenda for Today

- OpenFlow Part II
- Mininet
- NFV
  - Openstack
- Readings
  - Recommended: http://conferences.sigcomm.org/hotnets/2010/papers/a19-lantz.pdf
  - Optional: https://queue.acm.org/detail.cfm?id=2560327

Questions?