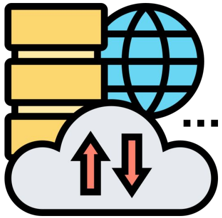
COEN 241
Introduction to Cloud Computing

Lecture 12 - Software Defined Networking



Midterm Feedback

- It would be awesome if you can fill out this survey!
  - https://forms.gle/HQ7xxptrNdkYy49p6

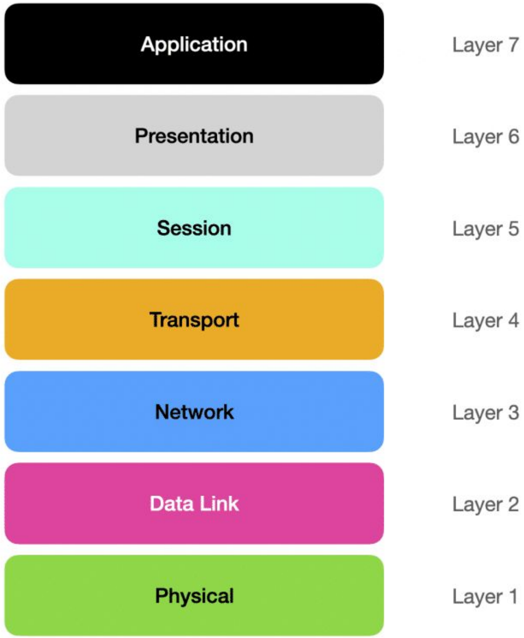
Lecture 11 Recap

- Networking 101
- Data Center Networking

OSI Model : Building Block of Networking

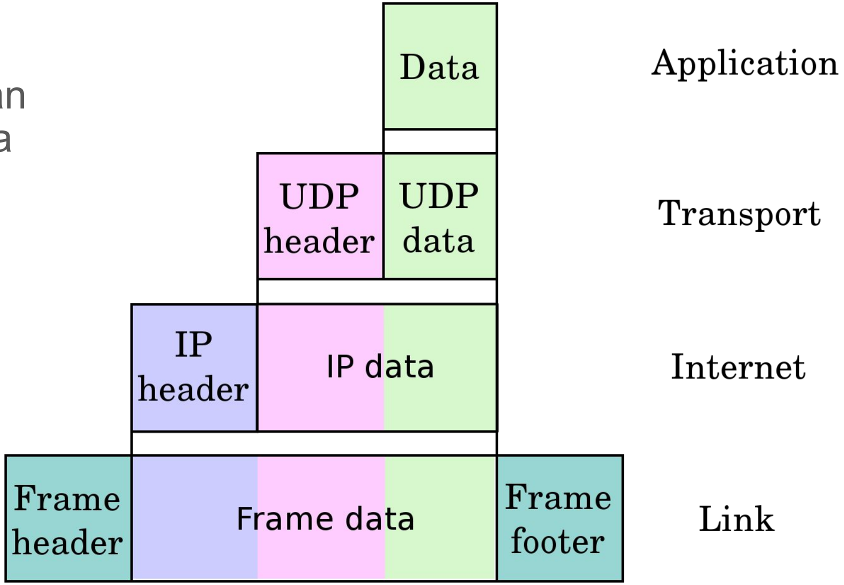
- Open Systems Interconnection provides a standard for different computer systems to be able to communicate with each other
- Can be seen as a universal language for computer networking
- It’s based on the concept of splitting up a communication system into **seven abstract layers**, each one stacked upon the last
- Each layer has different **protocols** implementations
    - **Protocols:** An established set of rules that determine how data is transmitted between different devices in the same network
    - Example: HTTP Status Codes: 200, 401, 500
HTTP Commands: GET, POST

OSI Model : Building Block of Networking



Data Carried at Different Layers

- Data for applications using IP is augmented at each layer below
    - Ethernet frames’ total size larger than IP packets’ total size for a given data
- UDP shown here transports datagrams (chunks of data)
- TCP instead transports coherent streams of data
    - Includes retransmissions and congestion control



Type of Traffic and Workloads in a Data Center

- Traffic **to/from Internet**
  - The node-to-node internal network use is small
- Consider scale-out applications within a DC
  - Interacting servers do so with distributed effect & high volume
  - Application dependent
    - MapReduce, Storage Replication
- New ‘interesting’ traffic patterns: (i.e., non-unicast)
  - Anycast: traffic reaches any ‘nearby’ applicable host
  - Multicast or 1–N: traffic is being sent to multiple hosts simultaneously

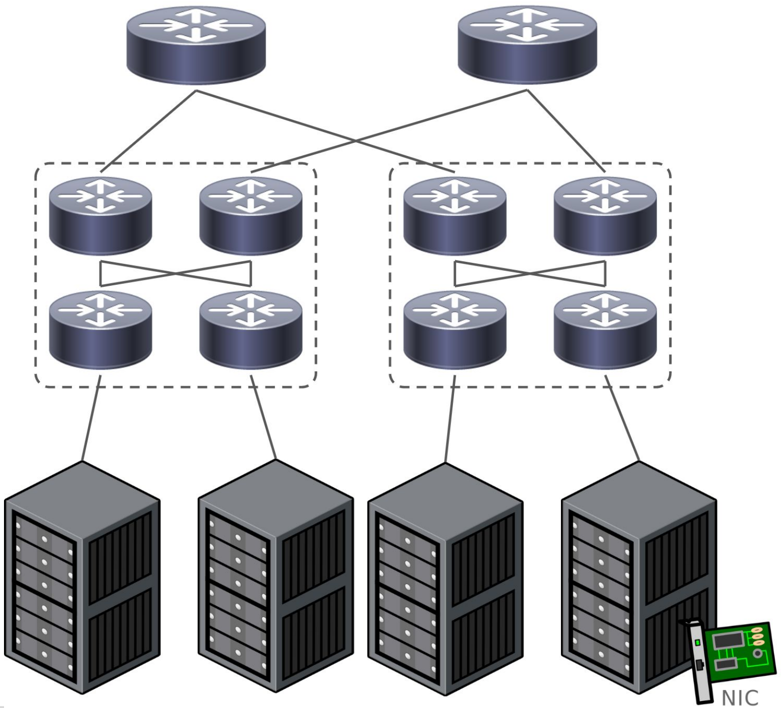
Data Center Networks

- Once users' packet reaches the data center, it enters the DC network

- Data centers have a structured physical layout
  - Physical servers are grouped into racks, which usually provide:
    - Intra-rack network switches; power distribution; wiring management
  - Racks grouped into aisles or zones—for maintenance / access

- DC networking is hierarchically organized:
  - Core router receiving data from the internet
  - Aggregation layer receiving data from the core
  - Top-of-rack switches installed within racks
  - **Virtual networking within physical servers themselves**

Data Center Networks



Core Switch

Aggregation Switch

Edge/ToR Switch

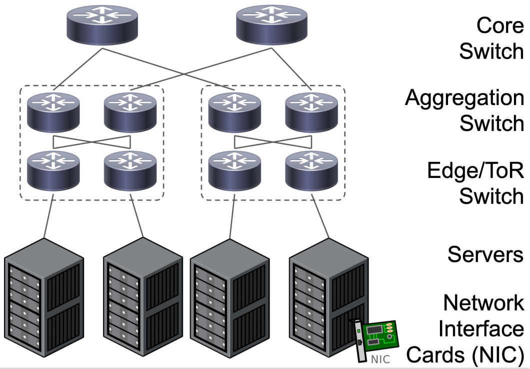
Servers

Network Interface Cards (NIC)

Data Center Network Topology (Fat-Tree)

- The three-tier, fat-tree topology mentioned is now outdated

- Multiple issues with the three-tier design
  - Upper-level routers: highly specialized and expensive
    - Designed to sustain high bandwidth in all directions
    - Core routers' prices in the order of $100,000 a piece
  - High energy usage
  - Not well suited for intra-DC traffic (why?)

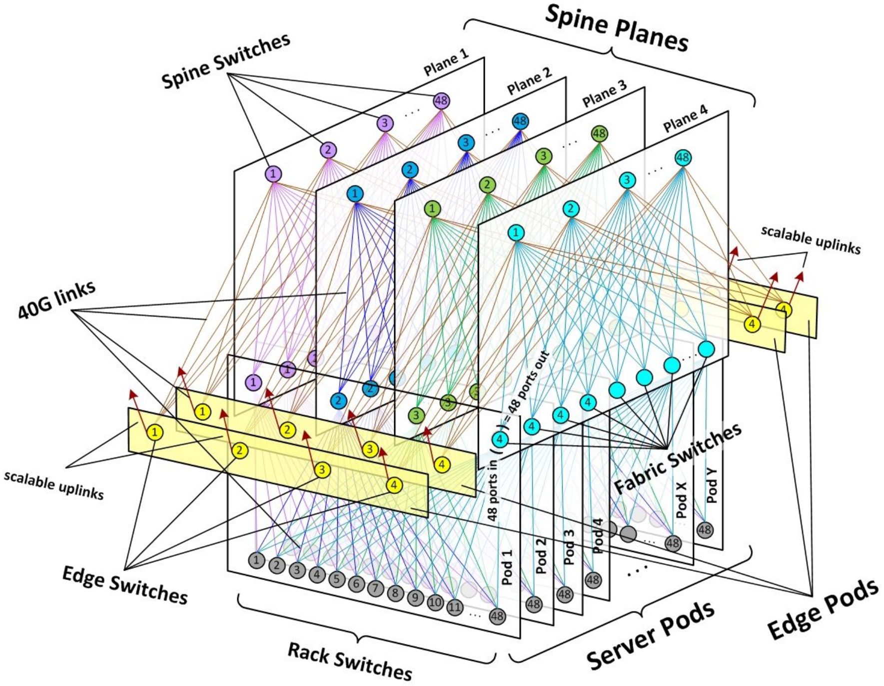


Data Center Network Topology (Clos)

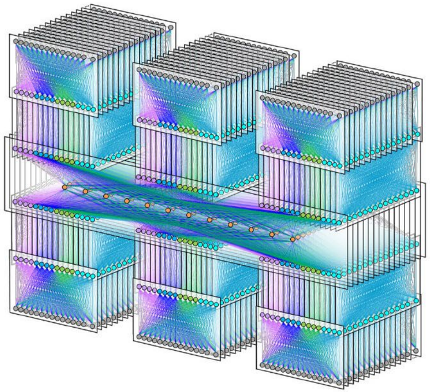
- Modern trend is toward using commodity switching kit
  - Create an aggregation switch from a set of cheaper switches
  - Employs a particular addressing scheme and routing algorithm
  - To build and configure by the DC operators

- Often related to a topology known as a Clos network
  - Have ingress, middle and egress stages built from switches
  - The structure can be used recursively (expand middle)
  - Simpler and flatter

Data Center Network Topology (Clos)

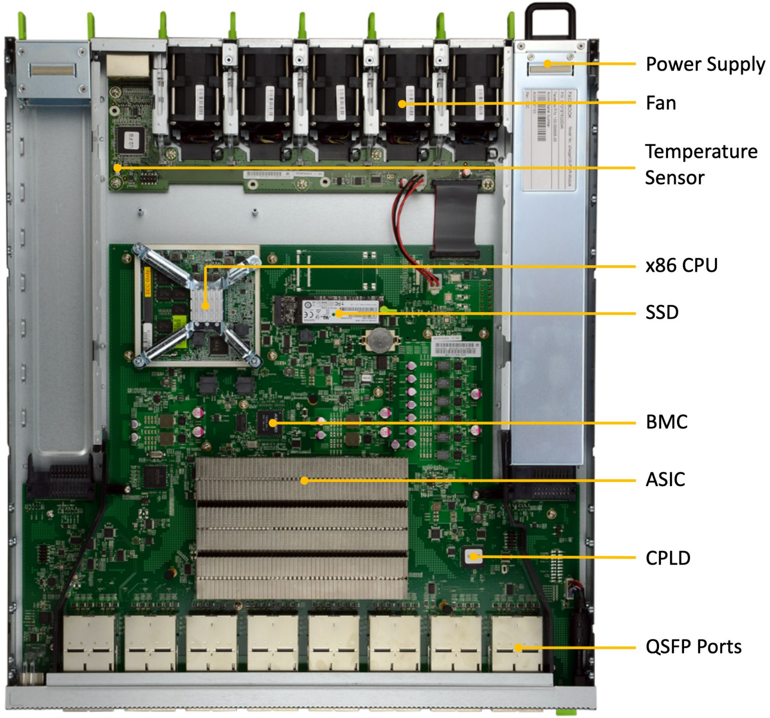


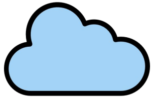
Data Center Network Topology (Clos)



https://engineering.fb.com/2019/03/14/data-center-engineering/f16-minipack/

Data Center Switch Architecture

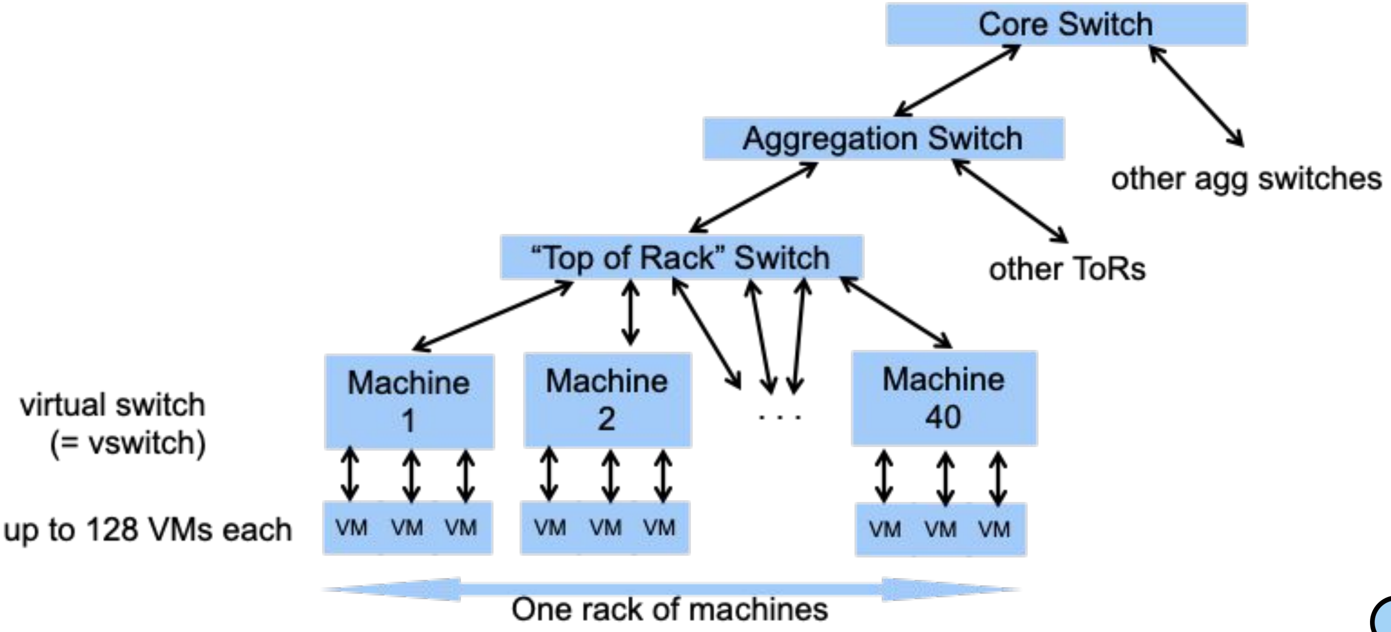




Virtual Switch

- Switch within a server that switches packets between VMS
  - Another big part of Data Center networking and **Software-Defined Networking**
- **Data Plane**
  - Software that is responsible for actual packet switching
  - Implemented in Intel DPDK and/or in Kernel modules
- **Control Plane**
  - Software that is responsible for making routing decisions
- Examples:
  - Open vSwitch (OvS)

Virtual Switch: DC Network After VMs



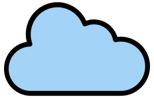
Agenda for Today

- 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

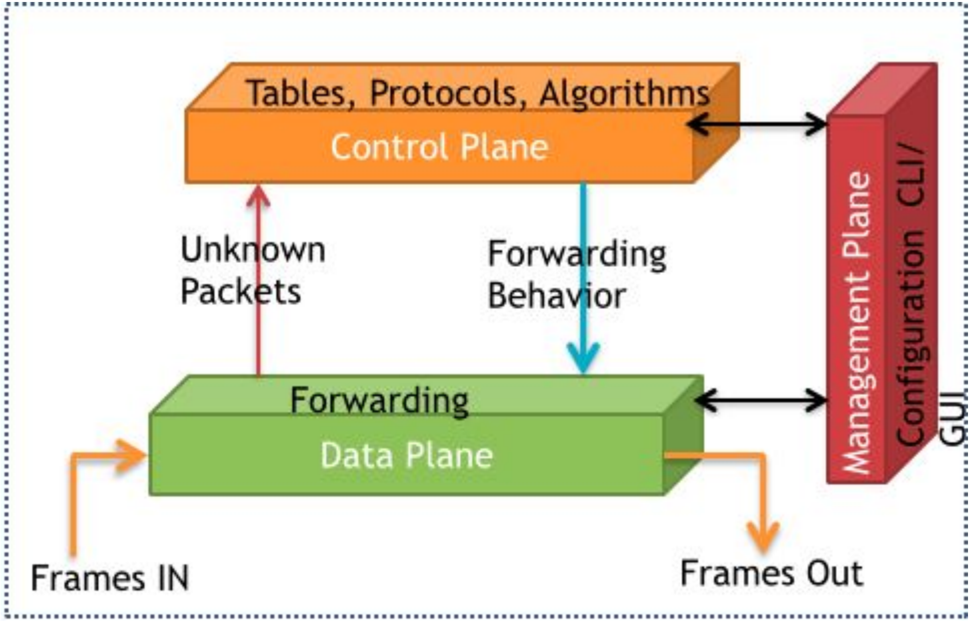
“Planes” in Networking

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, …

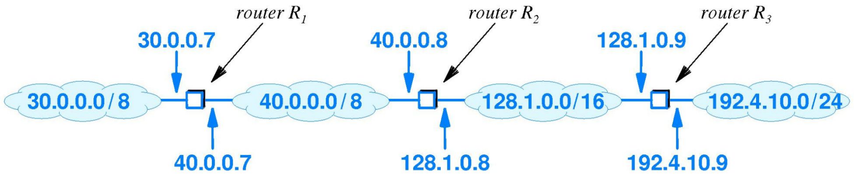


The Networking “Planes”



Data Plane

- Also known as the forwarding plane
- Streaming algorithms on packets
  - Matching on some header bits
  - Perform some actions
- Example
  - IP Forwarding
  - Firewall
  - etc...

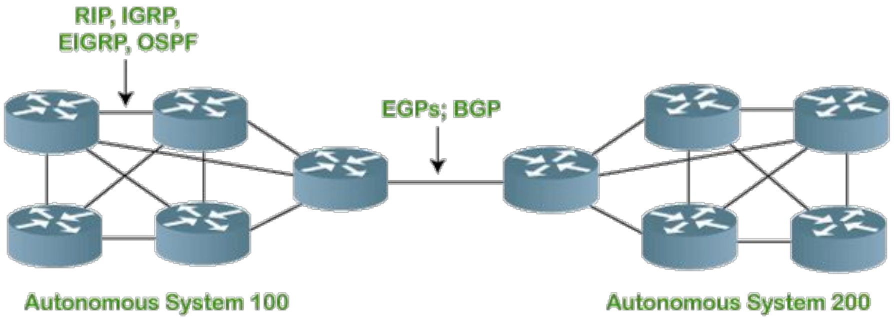


(a)

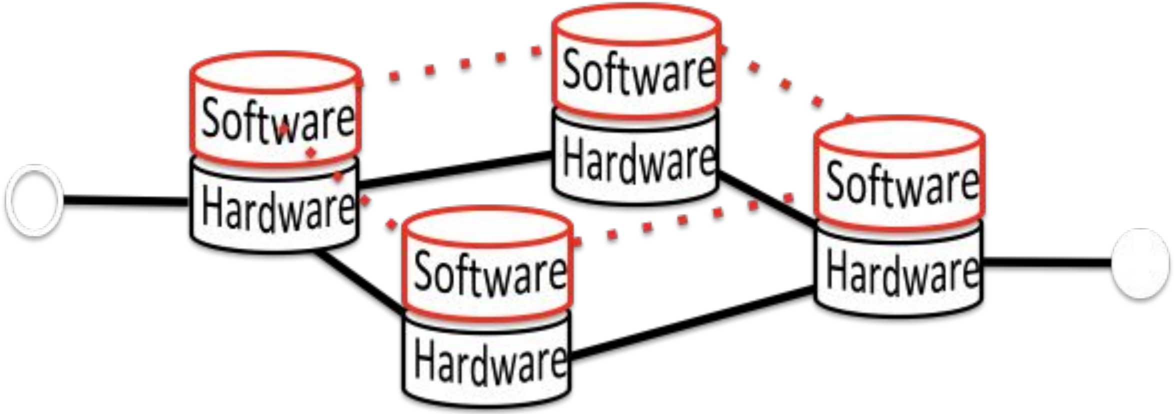
Destination	Mask	Next Hop
30.0.0.0	255.0.0.0	40.0.0.7
40.0.0.0	255.0.0.0	deliver direct
128.1.0.0	255.255.0.0	deliver direct
192.4.10.0	255.255.255.0	128.1.0.9

Control Plane

- Compute paths the packets will follow
  - Populate forwarding tables
  - Implement routing protocols, which are often calculated in a distributed way
- Example: Link-state routing (OSPF, IS-IS), BGP
  - Flood the entire topology to all nodes
  - Each node computes shortest paths
  - Dijkstra's algorithm

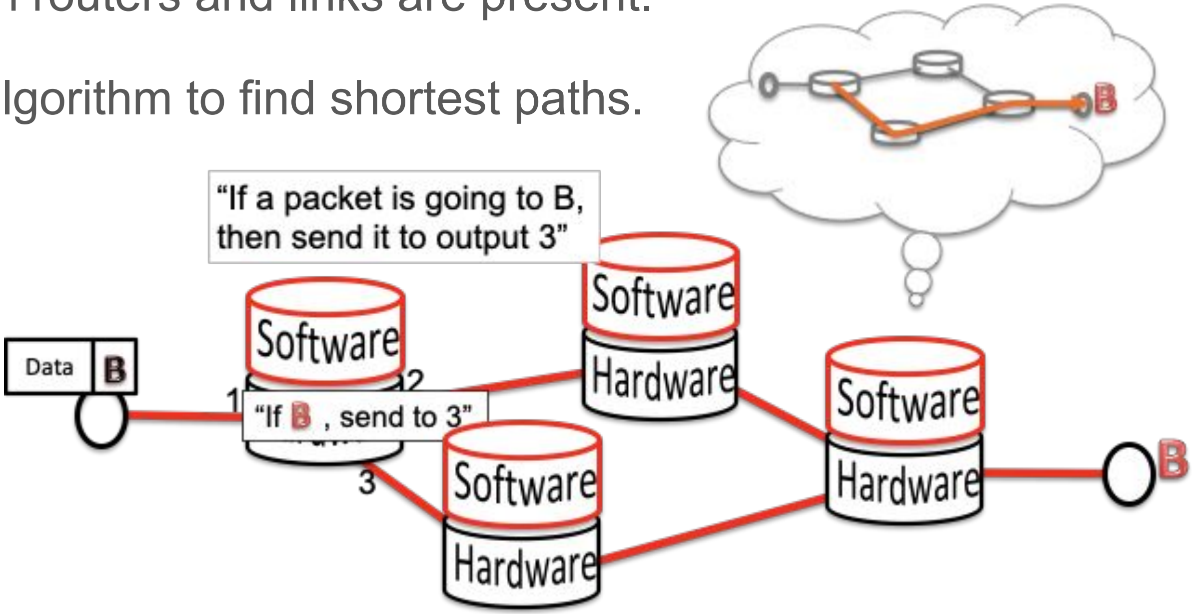


Routing Protocol Example



Routing Protocol Example

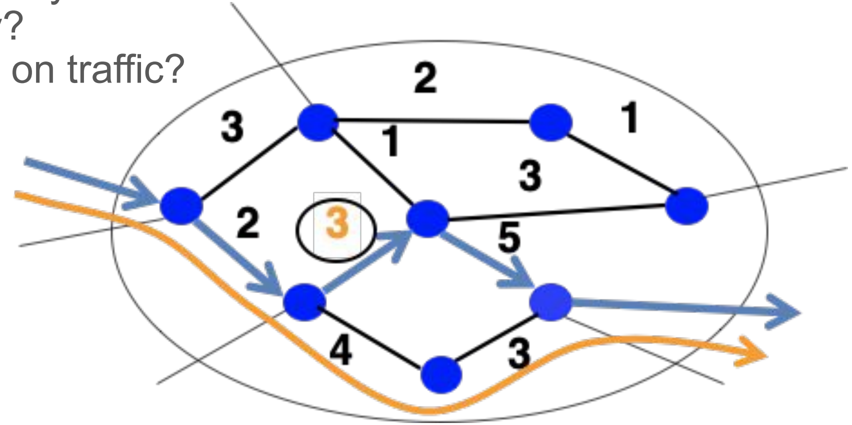
- Figure out which routers and links are present.
- Run Dijkstra's algorithm to find shortest paths.



Management Plane

- Traffic Engineering: Setting the weights for each links
    - To be used for the routing protocols

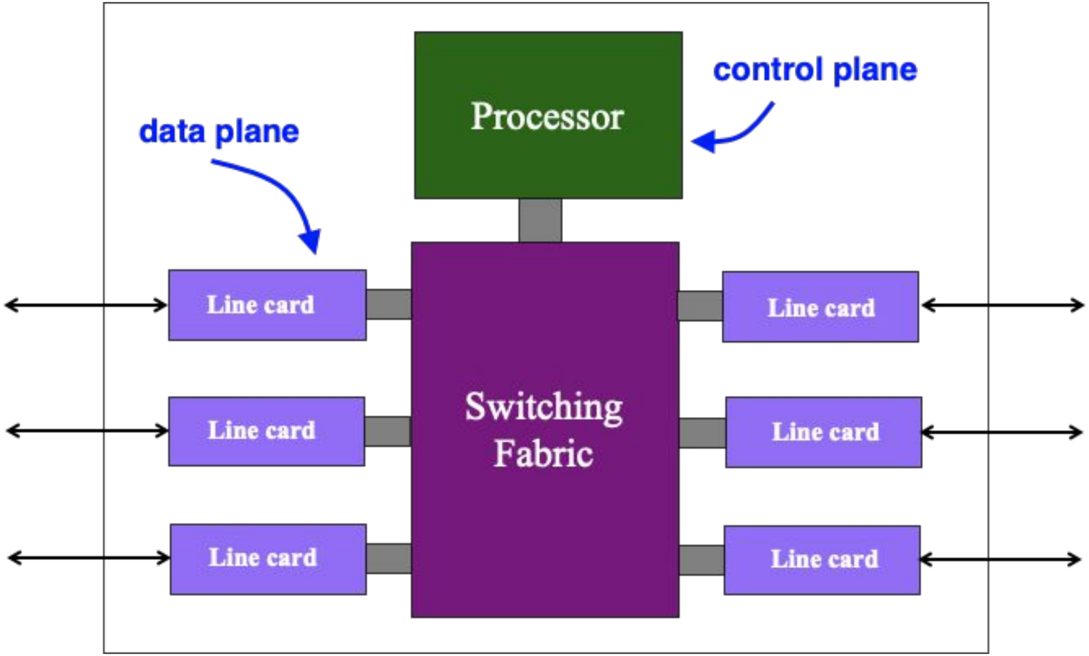
- How to set the right weights?
    - Inversely proportional to link capacity?
    - Proportional to propagation delay?
    - Network-wide optimization based on traffic?



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

Timescale & Location of Each “Planes”



Intro to
Software Defined Networking

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

An Example: Inter-domain Routing

- Today's inter-domain routing protocols, e.g., BGP, artificially limit routes
  - Routing depend only on **destination IP address blocks**
  - Can only influence **immediate neighbors**
  - Very difficult to incorporate other information
- Application-specific peering
  - Route video traffic one way, and non-video another
- Blocking denial-of-service traffic
  - Dropping unwanted traffic further upstream
- Inbound traffic engineering
  - Splitting incoming traffic over multiple peering links

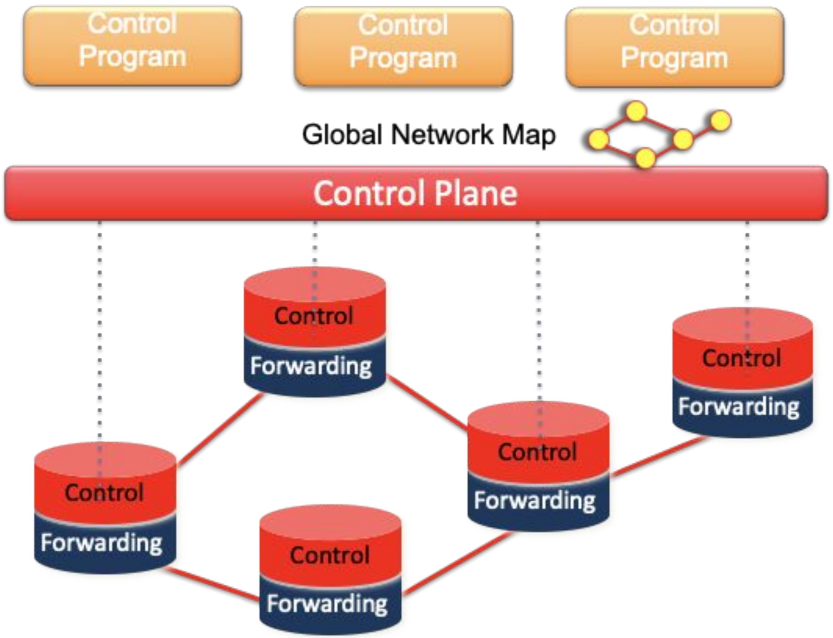
An Innovation from Stanford

- In 2006, **OpenFlow** was proposed, which provides an open protocol to program the flow-table in different switches and routers.
- In 2007, Nicira was founded by Martin Casado, Nick McKeown and Scott Shenker. This company focuses on software defined networking and network virtualization. It was acquired by VMware in 2012
- In 2007, a SIGCOMM paper named Ethane was published, presenting the initial ideas of **Software Defined Networking (SDN).**

Software Defined Networking Definition

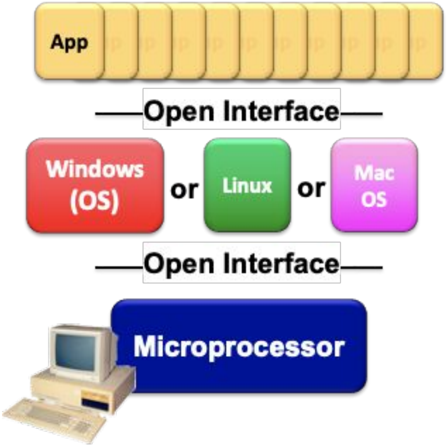
- A network in which the **control plane is physically separate from the data plane** and **a single (logically centralized) control plane** controls several forwarding devices (data planes).
- In a simple way, having a central, physically separate controller for many routers and switches in the network.

Software Defined Networking Overview



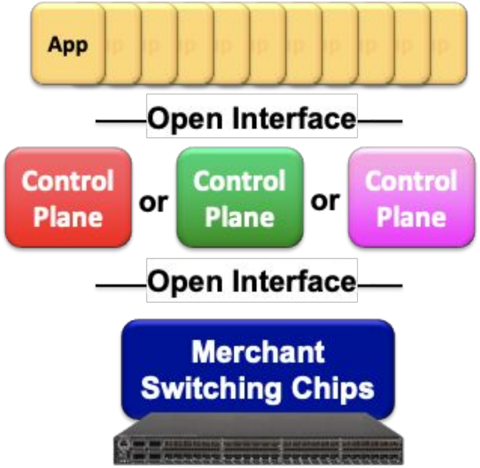
Helpful Analogy: Mainframe Computers

Vertically integrated
Closed, proprietary
Slow innovation
Small industry



Horizontal
Open interfaces
Rapid innovation
Huge industry

Helpful Analogy: Switches and Routers



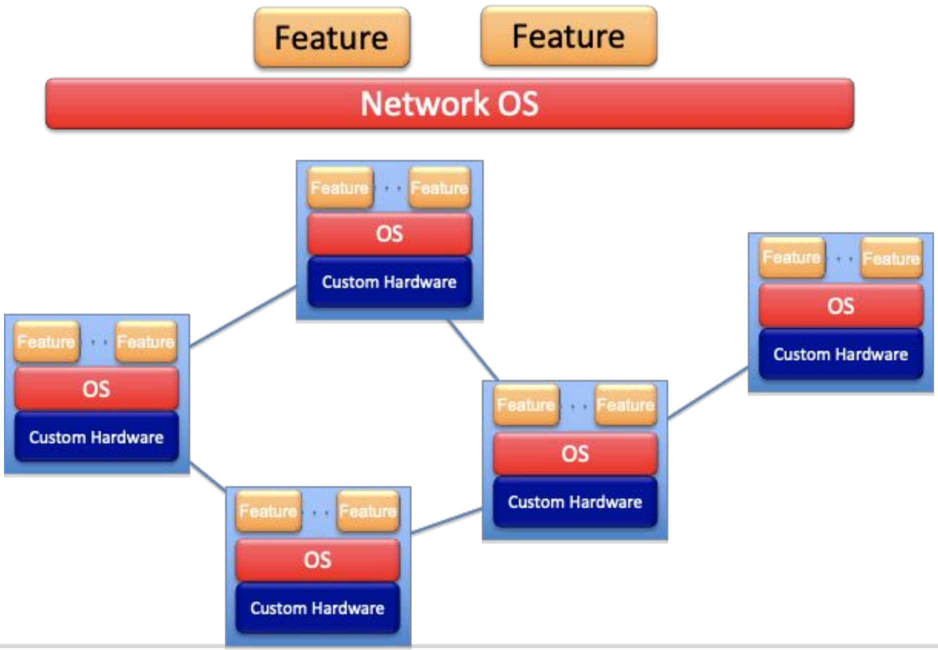
Vertically integrated
Closed, proprietary
Slow innovation

Horizontal
Open interfaces
Rapid innovation

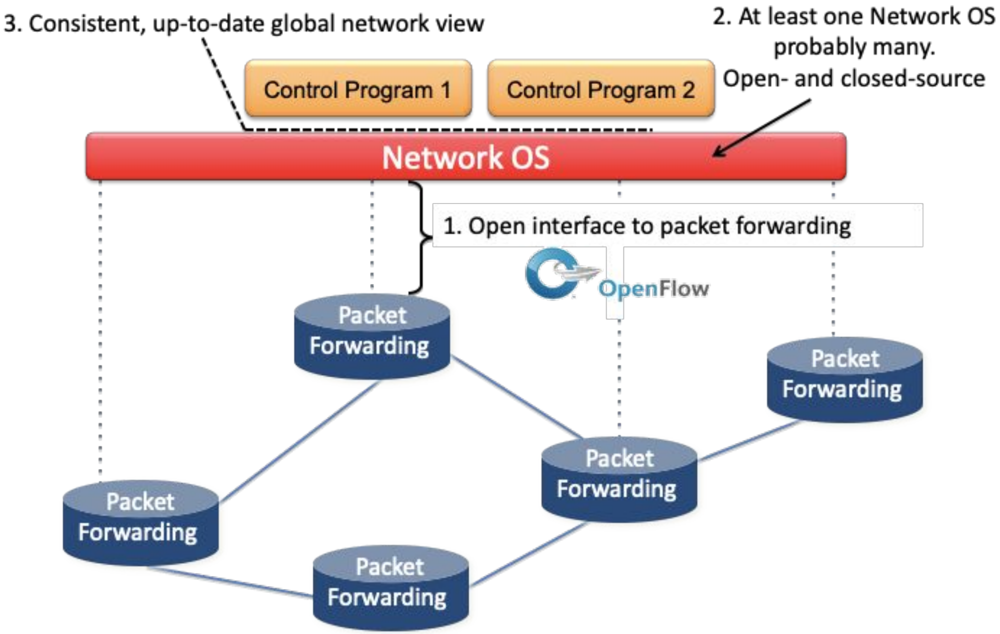
Why is this Better?

- 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

How SDN Changes the Network



Components of SDN



Forwarding Abstraction (with Open Interface)

- A standard way of defining forwarding state and communicating the state to the 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 of such an abstraction

Network Operating System

- Network OS: A distributed system that creates a consistent, up-to-date network view
  - Runs on servers (controllers) in the network
  - NOX, ONIX, FBOSS, Floodlight, Trema, OpenDaylight, HyperFlow, Kandoo, Beehive, Beacon, Maestro, … + more
- Uses forwarding abstraction to:
  - Get state information from forwarding elements
  - Give control directives to forwarding elements

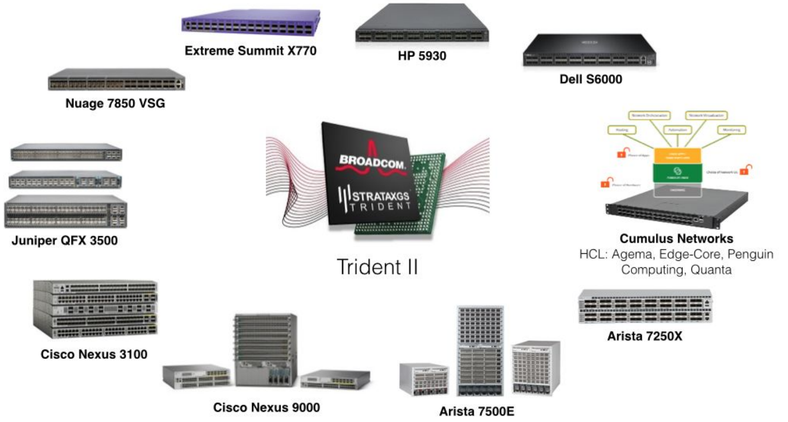
Control Program

- Control program operates on view of network
  - Input: global network view (graph/database)
  - Output: configuration of each network device
- Control program is not a distributed system
  - Abstraction hides details of distributed state

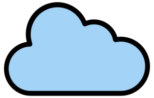
Does SDN Really Simplify the Network?

- Some are skeptical that simplicity can be sustained in the long run
  - Basic paradigm in networks (layers) is in fact a simple model.
  - The ever-changing performance requirements and functionality goals have forced more complexity into network design
- Abstraction doesn’t really eliminate complexity
  - NOS, Hypervisor are still complicated pieces of code
- However, SDN achieved the following:
  - Simpler interface for control program (user-specific)
  - Pushed complexity into reusable code (SDN platform)
- Analogous to how compilers are written

Why is SDN so Important?



- Rise of merchant switching silicon
    - Democratized switching
    - Vendors eager to unseat incumbents
- Cloud / Data centers
    - Operators face real network management problems
    - Extremely cost conscious; desire a lot of control
- The right balance between vision & pragmatism
    - OpenFlow compatible with existing hardware
- **A “killer app”: Network virtualization**



Why is SDN so Important?

- Consider a multi-tenant datacenter
  - Want to allow each tenant to specify virtual topology
  - This defines their individual policies and requirements
- Data center’s network hypervisor compiles these virtual topologies into set of switch configurations
  - Takes 1000s of individual tenant virtual topologies
  - Computes configurations to implement all simultaneously
- This is what people are paying money for….
  - Enabled by SDN’s ability to virtualize the networks

Practical Challenges in Implementation

- Scalability
  - Decision elements responsible for many routers
- Reliability
  - Surviving failures of decision elements and routers
- Response time
  - Delays between decision elements and routers
- Consistency
  - Ensuring multiple decision elements behave consistently
- Security
  - Network vulnerable to attacks on decision elements
- Interoperability
  - Legacy routers and neighboring domains

OpenFlow

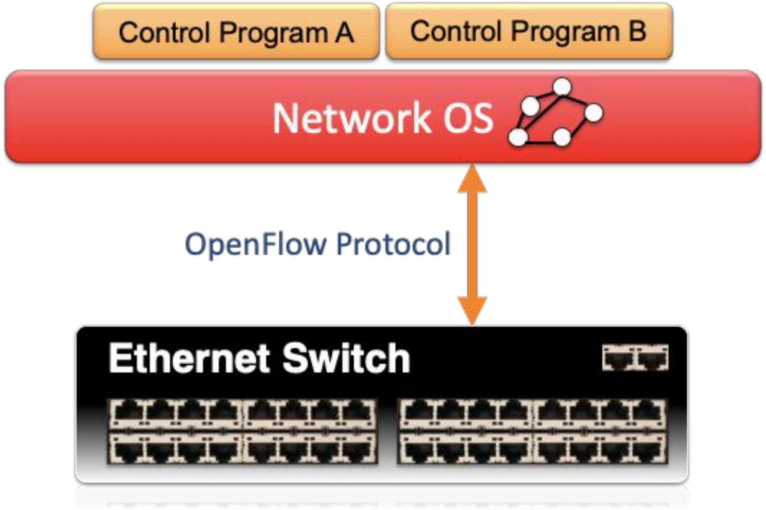
OpenFlow

- In 2006, OpenFlow was proposed, which provides an open protocol to program the flow-table in different switches and routers.
- People can try new routing protocols and security models by a software controller
- http://ccr.sigcomm.org/online/files/p69-v38n2n-mckeown.pdf

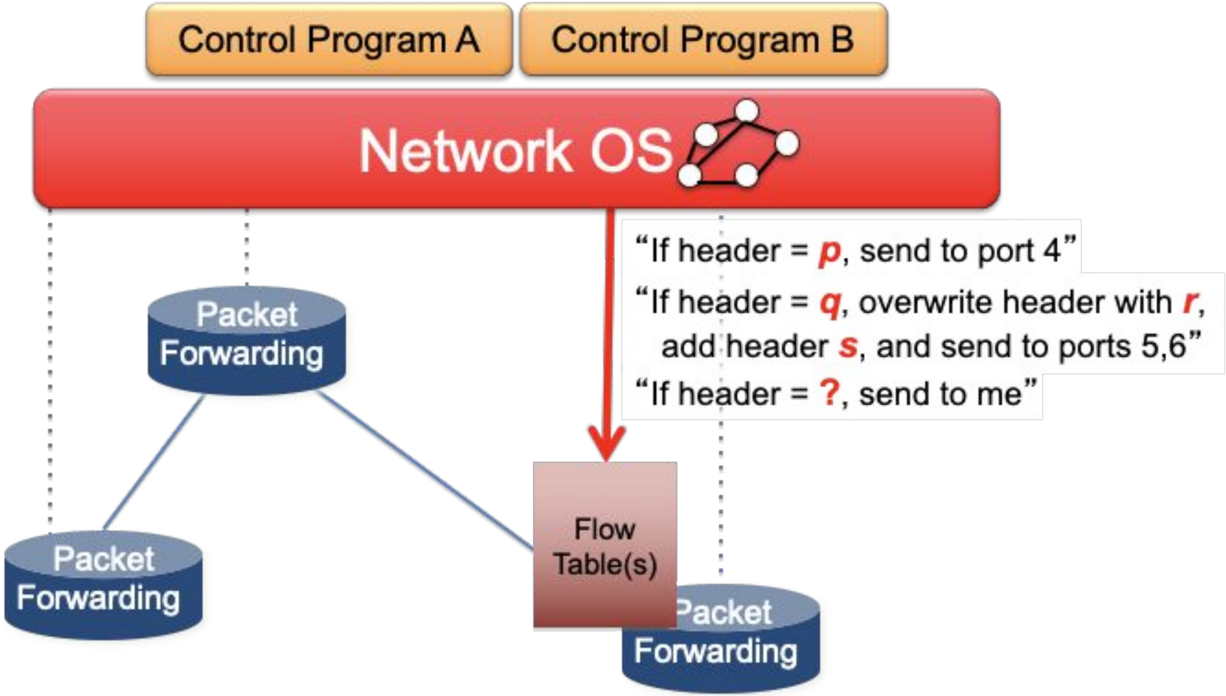
OpenFlow

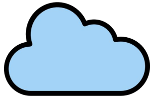
- OpenFlow allows remote management of switch rules
  - OF switches use dedicated network link to a controller
    - Controller is often a ‘normal’ server, e.g., running Linux
  - Typically for first-time packet forwarding, call out to controller
  - Controller provides resulting packet matching rules & 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

OpenFlow


Control Program A
Control Program B
Network OS
OpenFlow Protocol
Ethernet Switch

OpenFlow


Control Program A
Control Program B
Network OS
"If header = p, send to port 4"
"If header = q, overwrite header with r, add header s, and send to ports 5,6"
"If header = ?, send to me"
Packet Forwarding
Packet Forwarding
Flow Table(s)
Packet Forwarding



TODOs!

- Final Project
- Quiz 3 is out at the end of class
- HW 3 is out

Agenda for Today

- 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

Questions?