Data-intensive computing systems

Data Centers

University of Verona
Computer Science Department

Damiano Carra

Acknowledgements

❑ *Credits*

- *Part of the course material is based on slides provided by the following authors*
  - *Data centers: Andreas Haeberlen, Zachary G. Ives*
  - *SDN: Jennifer Rexford, Nick McKeown*
  - *Cloud OS: Matei Zaharia, Sanjay P. Ahuja*

Datacenter general overview

Scaling up

- What if one computer is not enough?
  - Buy a bigger (server-class) computer
- What if the biggest computer is not enough?
  - Buy many computers

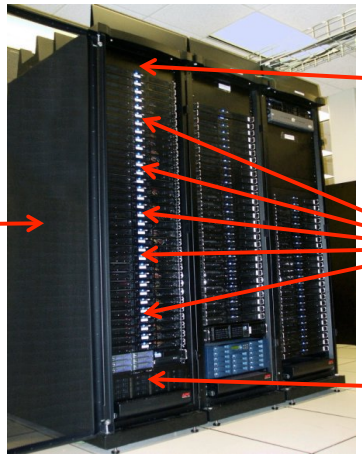
PC

Server

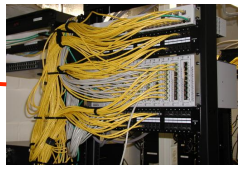
Cluster

Clusters

- Characteristics of a cluster:
  - Many similar machines, close interconnection (same room?)
  - Often special, standardized hardware (racks, blades)
  - Usually owned and used by a single organization



Rack



Network switch (connects nodes with each other and with other racks)

Many nodes/blades (often identical)

Storage device(s)

Power and cooling

- Clusters need lots of power
  - Example: 140 Watts per server
  - Rack with 32 servers: 4.5kW (needs special power supply!)
  - Most of this power is converted into heat

- Large clusters need massive cooling
  - 4.5kW is about 3 space heaters
  - And that's just one rack!

Scaling up

- What if your cluster is too big (hot, power hungry) to fit into your office building?
  - Build a separate building for the cluster
  - Building can have lots of cooling and power
  - Result: Data center

PC

Server

Cluster

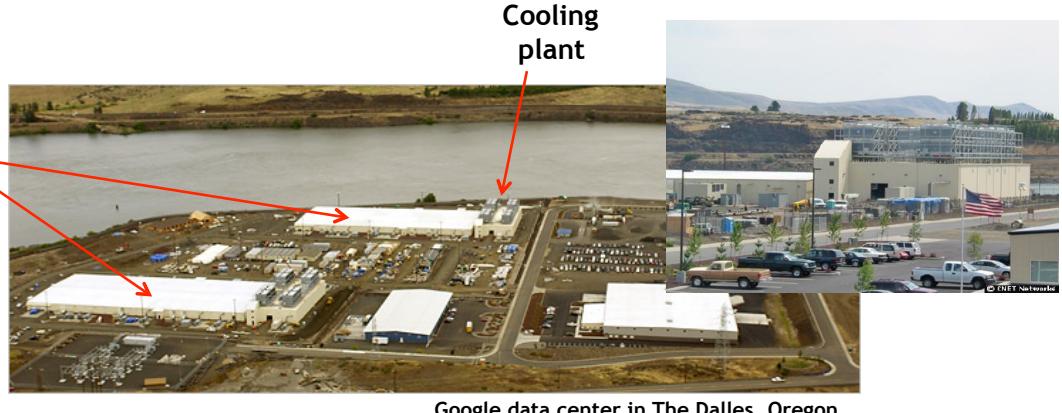
Data center

What does a data center look like?

- A warehouse-sized computer
  - A single data center can easily contain 10,000 racks with 100 cores in each rack (1,000,000 cores total)

Cooling plant

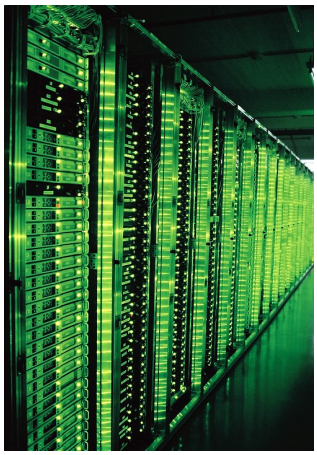
Data centers (size of a football field)



Google data center in The Dalles, Oregon

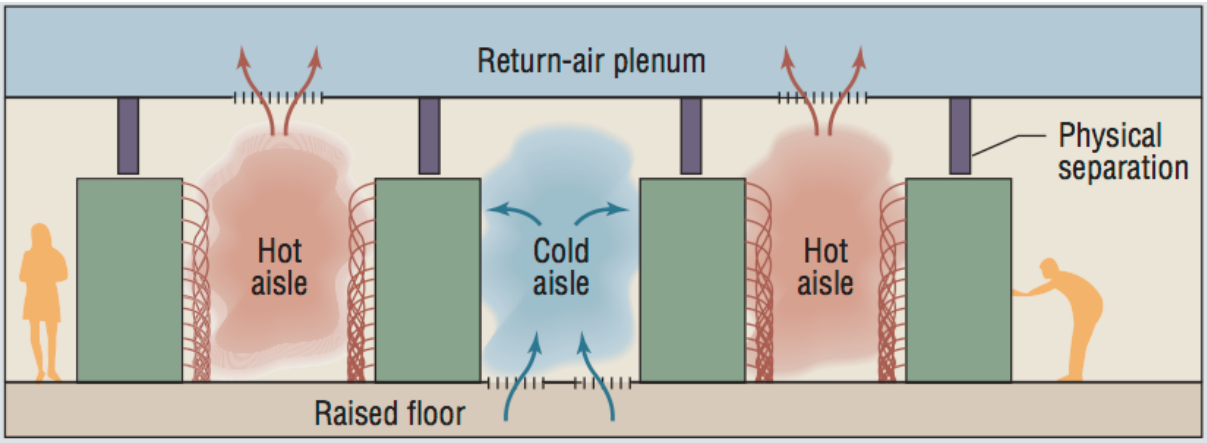
What's in a data center?

- Hundreds or thousands of racks
  - Each rack has 20-60 servers



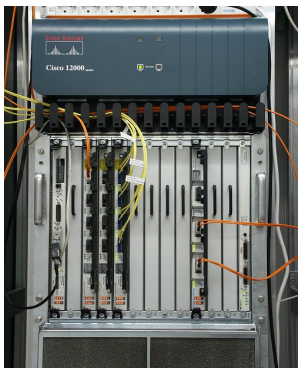
Source: 1&1

Rack organization



What's in a data center?

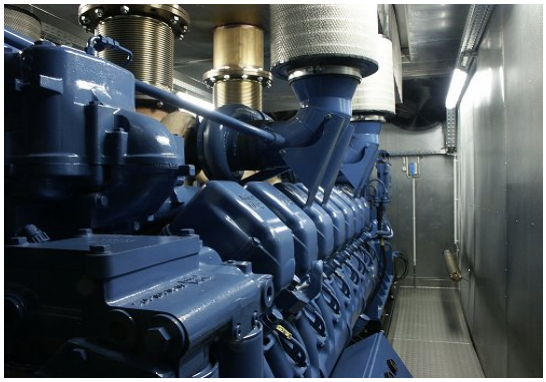
- Massive networking
  - Each rack has a Top-of-the-Rack (ToR) switch to interconnect the servers in the rack
  - How ToR switches are connected to one another?
    - We will shortly see some details



Source: 1&1

What's in a data center?

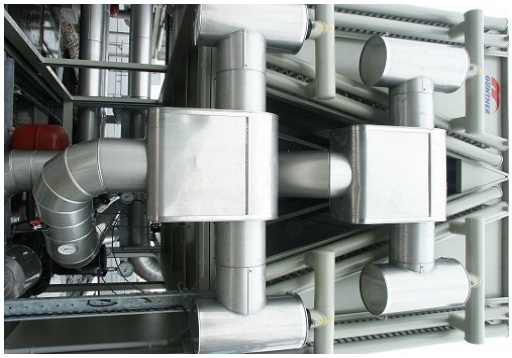
- Emergency power supplies



Source: 1&1

What's in a data center?

- Massive cooling



Source: 1&1

Energy matters!

- Data centers consume a lot of energy
  - Makes sense to build them near sources of cheap electricity
  - Example: Price per KWh is 3.6ct in Idaho (near hydroelectric power), 10ct in California (long distance transmission), 18ct in Hawaii (must ship fuel)
  - Most of this is converted into heat → Cooling is a big issue!

Company	Servers	Electricity	Cost
eBay	16K	~$0.6*10^5$ MWh	~$3.7M/yr
Akamai	40K	~$1.7*10^5$ MWh	~$10M/yr
Rackspace	50K	~$2*10^5$ MWh	~$12M/yr
Microsoft	>200K	>$6*10^5$ MWh	>$36M/yr
Google	>500K	>$6.3*10^5$ MWh	>$38M/yr
USA (2006)	**10.9M**	**$610*10^5$ MWh**	**$4.5B/yr**

Source: Qureshi et al., SIGCOMM 2009

Scaling up

- What if even a data center is not big enough?
  - Build additional data centers
  - Where? How many?

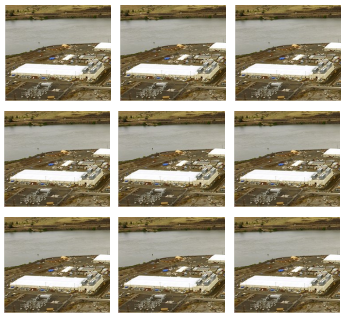
PC

Server

Cluster

Data center

Network of data centers



Global distribution

- Data centers are often globally distributed
  - Example above: Google data center locations (inferred)
- Why?
  - Need to be close to users (physics!)
  - Cheaper resources
  - Protection against failures

Trend: Modular data center

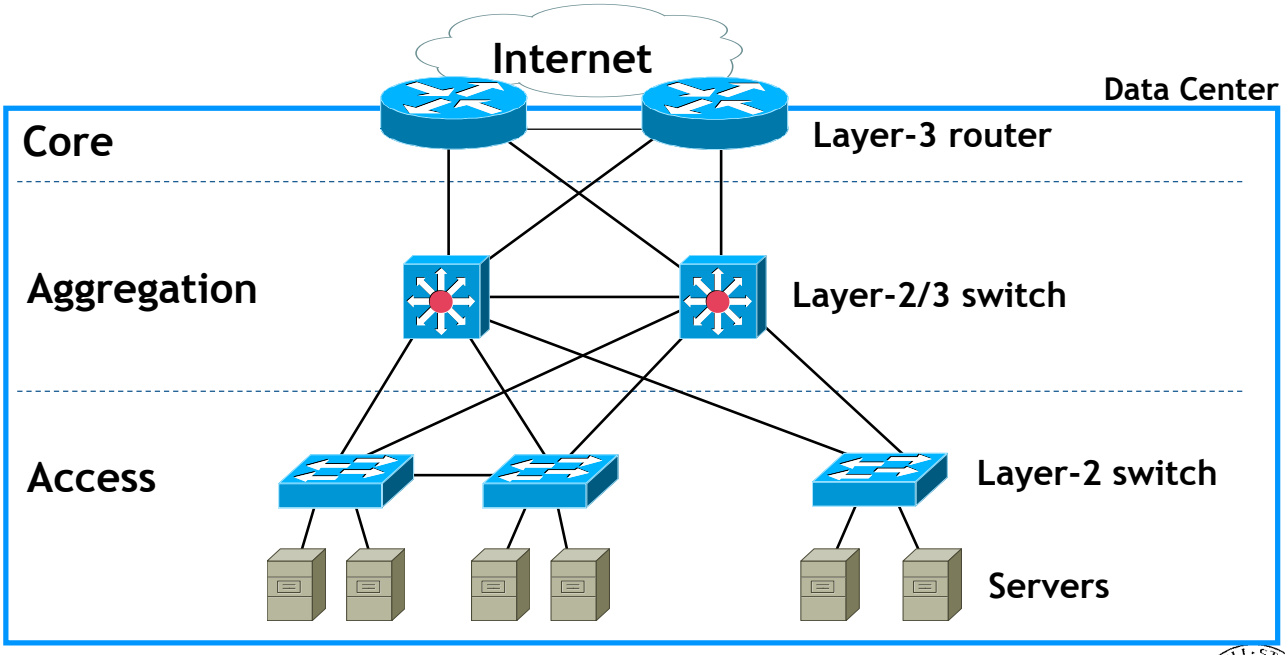
- Need more capacity? Just deploy another container!

Datacenter network topologies

Review of Layer 2 & Layer 3

- Layer 2
  - One spanning tree for entire network
    - Prevents looping
    - Ignores alternate paths
- Layer 3
  - Shortest path routing between source and destination
  - Best-effort delivery

Common data center topology



Pros and Cons of common DC topology

- Pros
  - Backwards compatible with existing infrastructure
    - No changes in application
    - Support of layer 2 (Ethernet)
  - Cost effective
    - Low power consumption & heat emission
    - Cheap infrastructure

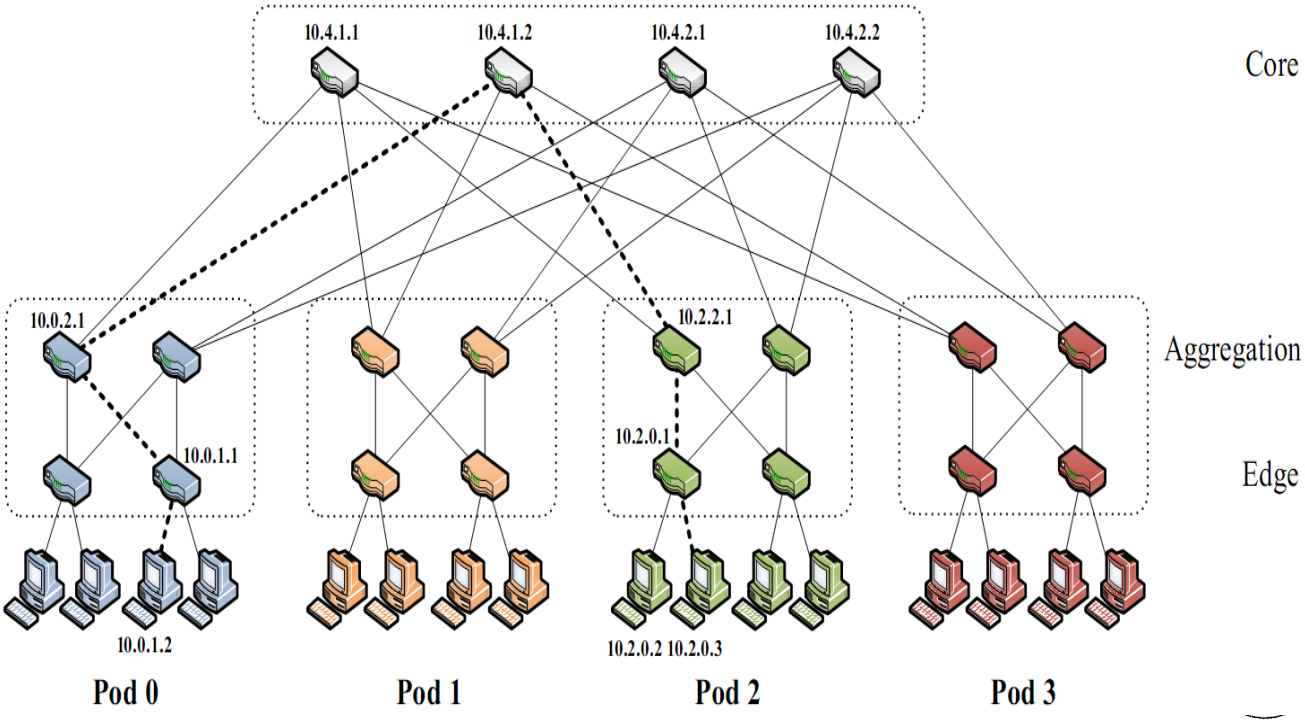
- Cons
  - Single point of failure
  - Over subscript of links higher up in the topology
    - Trade off between cost and provisioning

Fat-tree based solution

- Connect end-host together using a *fat tree* topology
  - Infrastructure consist of cheap devices
    - Each port supports same speed as endhost
  - All devices can transmit at line speed if packets are distributed along existing paths

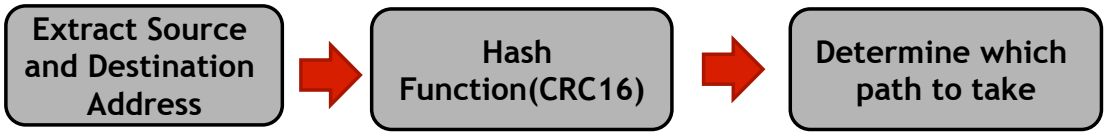
- A k-port fat tree can support $k^3/4$ hosts
- Between any two hosts there are (k/2)*(k/2) possible shortest paths

Fat-Tree Topology



ECMP (Equal-Cost Multiple-Path Routing)

- How to exploit the multiple available paths?
  - Load balancing based on source/destination address
- Pros
  - All the packets of a given flow follow the same path
- Cons
  - Not all flows are equal (mice, elephants)

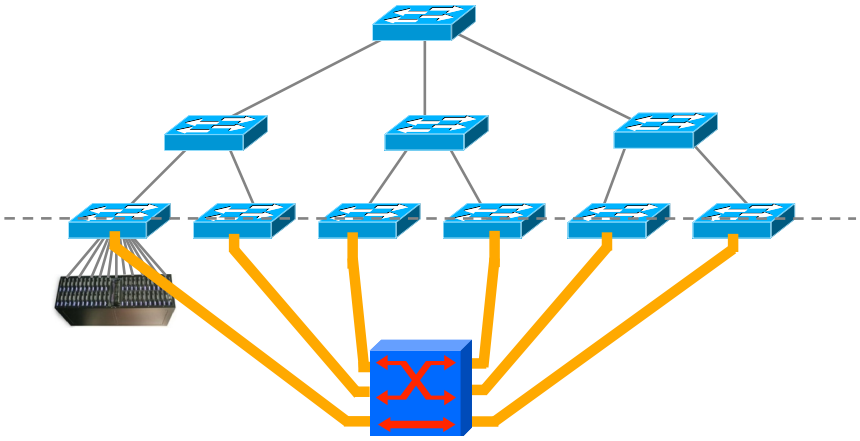


Problems with Fat-tree

- Layer 3 will only use one of the existing equal cost paths
  - And ECMP is not optimal
- No inherent support for VLAN traffic
  - Fat-tree topology ignores the traffic patterns that may arise from same user VMs
- Data center is fixed in size
  - Low flexibility (addition of new racks)
- Ignored connectivity to the internet
  - A path that may be used more often than the others

Alternative architectures: Hybrid packet/circuit switched network

- Traffic pattern is different from all-to-all
  - Most of the flows are intra-rack
  - Few ToRs exchange long flows
- Basic idea
  - Build and dedicate a path to long flows

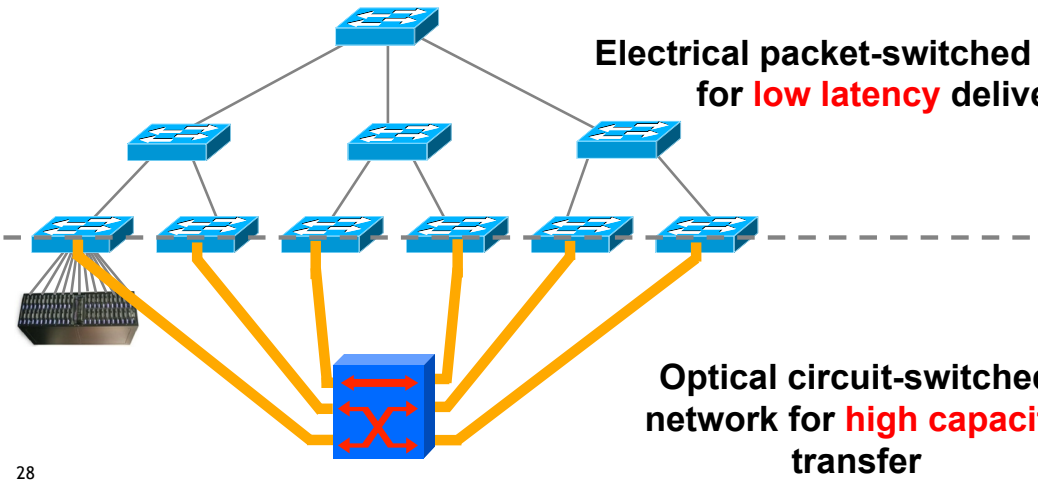


Optical circuit switching vs Electrical packet switching

	Electrical packet switching	Optical circuit switching
Switching technology	Store and forward	Circuit switching
Switching capacity	16x40 Gps at high end	320x100Gps on market
Switching time	Packet granularity	Less than 10 ms

Hybrid packet/circuit switched network architecture

- Full bisection bandwidth at packet granularity may not be necessary
- Optical paths are provisioned rack-to-rack
  - Aggregate traffic on per-rack basis to better utilize optical circuits

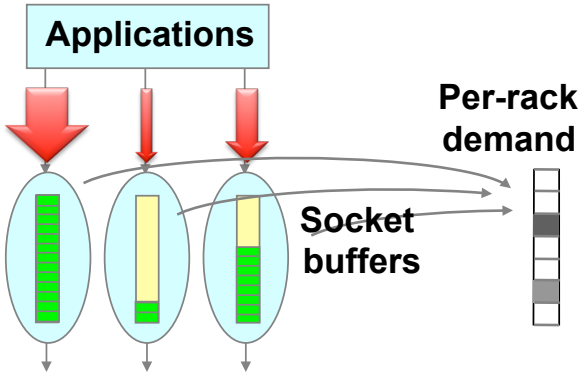


Design issues

- Control plane
  - Traffic demand estimation
  - Optical circuit configuration
  - → A centralized control is necessary
- Data plane
  - Dynamic traffic routing (multiplexing / de-multiplexing)

Traffic demand estimation and traffic batching

- Accomplish two requirements
  - Traffic demand estimation
  - Pre-batch data to improve optical circuit utilization



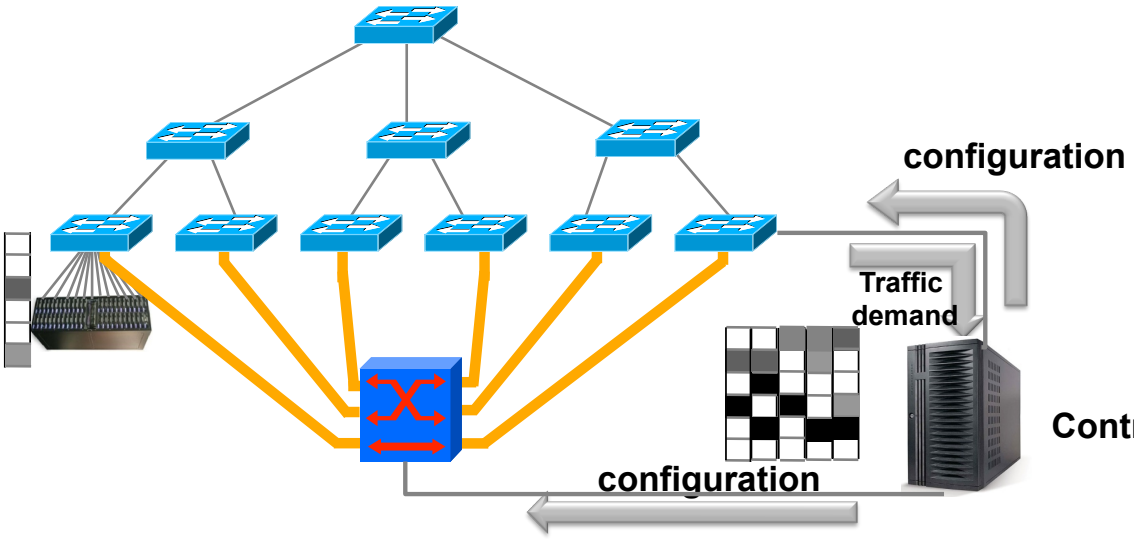
1. Transparent to applications

2. Packets are buffered per-flow to avoid HOL blocking

30

Optical circuit configuration

- Maximum weighted matching algorithm to compute optimal configuration
  - Slow algorithm!



Hybrid architectures: considerations

- Interesting approach
  - Exploit traffic patterns
  - The circuit switched network can be upgraded as technology evolves
- Focused on throughput, delay not considered
  - Recently, many hybrid architectures started to consider also the delay
- Main issue / opportunity
  - A centralized controller is necessary

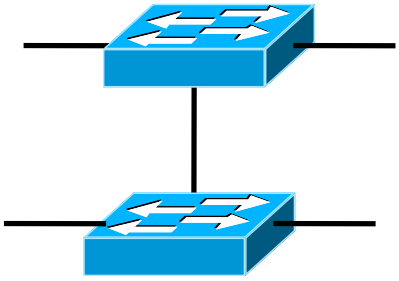
Software Defined Networks (SDN)

The Internet: A Remarkable Story

- Tremendous success
  - From research experiment to global infrastructure
- Brilliance of under-specifying
  - Network: best-effort packet delivery
  - Hosts: arbitrary applications
- Enables innovation in applications
  - Web, P2P, VoIP, social networks, virtual worlds
- But, change is easy only at the edge…

Inside the Network: A Different Story…

- Closed equipment
  - Software bundled with hardware
  - Vendor-specific interfaces
- Over specified
  - Slow protocol standardization
- Few people can innovate
  - Equipment vendors write the code
  - Long delays to introduce new features

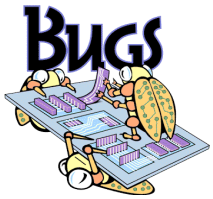


→ Impacts performance, security, reliability, cost…

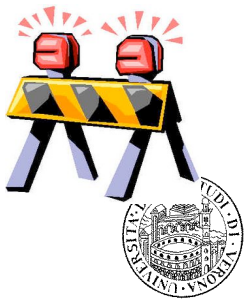
Networks are Hard to Manage

- Operating a network is expensive
  - More than half the cost of a network
  - Yet, operator error causes most outages

- Buggy software in the equipment
  - Routers with 20+ million lines of code
  - Cascading failures, vulnerabilities, etc.



- The network is “in the way”
  - Especially a problem in data centers
  - … and home networks



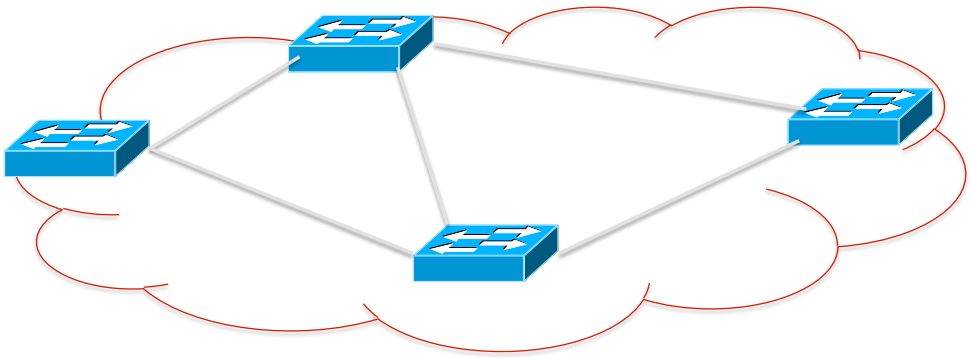
Creating Foundation for Networking

- A domain, not (yet?) a discipline
  - Alphabet soup of protocols
  - Header formats, bit twiddling
  - Preoccupation with artifacts
- From practice, to principles
  - Intellectual foundation for networking
  - Identify the key abstractions
  - … and support them efficiently
- To build networks worthy of society's trust

Traditional Computer Networks

The "Division of Labor"

Data plane:
Packet
streaming

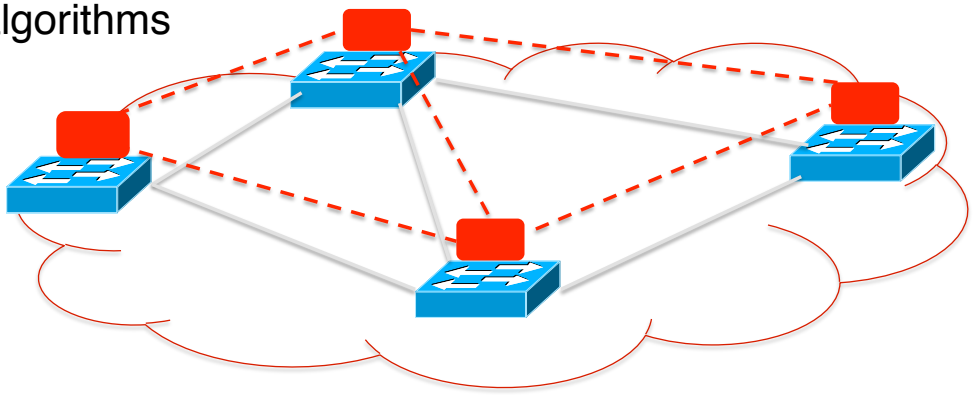


Forward, filter, buffer, mark,
rate-limit, and measure packets

Traditional Computer Networks

The “Division of Labor”

Control plane:
Distributed algorithms

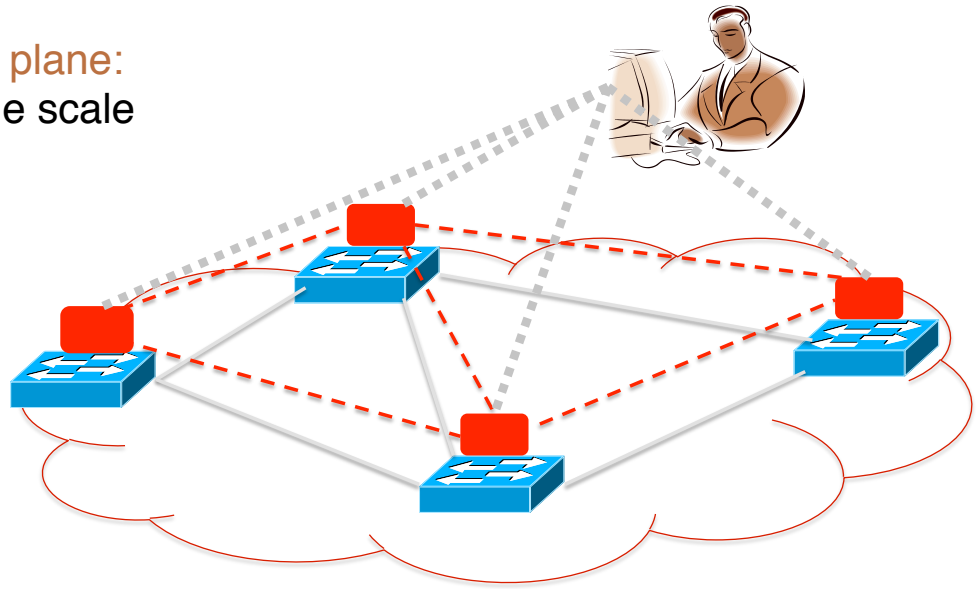


Track topology changes, compute routes, install forwarding rules

Traditional Computer Networks

The “Division of Labor”

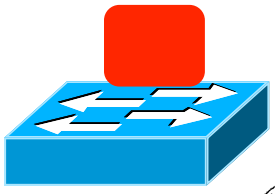
Management plane:
Human time scale



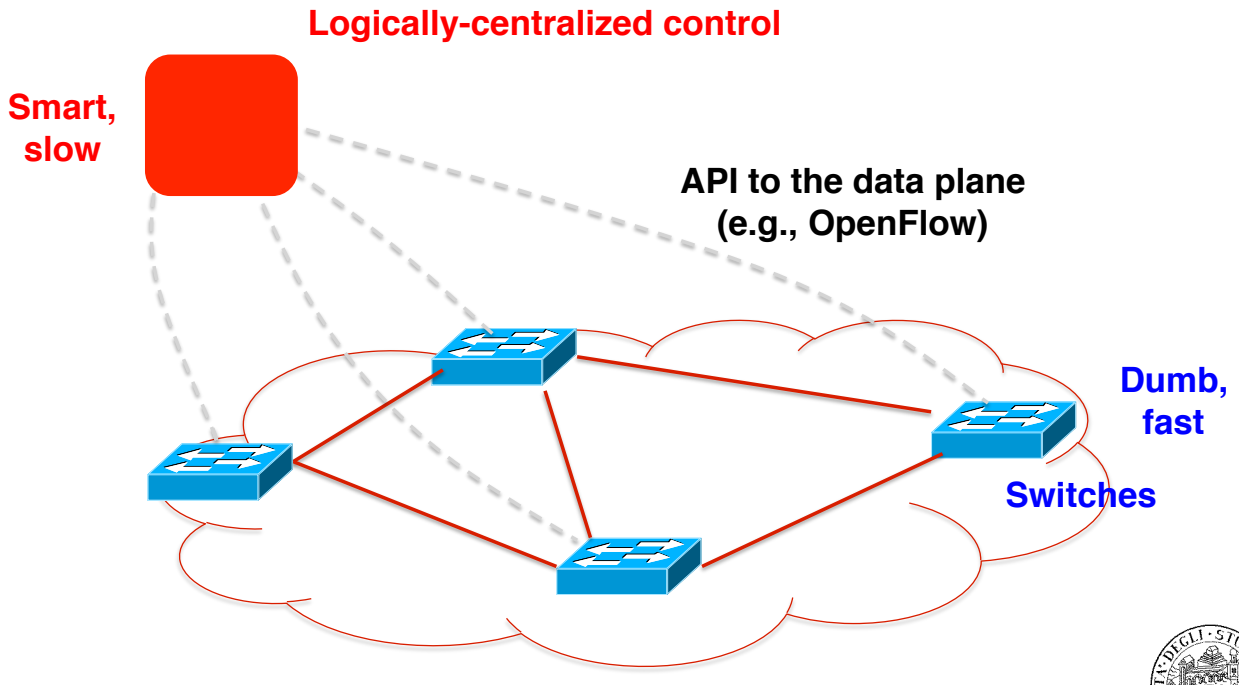
Collect measurements and configure the equipment

Death to the Control Plane!

- 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



Software Defined Networking (SDN)



Data-Plane: Simple Packet Handling

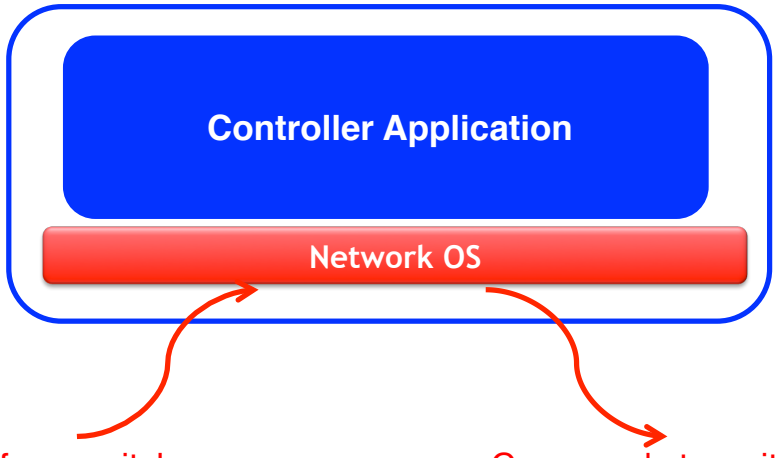
- Simple packet-handling rules
  - Pattern: match packet header bits
  - Actions: drop, forward, modify, send to controller
  - Priority: disambiguate overlapping patterns
  - Counters: #bytes and #packets

1. **src=1.2.*.*, dest=3.4.5.* → drop**
2. **src = *.*.*.*, dest=3.4.*.* → forward(2)**
3. **src=10.1.2.3, dest=*.*.*.* → send to controller**

Unifies Different Kinds of Boxes

- Router
  - Match: longest destination IP prefix
  - Action: forward out a link
- Switch
  - Match: destination MAC address
  - Action: forward or flood
- Firewall
  - Match: IP addresses and TCP/UDP port numbers
  - Action: permit or deny
- NAT
  - Match: IP address and port
  - Action: rewrite address and port

Controller: Programmability



Events from switches
Topology changes,
Traffic statistics,
Arriving packets

Commands to switches
(Un)install rules,
Query statistics,
Send packets

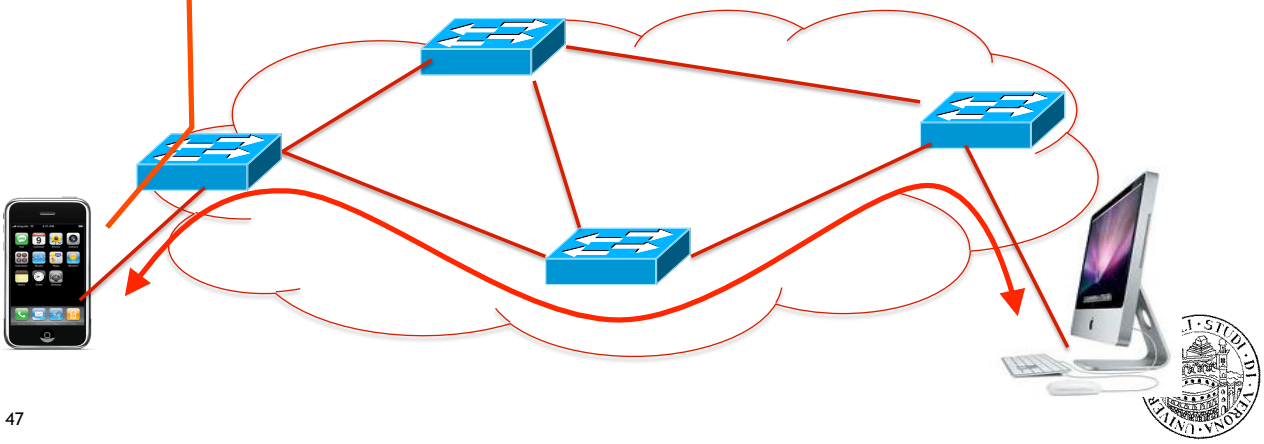
Example OpenFlow Applications

- **Dynamic access control**
- **Seamless mobility/migration**
- **Server load balancing**
- **Network virtualization**
- Using multiple wireless access points
- Energy-efficient networking
- Adaptive traffic monitoring
- Denial-of-Service attack detection

See http://www.openflow.org/videos/

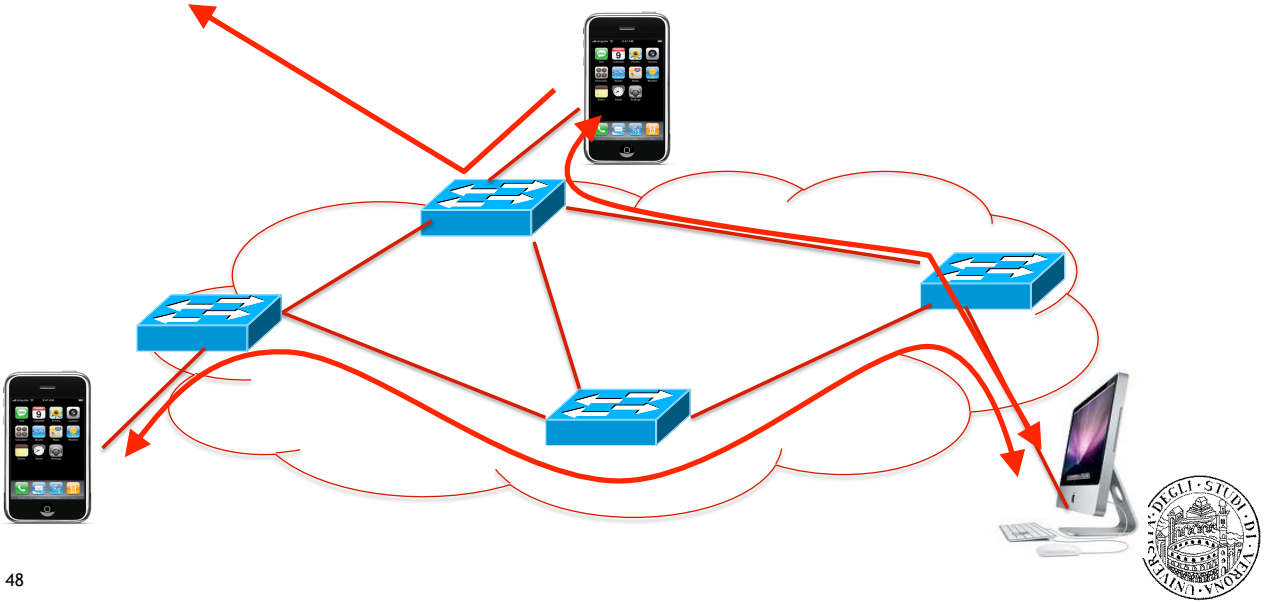
E.g.: Dynamic Access Control

- Inspect first packet of a connection
- Consult the access control policy
- Install rules to block or route traffic



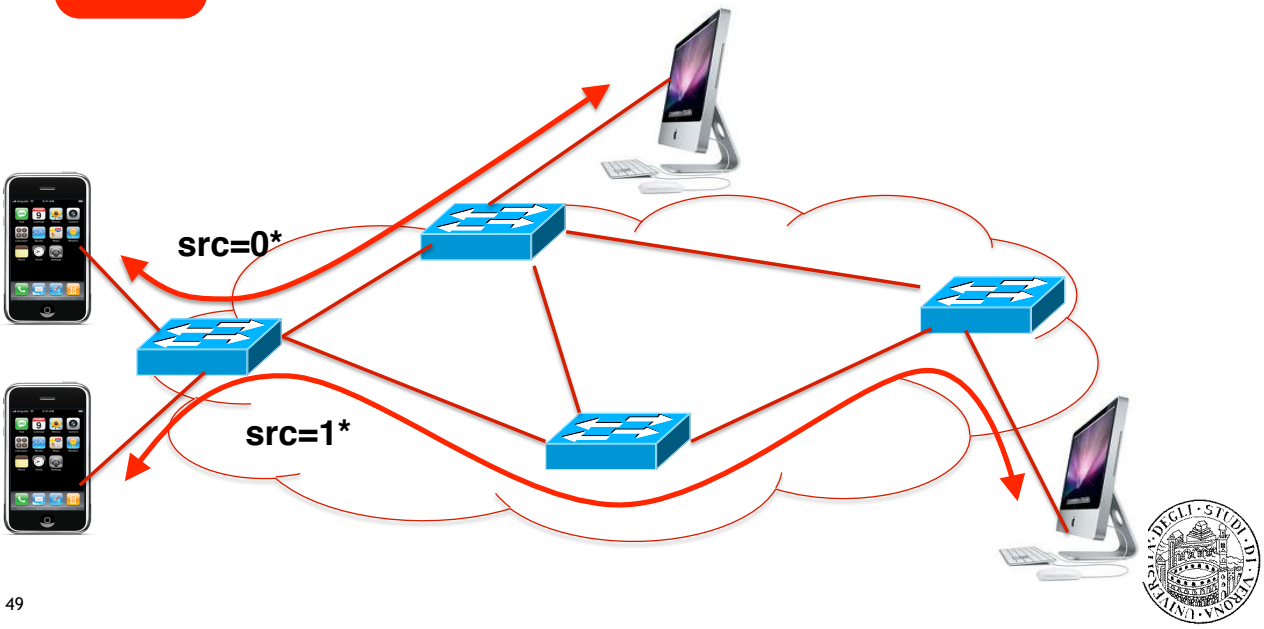
E.g.: Seamless Mobility/Migration

- See host send traffic at new location
- Modify rules to reroute the traffic

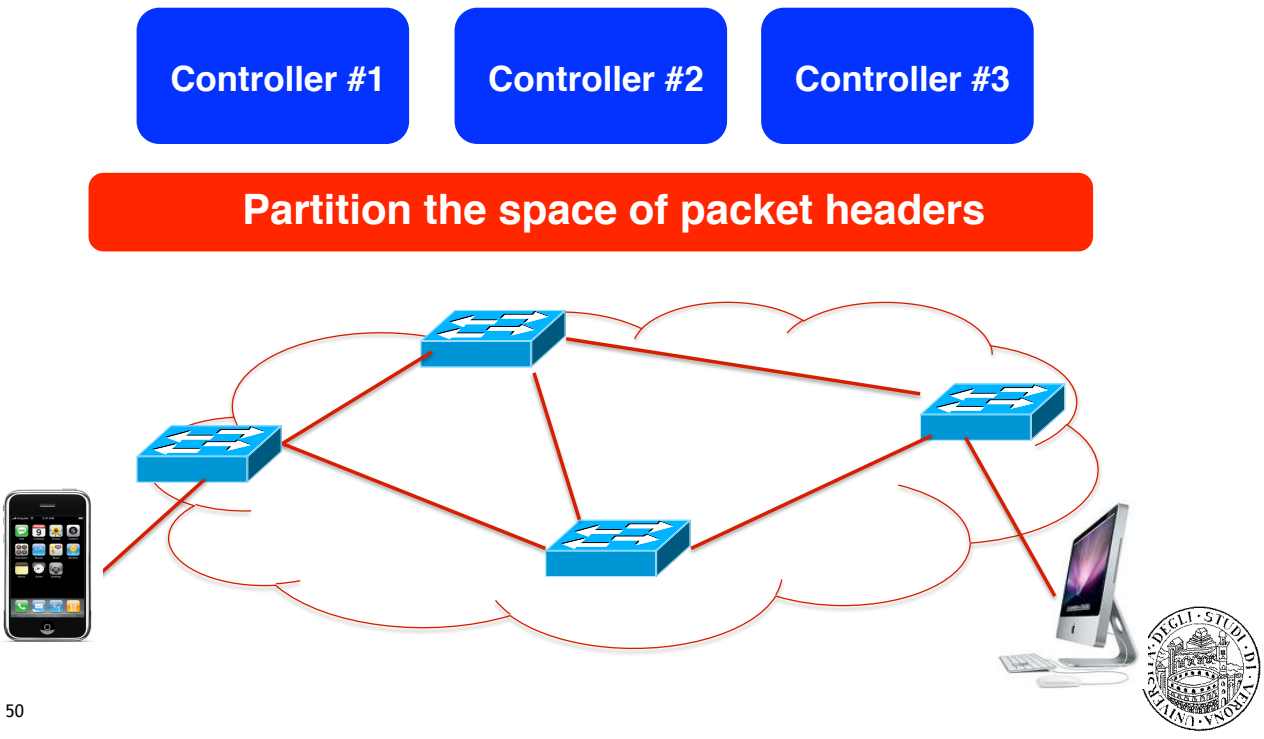


E.g.: Server Load Balancing

- Pre-install load-balancing policy
- Split traffic based on source IP



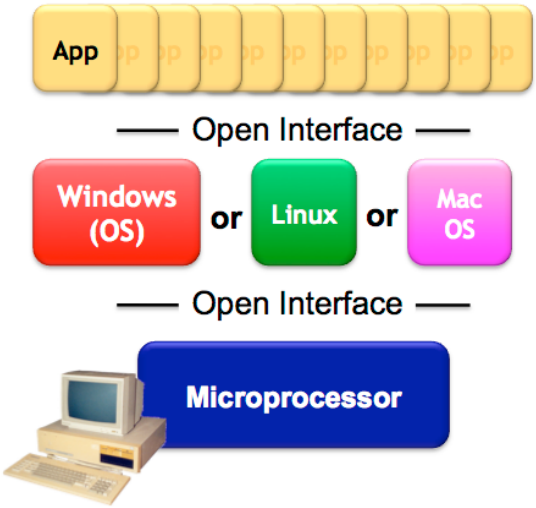
E.g.: Network Virtualization



OpenFlow in the Wild

- Open Networking Foundation
  - Google, Facebook, Microsoft, Yahoo, Verizon, Deutsche Telekom, and many other companies
- Commercial OpenFlow switches
  - HP, NEC, Quanta, Dell, IBM, Juniper, …
- Network operating systems
  - NOX, Beacon, Floodlight, Nettle, ONIX, POX, Frenetic
- Network deployments
  - Eight campuses, and two research backbone networks
  - Commercial deployments (e.g., Google backbone)

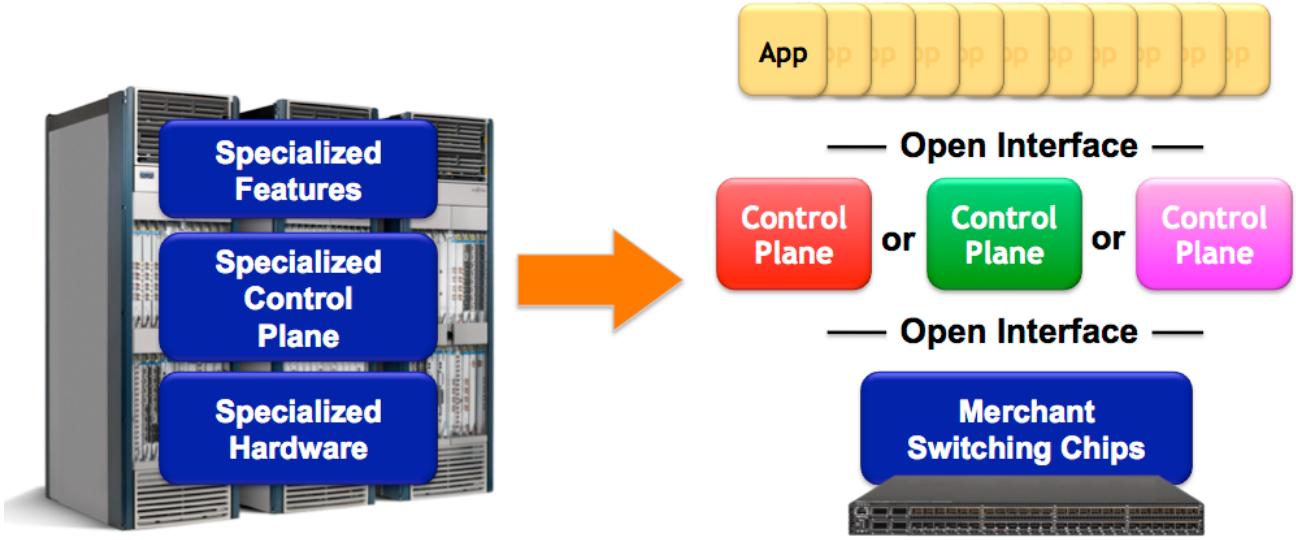
A Helpful Analogy: Mainframes



Vertically integrated
Closed, proprietary
Slow innovation
Small industry

Horizontal
Open interfaces
Rapid innovation
Huge industry

A Helpful Analogy (cont'd): Routers/Switches



Vertically integrated
Closed, proprietary
Slow innovation

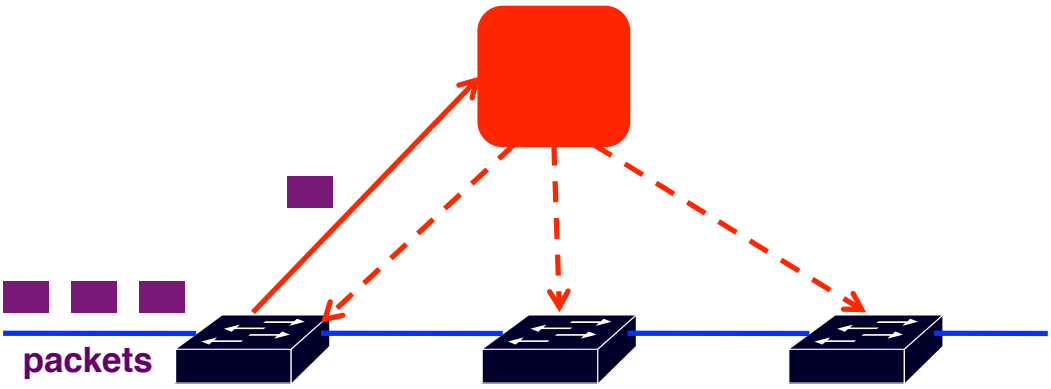
Horizontal
Open interfaces
Rapid innovation

Challenges: Heterogeneous Switches

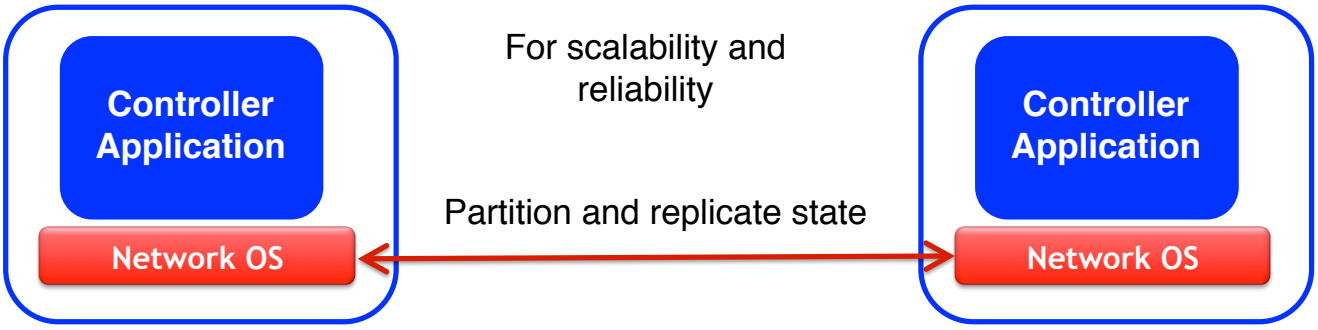
- Number of packet-handling rules
- Range of matches and actions
- Multi-stage pipeline of packet processing
- Offload some control-plane functionality (?)

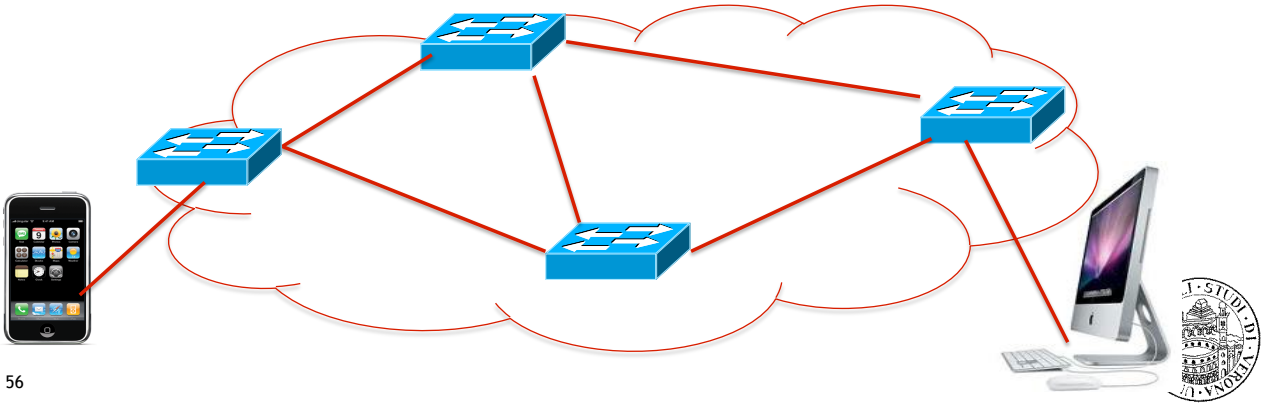
Challenges: Controller Delay and Overhead

- Controller is much slower the the switch
- Processing packets leads to delay and overhead
- Need to keep most packets in the "fast path"



Challenges: Distributed Controller





Challenges: Testing and Debugging

- OpenFlow makes programming possible
  - Network-wide view at controller
  - Direct control over data plane
- Plenty of room for bugs
  - Still a complex, distributed system
- Need for testing techniques
  - Controller applications
  - Controller and switches
  - Rules installed in the switches

SDN – Conclusion

- Rethinking networking
  - Open interfaces to the data plane
  - Separation of control and data
  - Leveraging techniques from distributed systems
- Significant momentum
  - In both research and industry

Cloud OS

The Datacenter is the new Computer

- Growing diversity of applications
  - Computing frameworks: MapReduce, Dryad, Pregel, Percolator, …
  - Storage systems: GFS, BigTable, Dynamo, …
- Growing diversity of users
  - Application users: Hive, Pig, …
  - System users: pools of Hadoop machines for ad-hoc MapReduce analysis
- This new computer needs an operating system
  - i.e., a common software layer that manages resources and provides shared services for the whole datacenter
  - Why? Same reasons computers needed one!

What Operating Systems Provide

- Resource sharing across applications & users
- Data sharing between programs
- Programming abstractions (e.g. threads, IPC)
- Debugging facilities (e.g. ptrace, gdb)

Result: OSes enable a highly interoperable software ecosystem that we now take for granted

Today's Datacenter OS

- Hadoop MapReduce as common execution and resource sharing platform
- Hadoop InputFormat API for data sharing
- Abstractions for productivity programmers, but not for system builders
- Very challenging to debug across all the layers
- What happens with the next hot platform after Hadoop?

Tomorrow's Datacenter OS

- Resource sharing:
  - Lower-level interfaces for fine-grained sharing
  - Optimization for a variety of metrics
    - e.g. energy, server utilization
  - Integration with network scheduling mechanisms

- Data sharing:
  - Standard interfaces for cluster file systems, key-value stores, etc
  - In-memory data sharing and a unified system to manage this memory
  - Streaming data abstractions (analogous to pipes)

Tomorrow's Datacenter OS (cont'd)

- Programming abstractions:
  - Tools that can be used to build the next MapReduce / BigTable in a week
  - Efficient implementations of communication primitives (e.g. shuffle, broadcast)
  - New distributed programming models

- Debugging facilities:
  - Tracing and debugging tools that work across the cluster software stack
  - Replay debugging that takes advantage of limited languages / computational models
  - Unified monitoring infrastructure and APIs

Other aspects

- Datacenter OS advantages
  - Server consolidation
    - reduce hardware and power requirements
  - On-the-fly resizing of the physical infrastructure
  - Service workload balance among physical resources
    - improve efficiency and utilization
  - Server replication
    - support fault tolerance and high availability capabilities
  - Dynamic partitioning of physical infrastructure
    - provide isolation

Ongoing projects

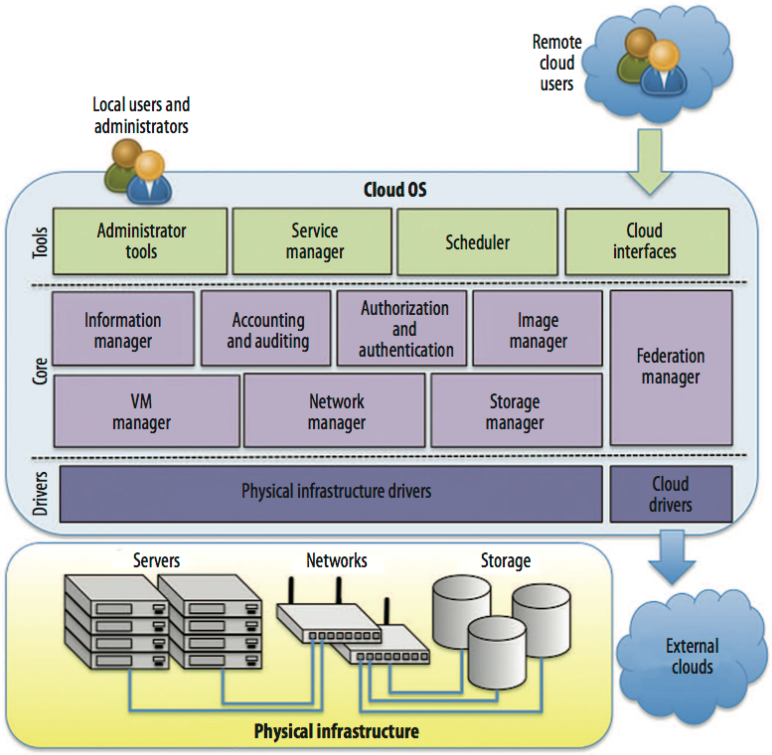
- Recently new projects started to address the Cloud OS issues
  - Prominent (open source) examples:
    - OpenNebula
    - CloudStack
    - Eucalyptus
    - OpenStack
- There are some differences among these projects, since they may target some different problems
  - e.g., interoperability with existing solutions, such as Amazon EC2
- Nevertheless, there is a common basic structure that all of them has
  - In the next slides we will provide a high level overview of the components of these systems

Cloud OS: Introduction

- Virtualization as a key enabling technology for datacenters
  - Abstraction of compute, network, and storage services

- The *virtual infrastructure manager* is an essential component of the datacenter architecture
  - Also known as the cloud operating system (cloud OS)
  - Two basic functions
    - It orchestrates the deployment of virtual resources
    - It manages the physical and virtual infrastructures
  - It may include the management of other federated datacenters or commercial clouds

The Cloud OS



Infrastructure and Cloud Drivers

- The cloud OS has different core components, such as:
  - Virtual machine manager
  - Network manager
  - Storage manager
  - Information manager

- The cloud OS can use adapters or *drivers* to interact with the virtualization technologies
  - Hypervisor, network, storage, and information drivers
  - It can include different cloud drivers to enable access to remote providers.

Virtual Machine Manager

- VM as the basic execution unit
  - A VM consists of a set of parameters and attributes, including the OS kernel, VM image, memory and CPU capacity, network interfaces etc.
  - The user can add VMs as needed (horizontal scaling)
  - Or resize a VM (vertical scaling)
    - if supported by the underlying hypervisor technology

- Isolation is provided at the user level
  - VMs in the same applications share a communication network and services when needed.

Virtual Machine Manager (cont'd)

- The VM manager is responsible for managing a VM's entire life cycle
  - deploy, migrate, suspend, resume, shut down
- The VM manager is also responsible for preserving the service-level agreements contracted with the users
  - e.g., VM availability
  - The VM manager includes VM crashes detection and automatic VM restarting

Network Manager

- Service deployment involves the instantiation of communication networks to interconnect the different service components
  - It also makes the service reachable for external users
- The network manager is responsible for
  - providing private networks
    - internal connectivity
  - managing public IP address pools
    - to connect the front-end service components to the Internet.
- Network drivers are used to provision virtual networks over the physical network infrastructure
  - Traffic isolation between virtual networks.

Storage Manager

- It provides storage services and final-user virtual storage systems
- The storage system must be
  - Scalable
    - It can grow dynamically according to service needs
  - Highly available and reliable
    - It avoids data access disruption in data access in case of failure
  - High-performance
    - It supports strong demands of data-intensive workloads
  - Easy to manage
    - It abstracts users from the underlying physical storage's complexity
- Storage drivers introduce a layer of abstraction between users or services and physical storage
  - Enable the creation of a storage resource pool where storage devices appear as one

Image Manager

- It handles the VM images belonging to different users
  - Different operating systems and software configurations
- A set of attributes defines the VM image
  - Image's name
  - Description of its contents
  - Type of image —public, private, or shared—
  - Image owner
  - Image's location within the repository.
- Basic image functionality includes tools for different operations
  - Create, deletion, clone, add or change an image attribute, share an image with other users, publish an image for public use

Information Manager

- Responsible for monitoring and gathering information about
  - the state of VMs
  - the state of physical servers,
  - the state of other components of virtual and physical infrastructures
    - network devices
    - storage systems
- At the physical server level, some tools are
  - Nagios (www.nagios.com)
  - Ganglia (http://ganglia.sourceforge.net).
- Monitoring at the VM level relies on the information provided by hypervisors

Authentication and Authorization

- Clouds include mechanisms
  - to authenticate users and administrators
  - to provide them with access only to authorized resources
    - VMs, networks, storage systems
- These mechanisms use policies to control and manage user privileges
  - Access control implemented using role-based mechanisms
  - Quota mechanisms used to limit the amount of resources a specific user can access
    - CPU, memory, network bandwidth, or disk

Accounting and Auditing

- Accounting
  - It obtains and records resource usage information of deployed services
  - It relies on the information manager to monitor resources and collect usage information from metric measurements
  - Essential for producing billing information

- Auditing
  - It provides information about activity in cloud resources
    - who accessed cloud resources
    - when they gained access
    - what operations they performed
  - Useful to improve cloud security

Scheduler

- Two levels of scheduling
  - At the cloud level, managed by the cloud OS scheduler
    - to decide the particular physical server where each VM is deployed
  - At the physical host level, managed by the hypervisor scheduler
    - to decide when VMs can obtain system resources and which physical CPUs are assigned to each VM

- The cloud OS scheduler's main function is to decide the initial placement of each VM following specific criteria
  - Based on user specified constraints
    - CPU, memory, type of hypervisor, OS, location, SLA
  - It provides dynamic optimization capabilities
    - Dynamic reallocation (migration) of VMs

Optimization Criteria for Allocation and Reallocation Policies of the Scheduler

Optimization criteria	Target	Allocation policy
Server consolidation	Reduce the number of servers in use to minimize energy consumption.	VMs should be placed using a minimum number of servers.
Workload balance	Balance the workload of all servers to avoid server saturation and performance slowdown.	VMs should be evenly distributed among the available servers.
CPU balance	Balance the use of CPUs to avoid server saturation and performance slowdown.	A new VM should be located in the server with the highest amount of available CPUs.
Thermal balance	Balance the temperature of all servers to avoid overheating and reduce cooling requirements.	A new VM should be located in the server exhibiting the lowest temperature.

Administrative Tools

- Interfaces for users and administrators to perform various tasks
- Privileged administration
  - Administration tools
    - Create, modify, or delete users and manage user authorization and access control policies
  - Physical infrastructure management tools
    - Boot or shut down physical servers, monitor physical infrastructure, …
- Unprivileged users
  - Manage their own infrastructure
    - Deploy, shut down, suspend, restore, or monitor a VM)
    - Create or delete virtual networks
    - Create, delete, or attach a virtual disk
    - Create, clone, or delete images

Cloud Interfaces

- Most cloud products and providers offer their own APIs
  - Amazon EC2
  - VMware's vCloud
- Some of these APIs are becoming de facto standards
  - But this heterogeneity makes it difficult to achieve interoperability and portability across clouds
- Several standards bodies are addressing interoperability and portability issues
  - Examples:
    - OGF OCCI (http://occi-wg.org)
    - OVF (http://dmtf.org/standards/cloud)

Example of a Cloud OS

OpenStack: The Open Source Cloud Operating System