Cloud computing security

INGI2347: COMPUTER SYSTEM SECURITY (Spring 2014)

Marco Canini

UCL Université catholique de Louvain

Lecture slides adapted from Upenn NETS212 by Andreas Haeberlen
Reproduced with permission

Plan for today

Lecture 12

- Cloud computing

  - The need for scaling from PCs to data centers
  - What is cloud computing?
  - Virtualization: How clouds work "under the hood"
  - Security challenges of cloud computing
- Software-defined networking (SDN)
  - The need for SDN
  - OpenFlow

How many users and objects?

- Flickr has >6 billion photos
- Facebook has 1.15 billion active users
- Google is serving >1.2 billion queries/day on more than 27 billion items
- >2 billion videos/day watched on YouTube

How much data?

- Modern applications use massive data:
  - Rendering 'Avatar' movie required >1 petabyte of storage
  - eBay has >6.5 petabytes of user data
  - CERN's LHC will produce about 15 petabytes of data per year
  - In 2008, Google processed 20 petabytes per day
  - German Climate computing center dimensioned for 60 petabytes of climate data
  - Google now designing for 1 exabyte of storage
  - NSA Utah Data Center is said to have 5 zettabyte (!)
- How much is a zettabyte?
  - 1,000,000,000,000,000,000,000 bytes
  - A stack of 1TB hard disks that is 25,400 km high

How much computation?

- No single computer can process that much data
  - Need many computers!
- How many computers do modern services need?
  - Facebook is thought to have more than 60,000 servers
  - Akamai has 95,000 servers in 71 countries
  - Intel has ~100,000 servers in 97 data centers
  - Microsoft reportedly had at least 200,000 servers in 2008
  - Google is thought to have more than 1 million servers, is planning for 10 million (according to Jeff Dean)

Scaling up

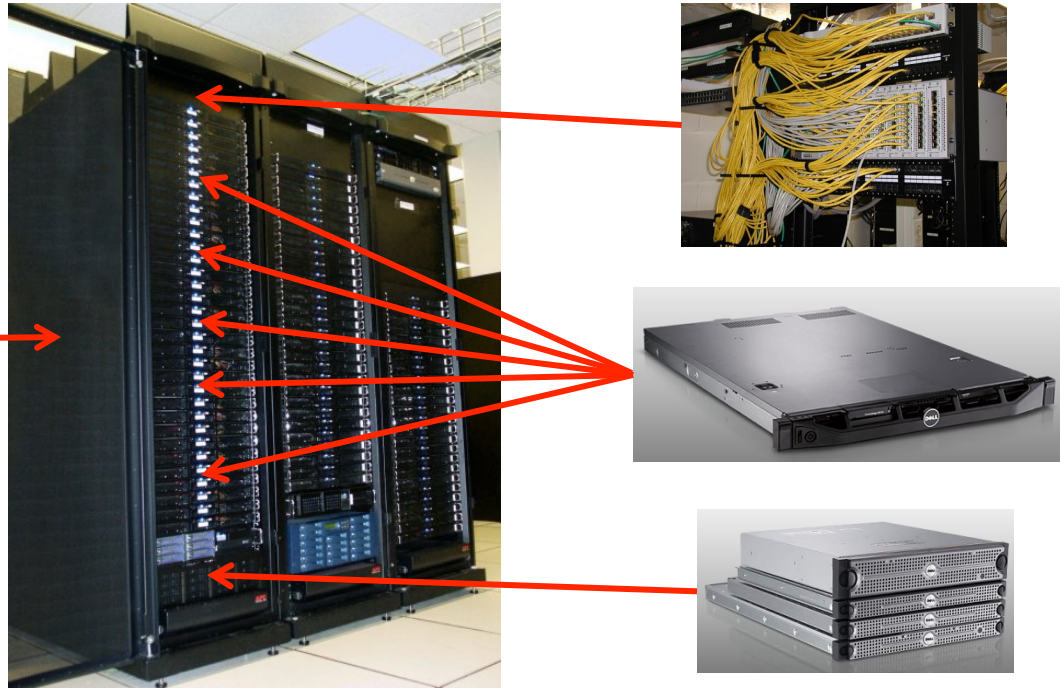
PC

Server

Cluster

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

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

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

PC

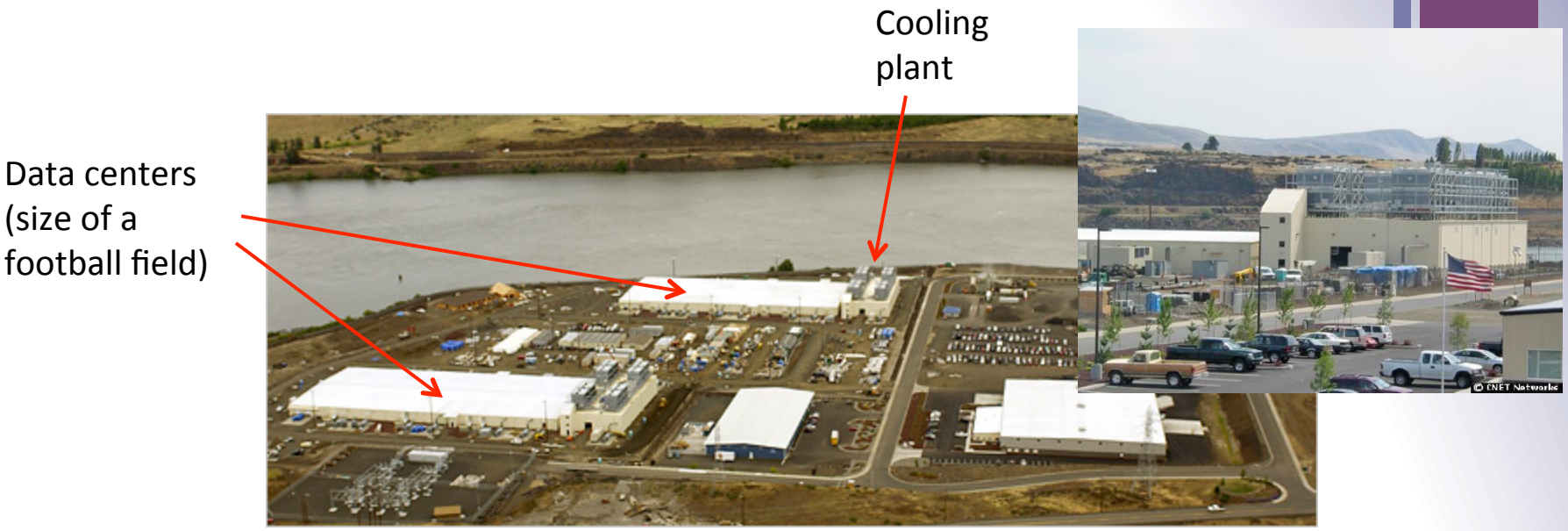
Server

Cluster

Data center

- 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

What does a data center look like?



Google data center in The Dalles, Oregon

- 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)

What's in a data center?

Source: 1&1

- Hundreds or thousands of racks

What's in a data center?

Source: 1&1

- Massive networking

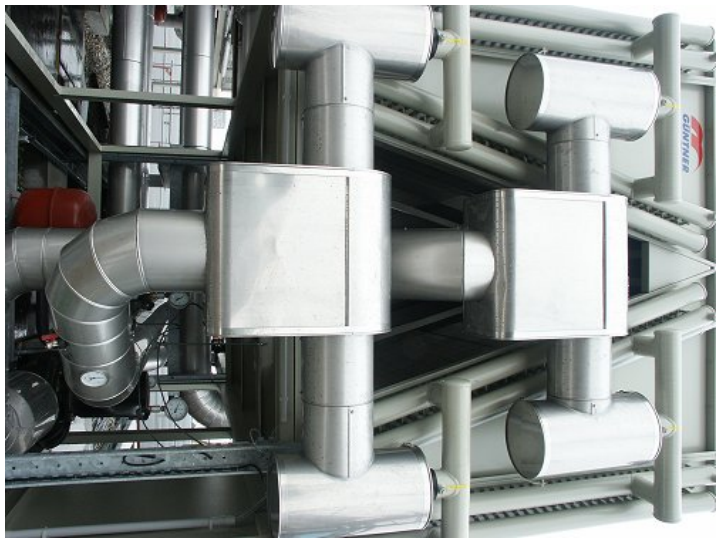
What's in a data center?



Source: 1&1

- Emergency power supplies

What's in a data center?



Source: 1&1

- Massive cooling

Energy matters!

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

- 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!

Scaling up

PC

Server

Cluster

Data center

Network of data centers

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

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!

What is cloud computing?

What is cloud computing?

- According to NIST:

  Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

- Essential characteristics:
  - On-demand self service
  - Broad network access
  - Resource pooling
  - Rapid elasticity
  - Measured service

Why is this a good thing?

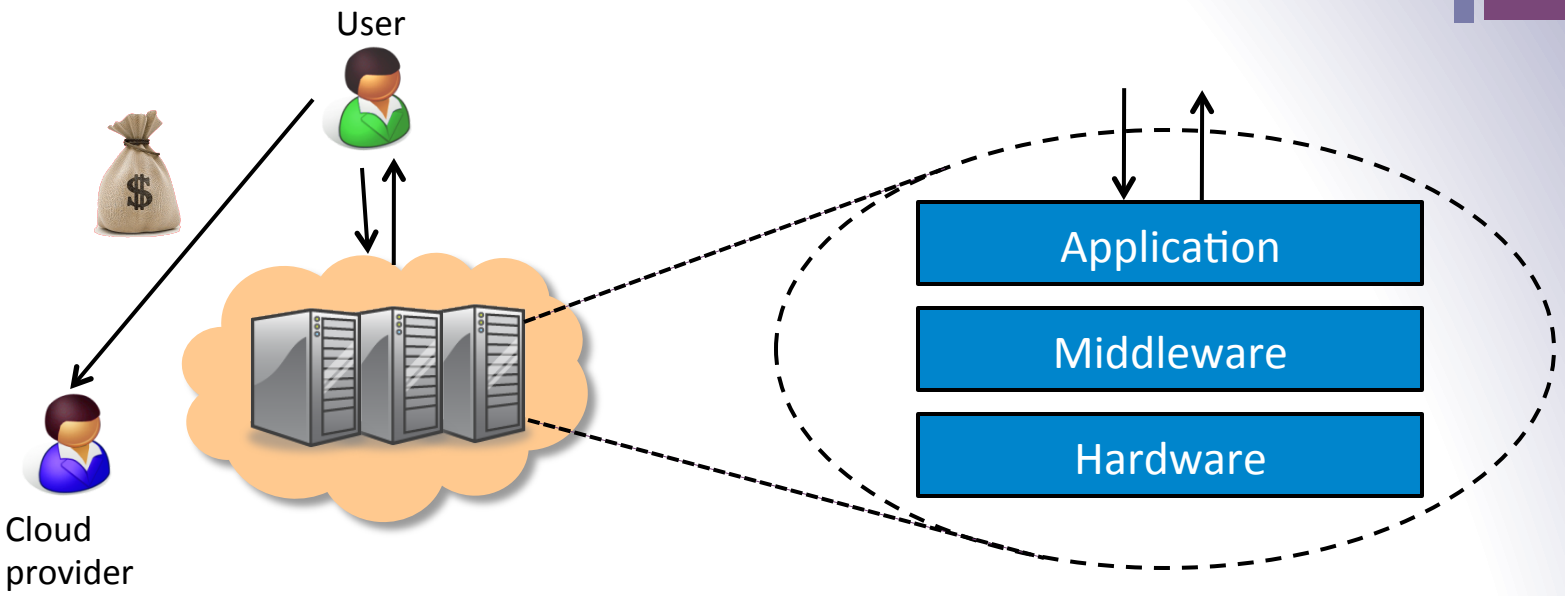
- Economies of scale
  - Cheaper to run one big data center than many small ones
- Statistical multiplexing
  - High utilization!
- No up-front commitment
  - No investment in data center; pay-as-you-go model
- Scalability
  - Thousands of computers available on demand; add more within seconds

What kinds of clouds exist today?

Three types commonly distinguished:

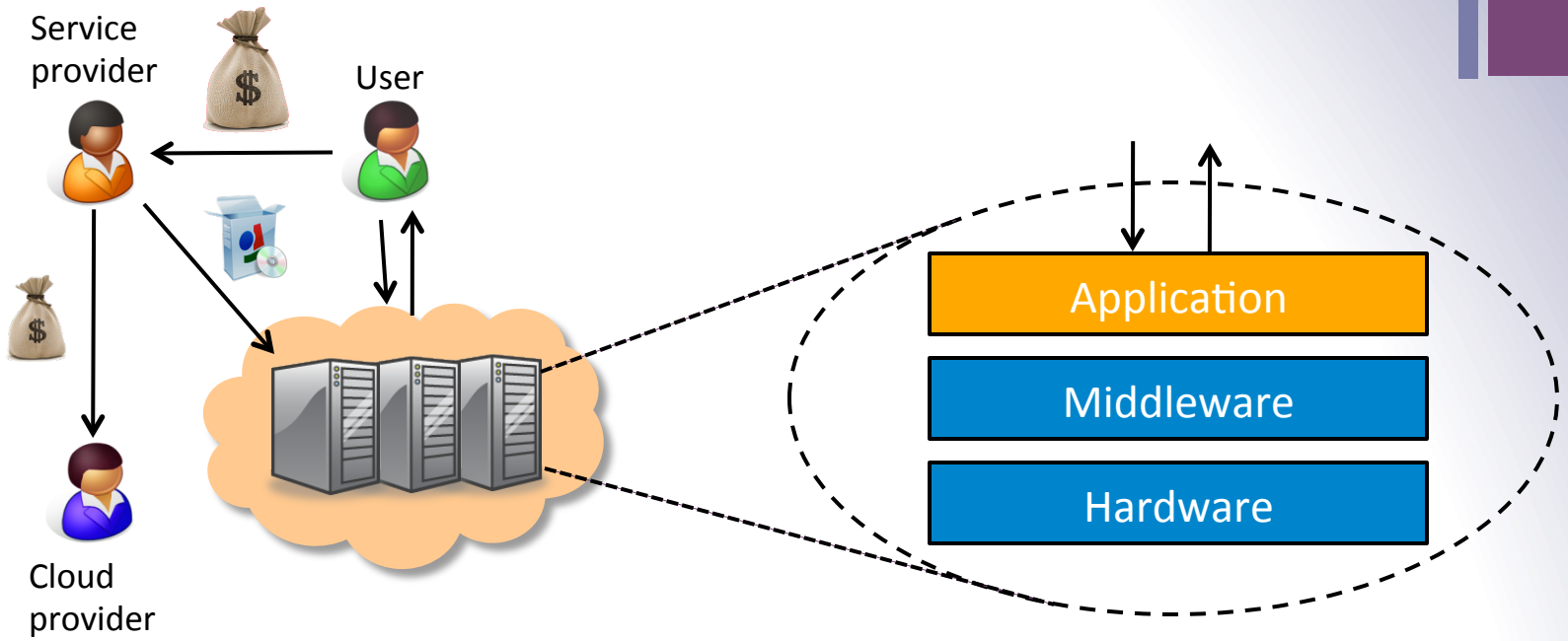
- Software as a service (SaaS)
- Platform as a service (PaaS)
- Infrastructure as a service (IaaS)
- Other xaaS types exist, but are less common
  - Desktop, Backend, Communication, Network, Monitoring, ...

Software as a Service (SaaS)



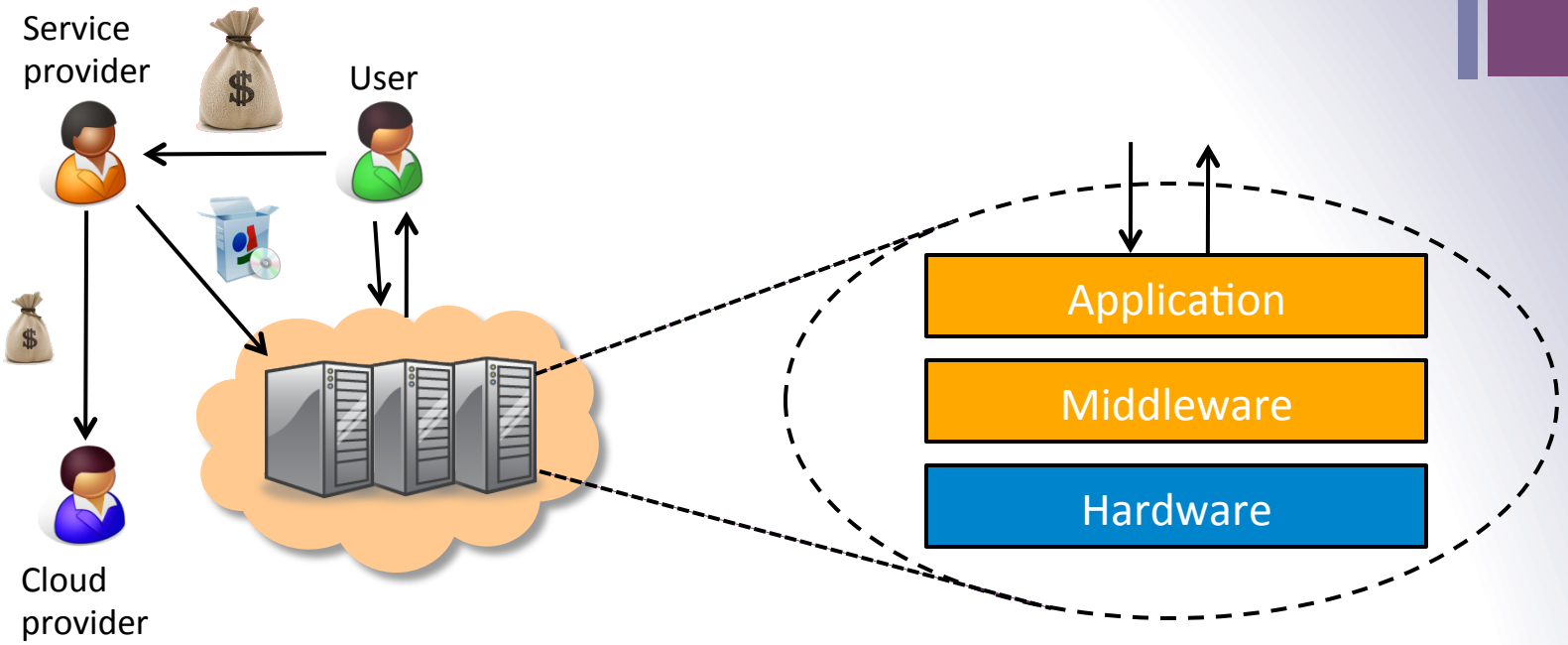
- Cloud provides an entire application
  - Word processor, spreadsheet, CRM software, calendar...
  - Customer pays cloud provider
  - Example: Google Apps, Salesforce.com

Platform as a Service (PaaS)



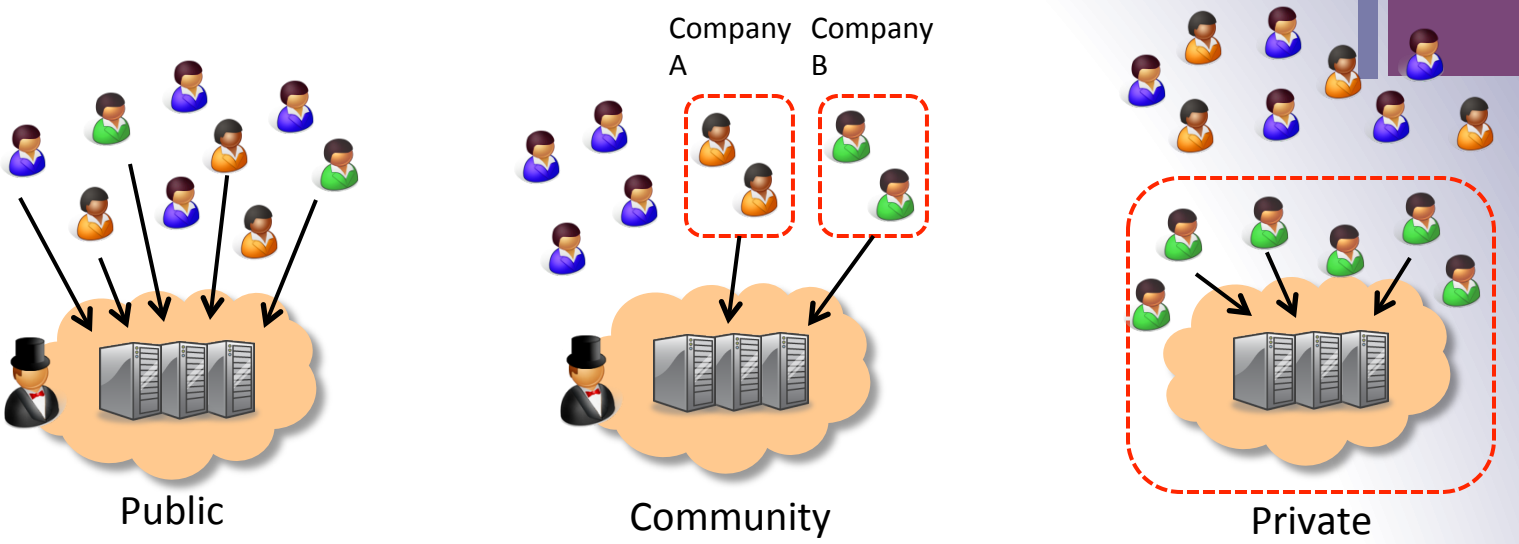
- Cloud provides middleware/infrastructure
  - Customer pays service provider for the service; service provider pays the cloud for the infrastructure
  - Example: Windows Azure, Google App Engine

Infrastructure as a Service (IaaS)



- Cloud provides raw computing resources
  - Virtual machine, blade server, hard disk, ...
  - Customer pays service provider for the service; service provider pays the cloud for the resources
  - Examples: Amazon Web Services, Rackspace Cloud, GoGrid

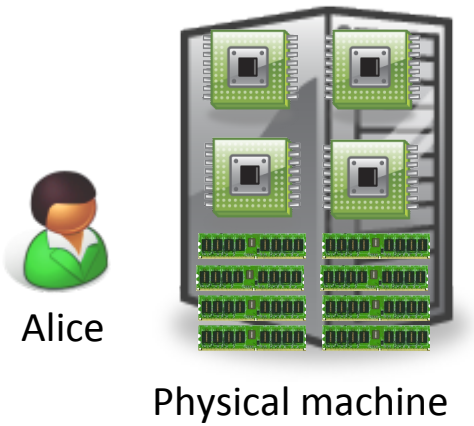
Private/hybrid/community clouds

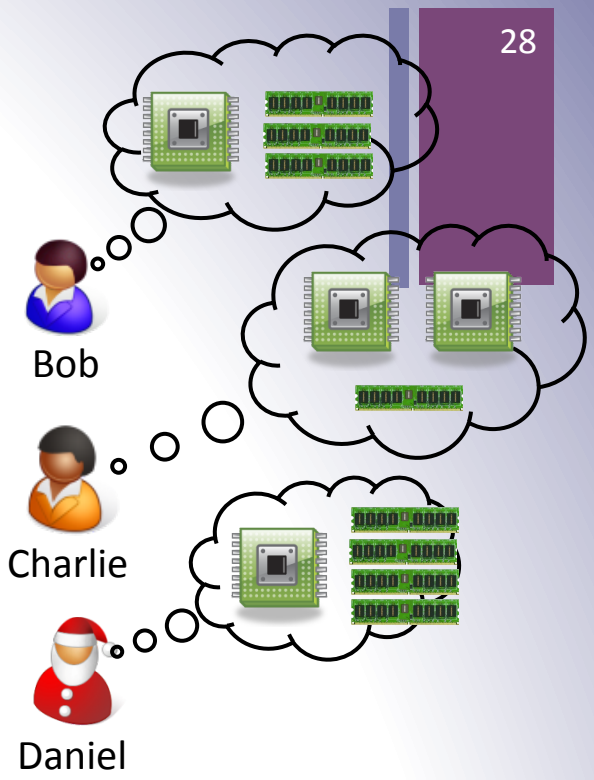


- Who can become a customer of the cloud?
  - Public cloud: Commercial service; open to (almost) anyone. Example: Amazon AWS, Microsoft Azure, Google App Engine
  - Community cloud: Shared by several similar organizations. Example: Google's "Gov Cloud"
  - Private cloud: Shared within a single organization. Example: Internal datacenter of a large company.

Virtualization: How clouds work "under the hood"

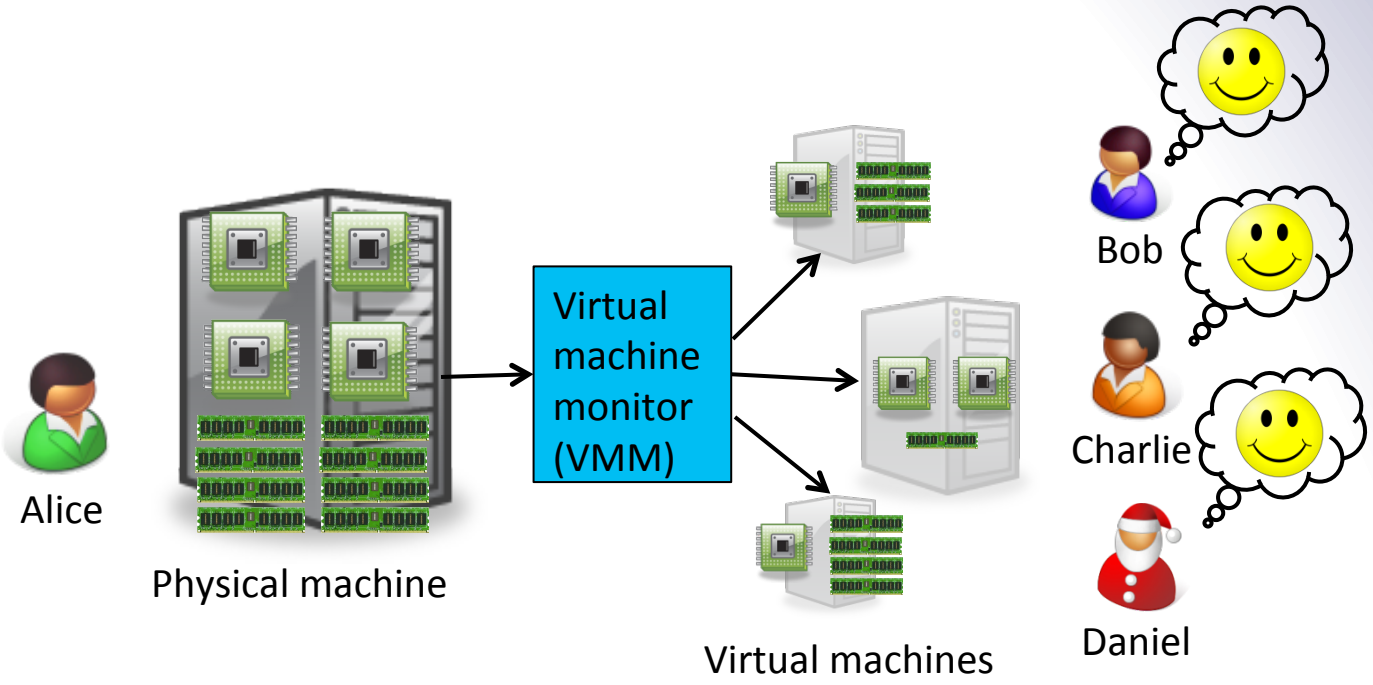
What is virtualization?





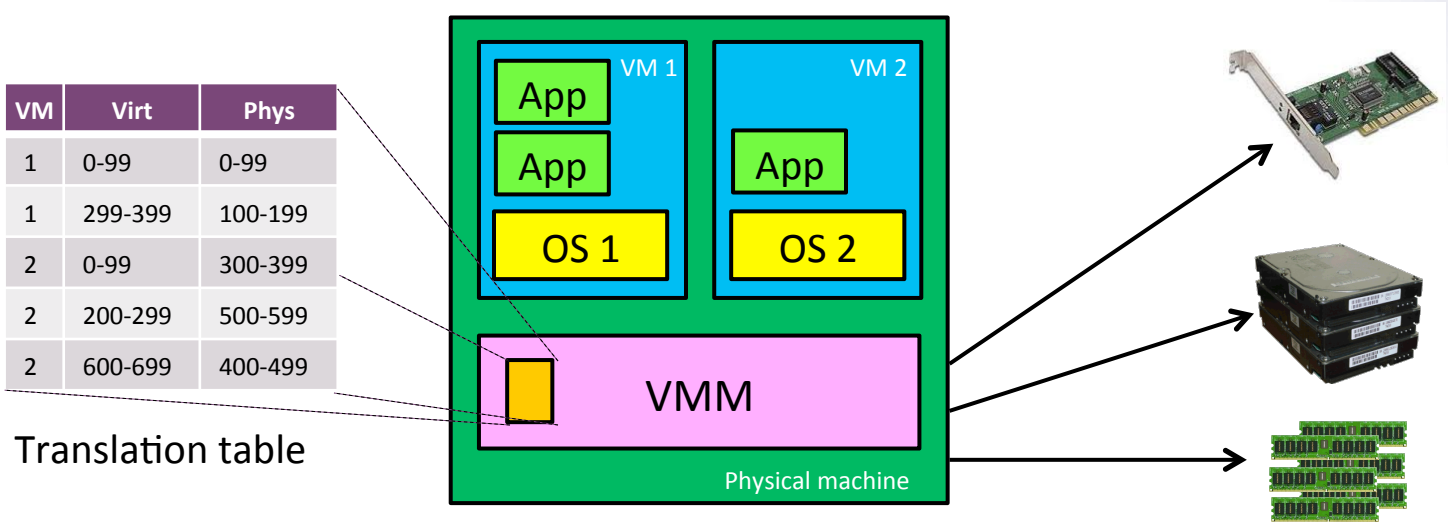
- Suppose Alice has a machine with 4 CPUs and 8 GB of memory, and three customers:
  - Bob wants a machine with 1 CPU and 3GB of memory
  - Charlie wants 2 CPUs and 1GB of memory
  - Daniel wants 1 CPU and 4GB of memory
- What should Alice do?

What is virtualization?



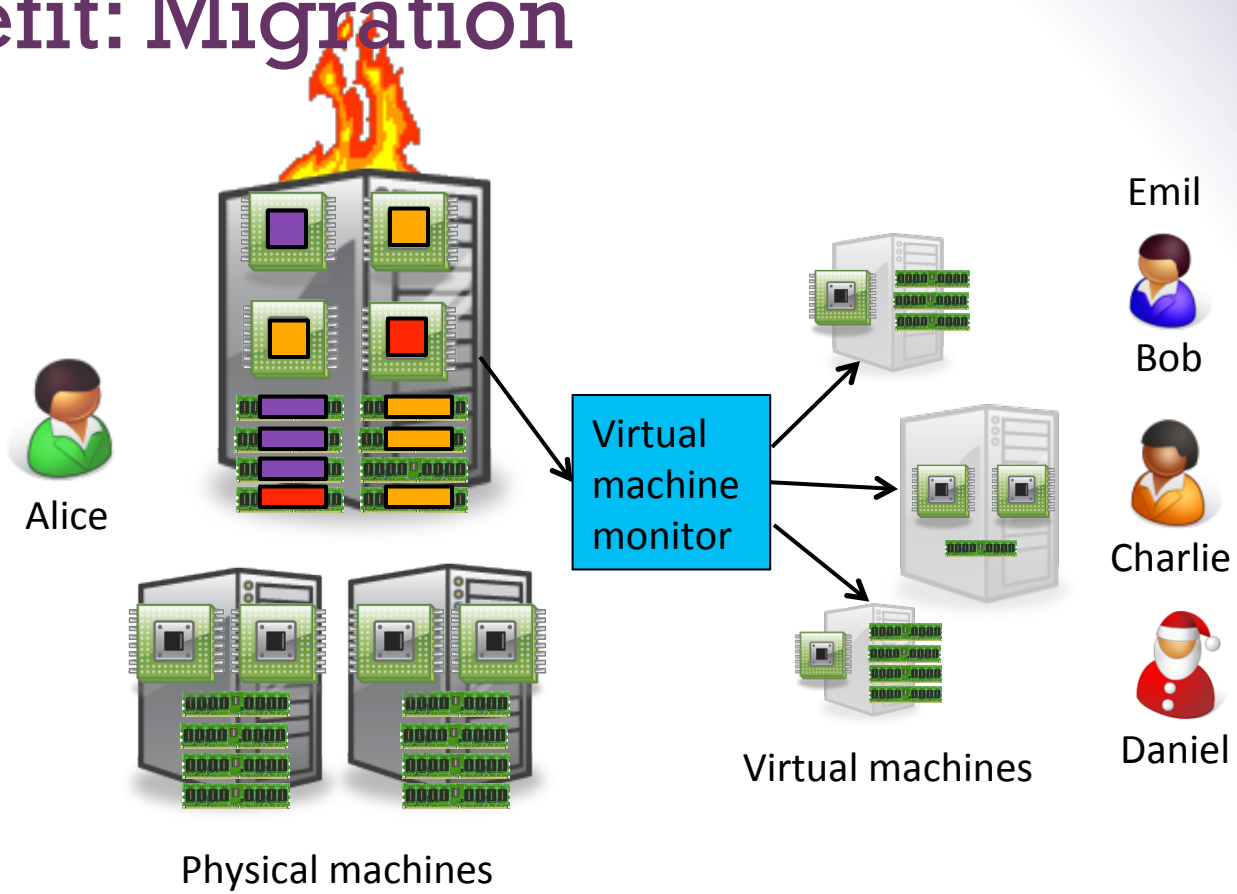
- Alice can sell each customer a virtual machine (VM) with the requested resources
  - From each customer's perspective, it appears as if they had a physical machine all by themselves (isolation)

How does it work?



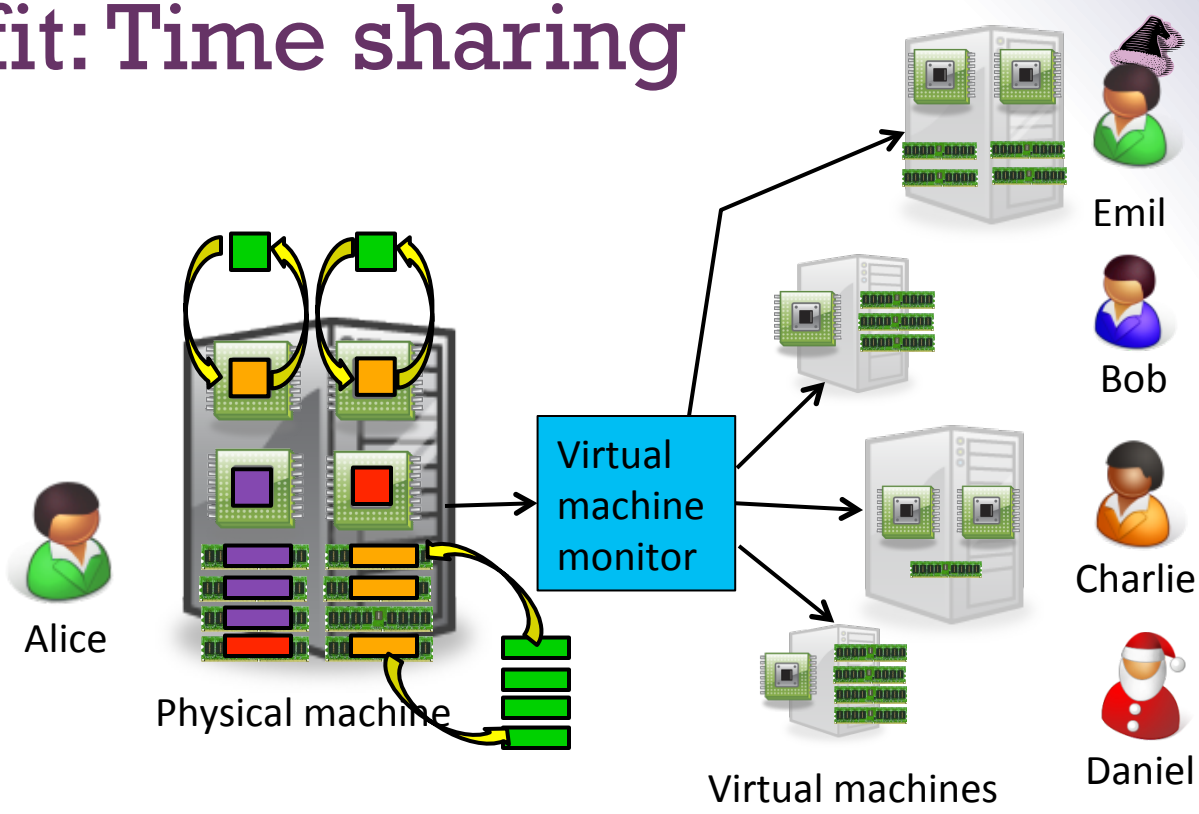
- Resources (CPU, memory, ...) are virtualized
  - VMM ("Hypervisor") has translation tables that map requests for virtual resources to physical resources
  - Example: VM 1 accesses memory cell #323; VMM maps this to memory cell #123.
  - For which resources does this (not) work?
  - How do VMMs differ from OS kernels?

Benefit: Migration



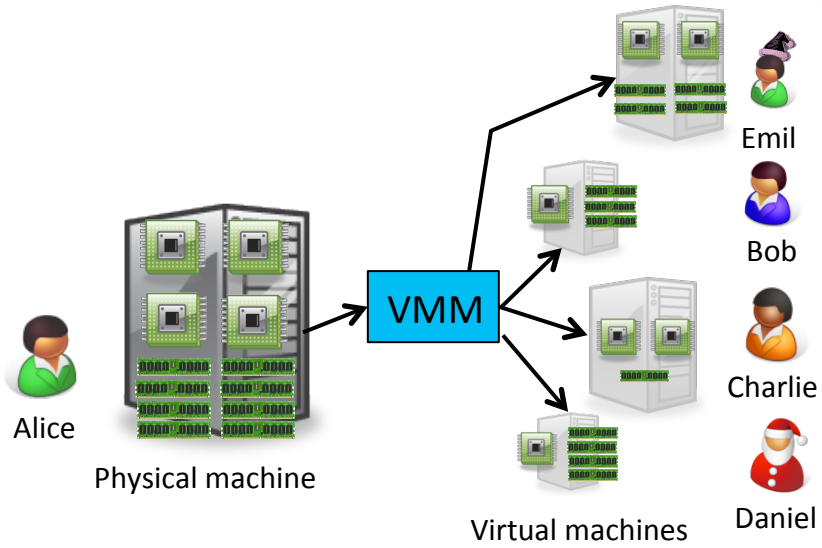
- What if the machine needs to be shut down?
  - e.g., for maintenance, consolidation, ...
  - Alice can migrate the VMs to different physical machines without any customers noticing

Benefit: Time sharing



- What if Alice gets another customer?
  - Multiple VMs can time-share the existing resources
  - Result: Alice has more virtual CPUs and virtual memory than physical resources (but not all can be active at the same time)

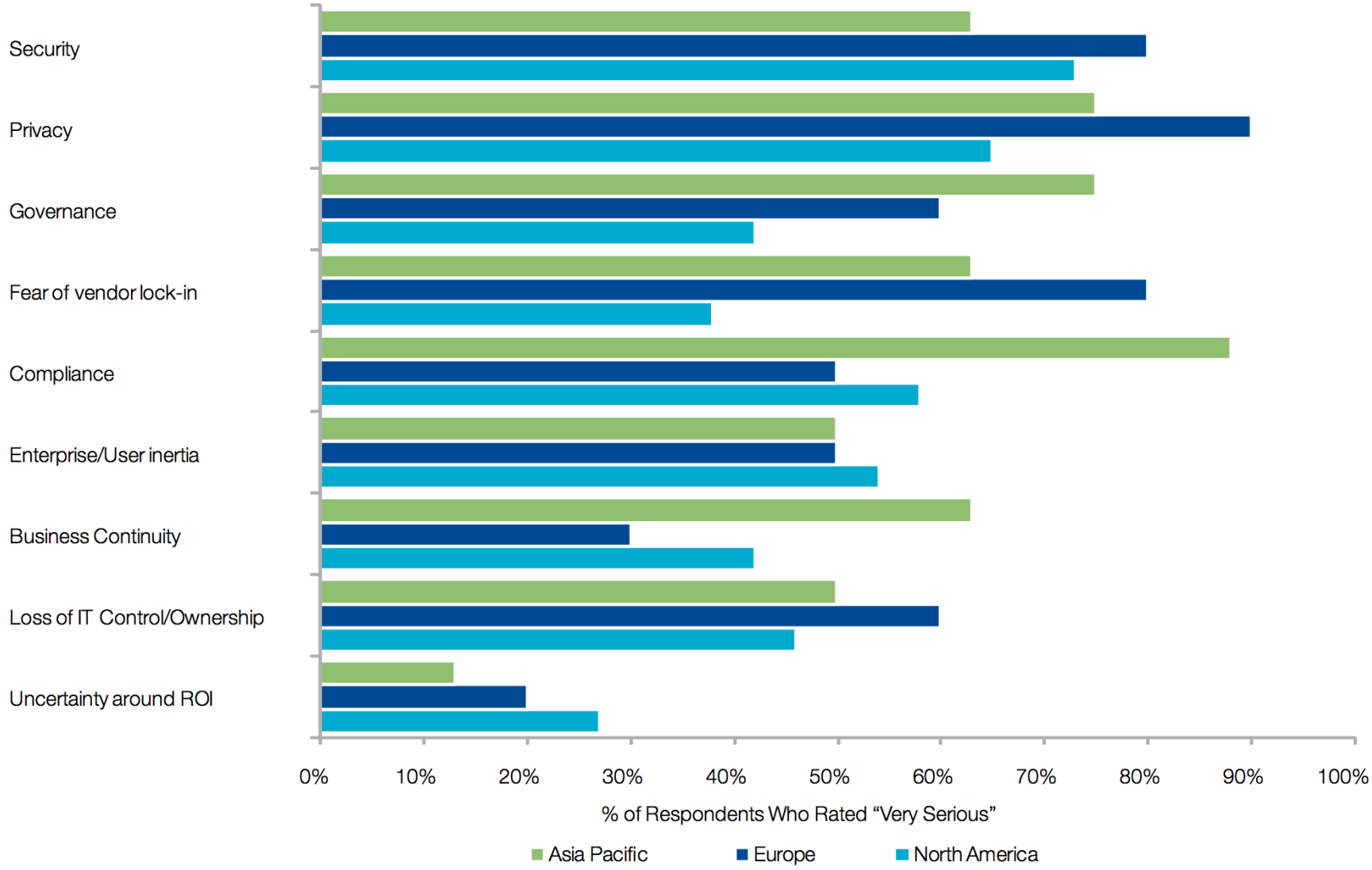
Benefit and challenge: Isolation



- Good: Emil can't access Charlie's data
- Bad: What if the load suddenly increases?
  - Example: Emil's VM shares CPUs with Charlie's VM, and Charlie suddenly starts a large compute job
  - Emil's performance may decrease as a result
  - VMM can move Emil's VM to a different CPU, or migrate it to a different machine

Security challenges of cloud computing

Survey: impediments to adoption



Source: Cloud Computing Survey 2009, World Economic Forum

Security challenges

- Traditional threats with cloud-specific twists
  - Impact amplified by vast amount of available resources
  - Must secure the infrastructure
  - Authentication & authorization
  - Protect against DDoS, phishing, SQL injections, cross-site scripting, etc.
- VMM vulnerabilities
  - New attack channels for malicious users
  - Starvation of resources
  - VM side-channel attacks by rouge VM
  - Buffer overflow attacks
  - Identify path followed by attacker is much more difficult

Security challenges

- Data confidentiality and auditability
  - How do I make sure that the cloud doesn't leak my confidential data?
  - Can I comply with regulations? E.g., will the stored data remain in EU?
- Availability
  - What happens to my business if there is an outage in the cloud?
- Data lock-in
  - How do I move my data from one cloud to another?

Service	Duration	Date
S3	6-8 hrs	7/20/08
AppEngine	5 hrs	6/17/08
Gmail	1.5 hrs	8/11/08
Azure	22 hrs	3/13/09
Intuit	36 hrs	6/16/10
EBS	>3 days	4/21/11
ECC	~2 hrs	6/30/12

Some recent cloud outages

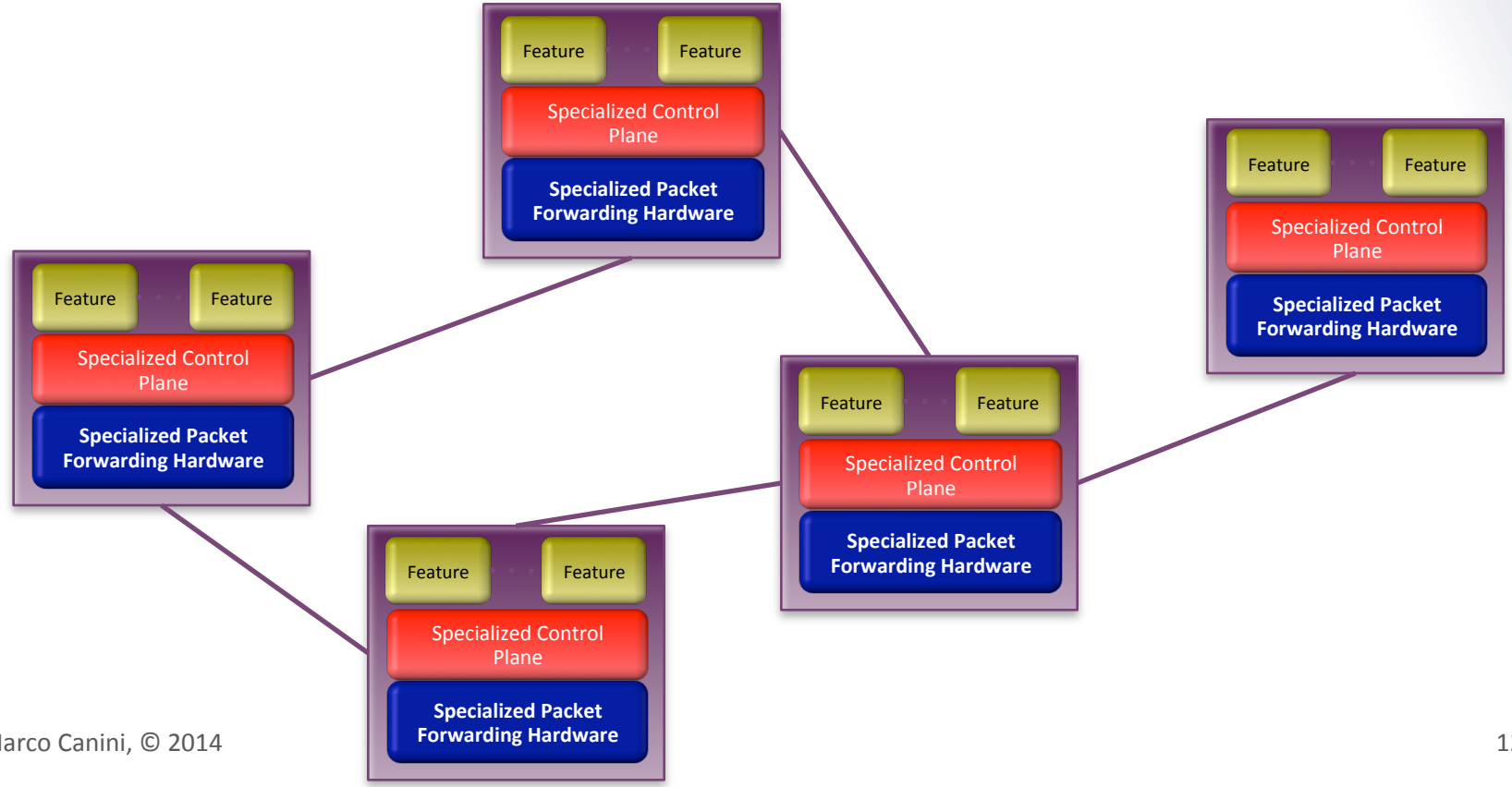
Security challenges

- Third-party control
  - Lack of transparency and limited user control
  - How can we trust resources? What if cloud provider subcontracts resources?
- Abuse of cloud
  - APIs are not fully secure
  - Malicious insiders (employees of cloud provider)
  - Data loss and leakage

Software-Defined Networking (SDN)

Classical network architecture

- Distributed control plane
- Distributed protocols compute routing state
  - OSPF, IS-IS, BGP, etc.



Networks are Hard to Manage

New requirements led to great complexity

- Security, network virtualization, VM migration, perf. isolation, …

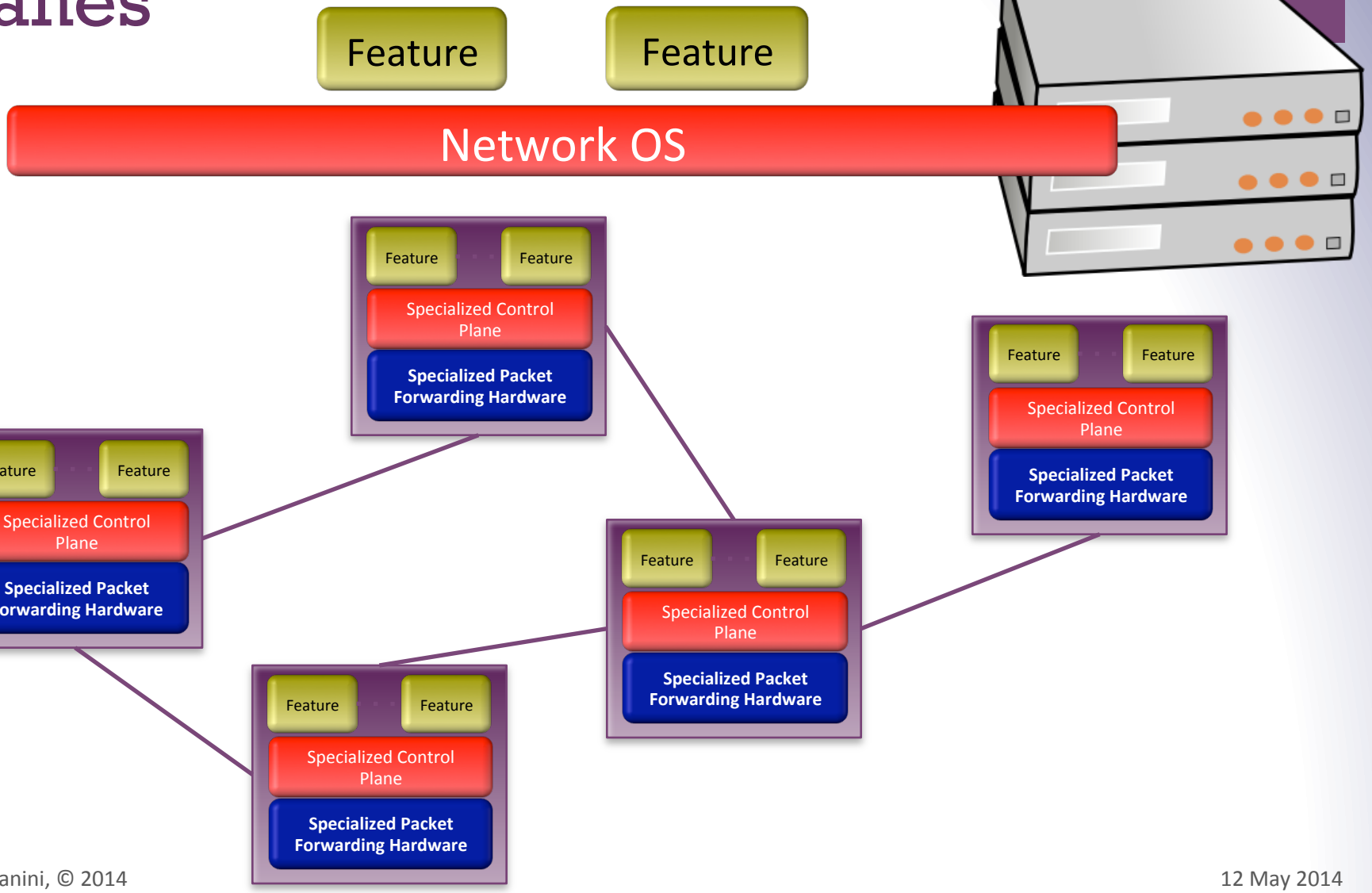
Kept working by
"**Masters of Complexity**"

When things don't work?

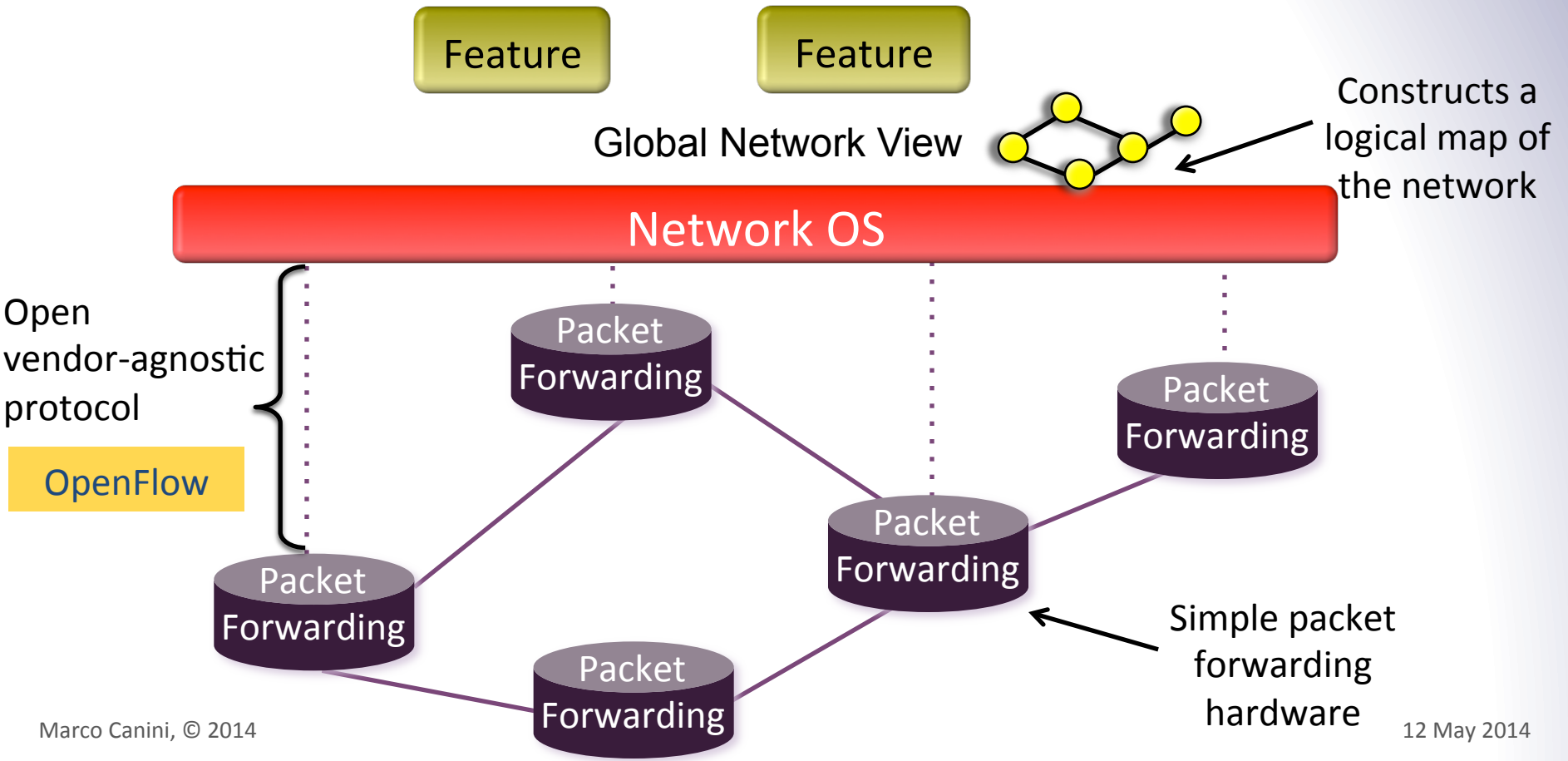
- Only **limited tools**:

```
ping, traceroute, tcpdump, SNMP, NetFlow
```

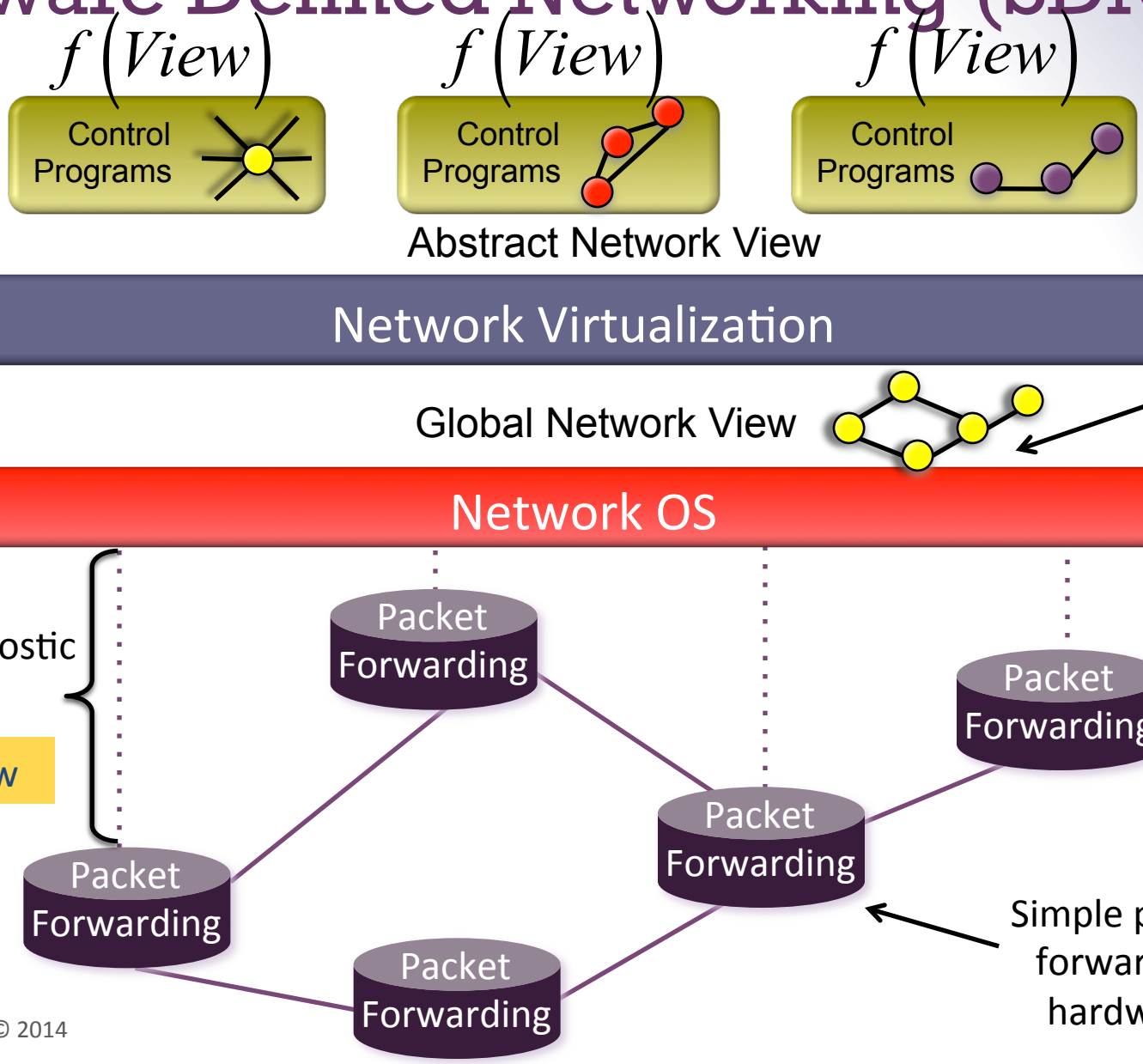
Separation of control and data planes



Software Defined Networking (SDN)



Software Defined Networking (SDN)



Software Defined Networking (SDN)

$f(View)$

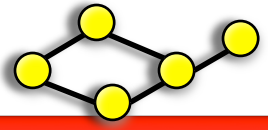
Control Programs

```
firewall.c
    …

    if( pkt->tcp->dport == 22)
            dropPacket(pkt);

    …
```

Abstract Network View

Network Virtualization

Global Network View



Network OS

Packet Forwarding

1. <Match, Action>
2. <Match, Action>
3. <Match, Action>
4. <Match, Action>
5. <Match, Action>
6. …
7. …

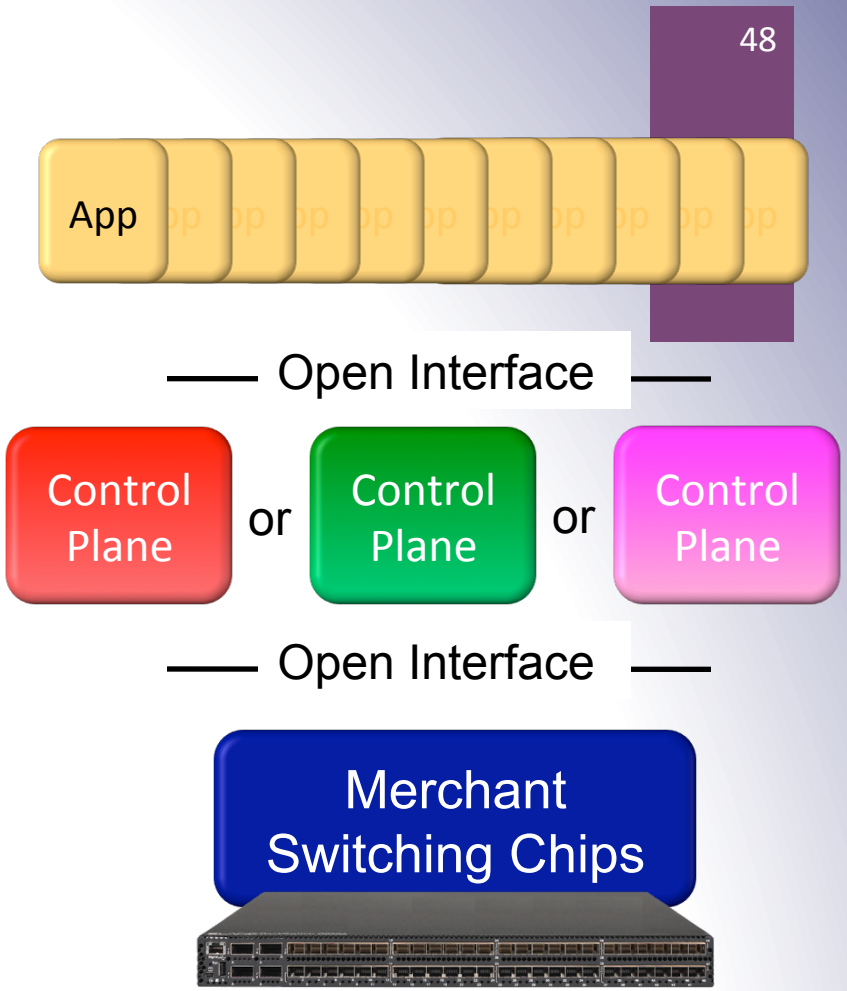
What problem does SDN solve?

- Great tool to enable innovation in network control
- Platform to help solve longstanding problems in managing networks and deploying new functionality

SDN applications

- Network management
  - Enterprise access, middleboxes waypointing
- Monitoring and measurement
- Network virtualization
- User mobility and VM migration
- Server load balancing
- Traffic engineering
  - Energy efficiency, high WAN utilization
- Exposing an API to host applications

Timing matters



Vertically integrated
Closed, proprietary
Slow innovation

Horizontal
Open interfaces
Rapid innovation

Where SDN will be deployed

1. Multi-tenant "virtualized" data centers
   - Public and private clouds
2. WANs
   - Google WAN; eventually, public WANs
3. Enterprise networks
   - Greater control, fewer middleboxes
4. IXP networks
   - More flexible peering policies

Where SDN will be deployed (2)

5. Home networks
   - Outsourced management

6. Cellular Networks
   - Separation of service from physical infrastructure

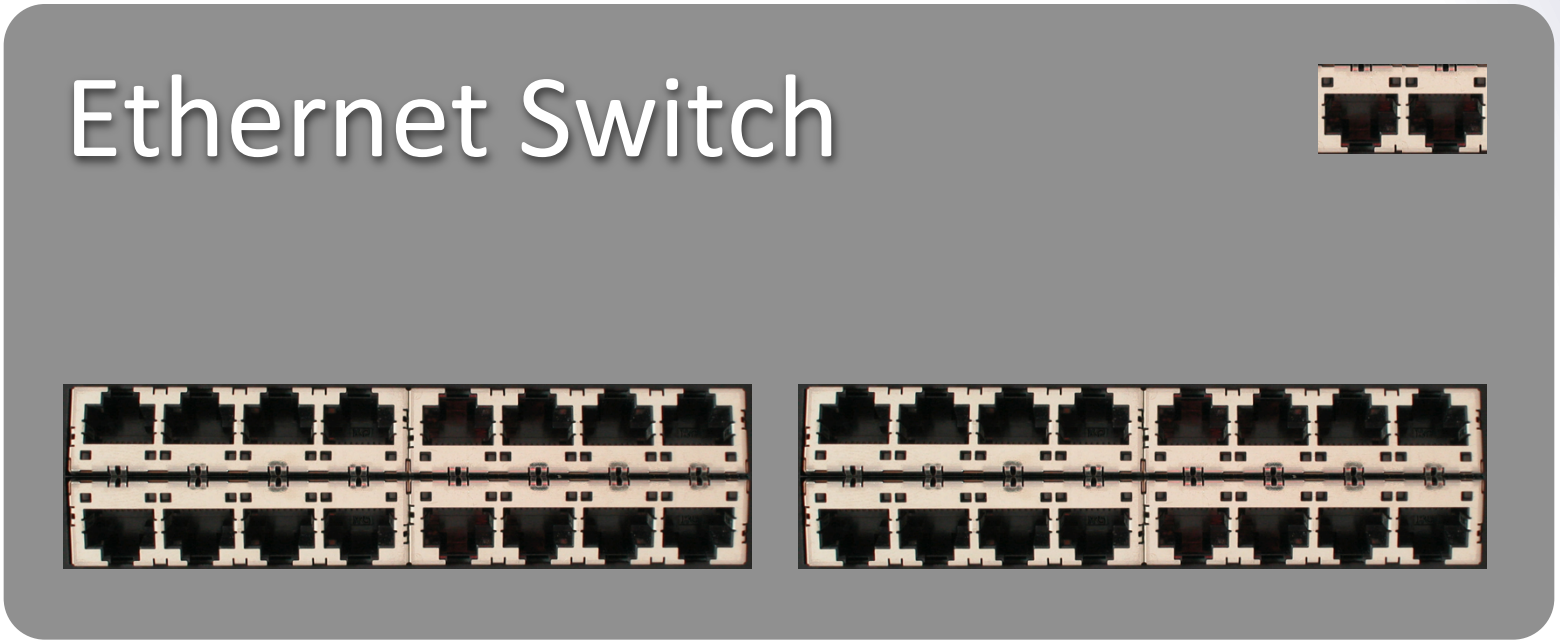
7. Research and Education Networks
   - National backbones
   - College campus networks (@ UCL too)

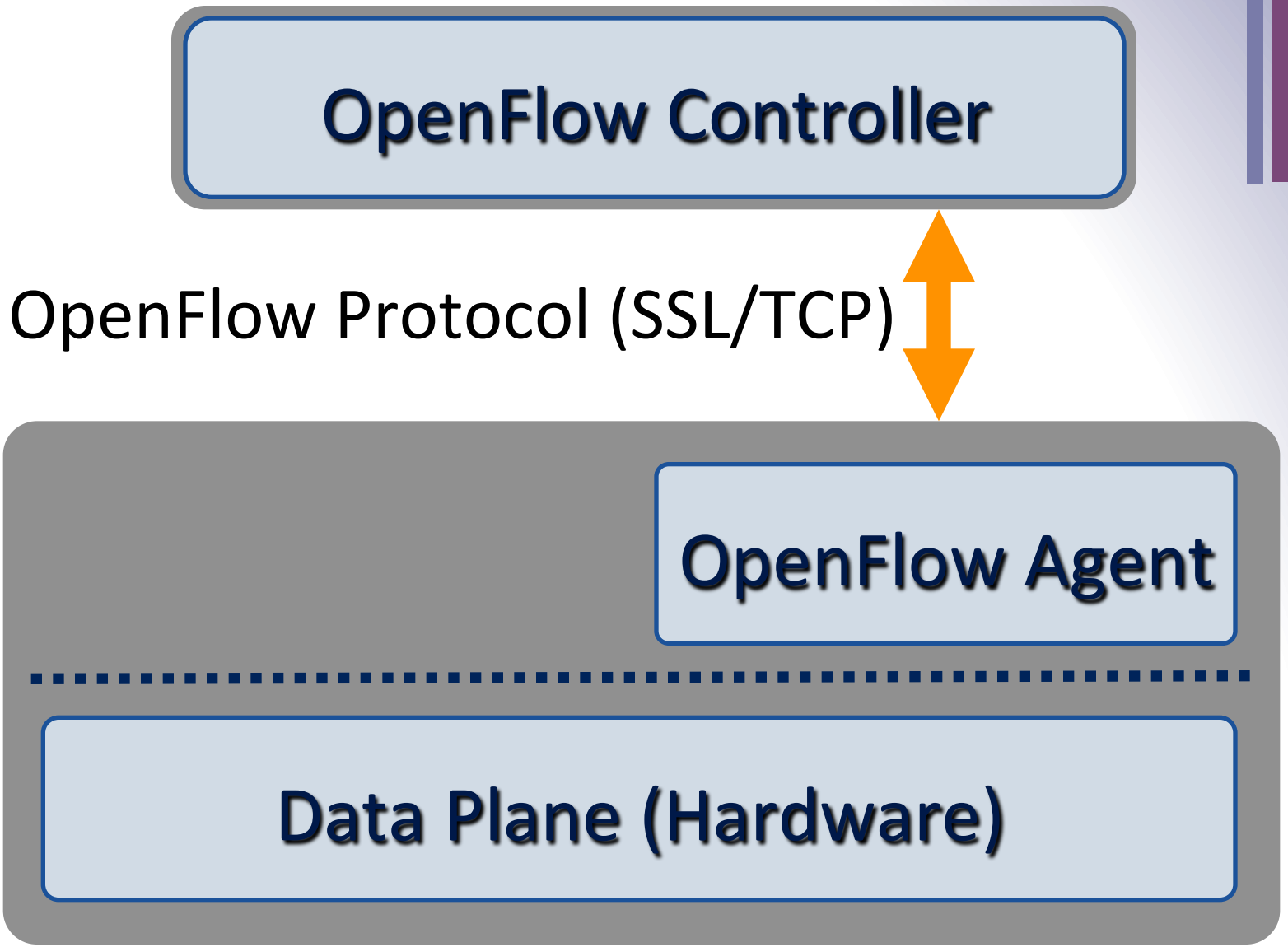
OpenFlow

OpenFlow

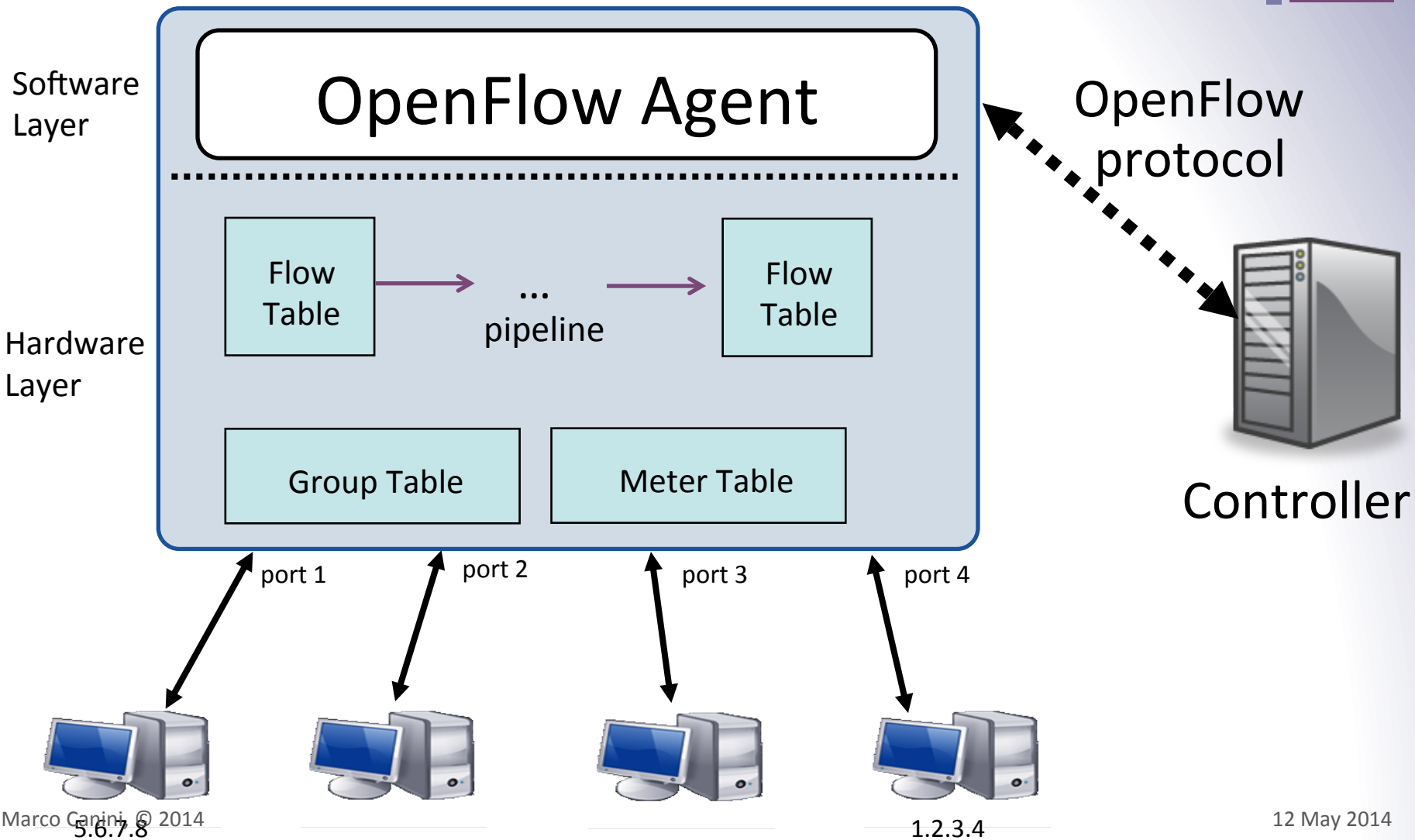
- is a protocol for remotely controlling the forwarding table of a switch or router
- is one element of SDN

How does OpenFlow work?




OpenFlow Controller
OpenFlow Protocol (SSL/TCP)
OpenFlow Agent
Data Plane (Hardware)

Main components of an OpenFlow switch



Flow Table Entries

Main components of a flow entry in a flow table.

Match fields	To match against packets. These consist of the ingress port and packet headers
Priority	Matching precedence of the flow entry
Counters	e.g. packet and byte counters
Instructions	Determine action set or pipeline processing
Timeouts	Maximum amount of time or idle time before flow is expired by the switch
Cookies	Opaque data value chosen by the controller. Not used when processing packets.

Switch Port	VLAN ID	VLAN pcp	MAC src	MAC dst	Eth type	IP Src	IP Dst	IP ToS	IP Prot	L4 sport	L4 dport

The match field contains either a specific value or a “wildcard”

Match/action examples

Switching

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

Flow Switching

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

Firewall

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

Examples

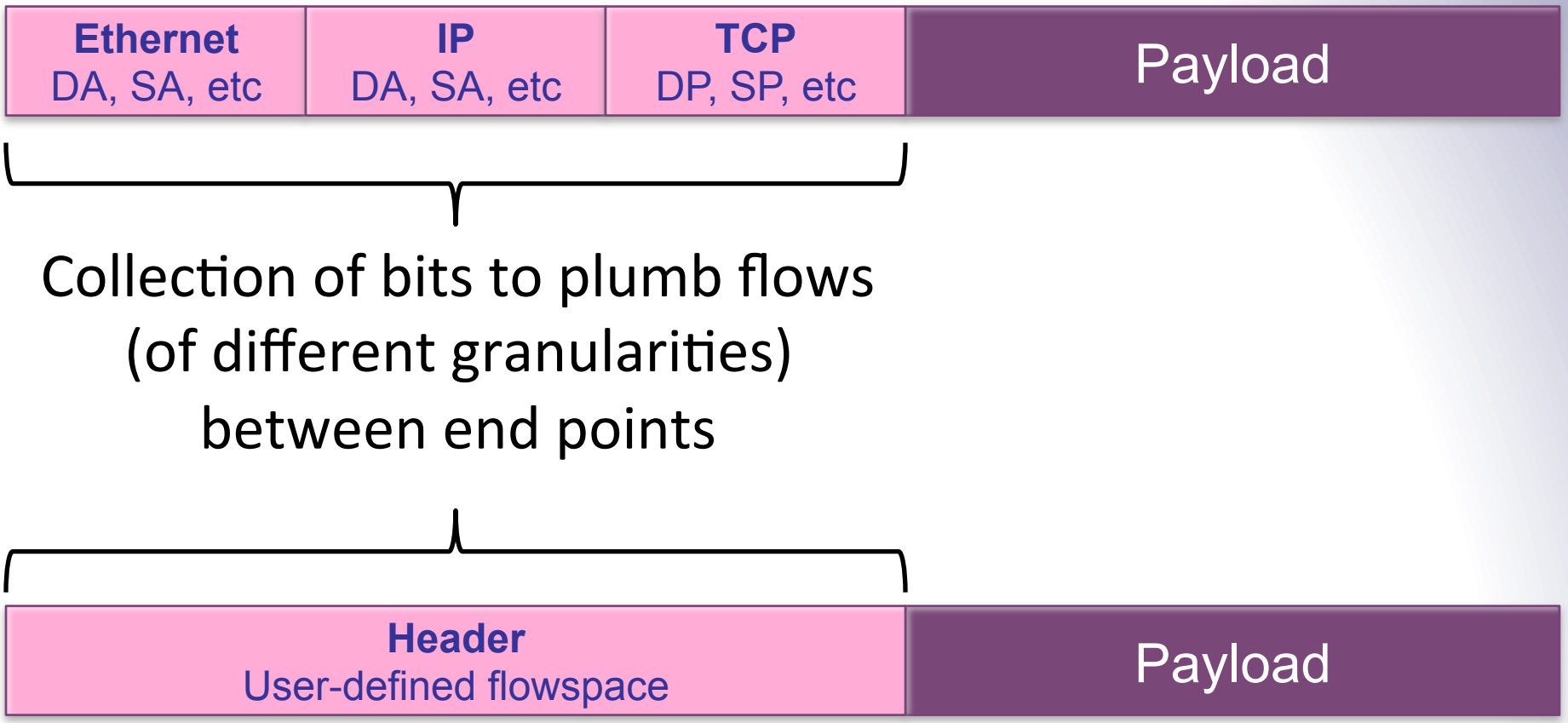
Routing

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

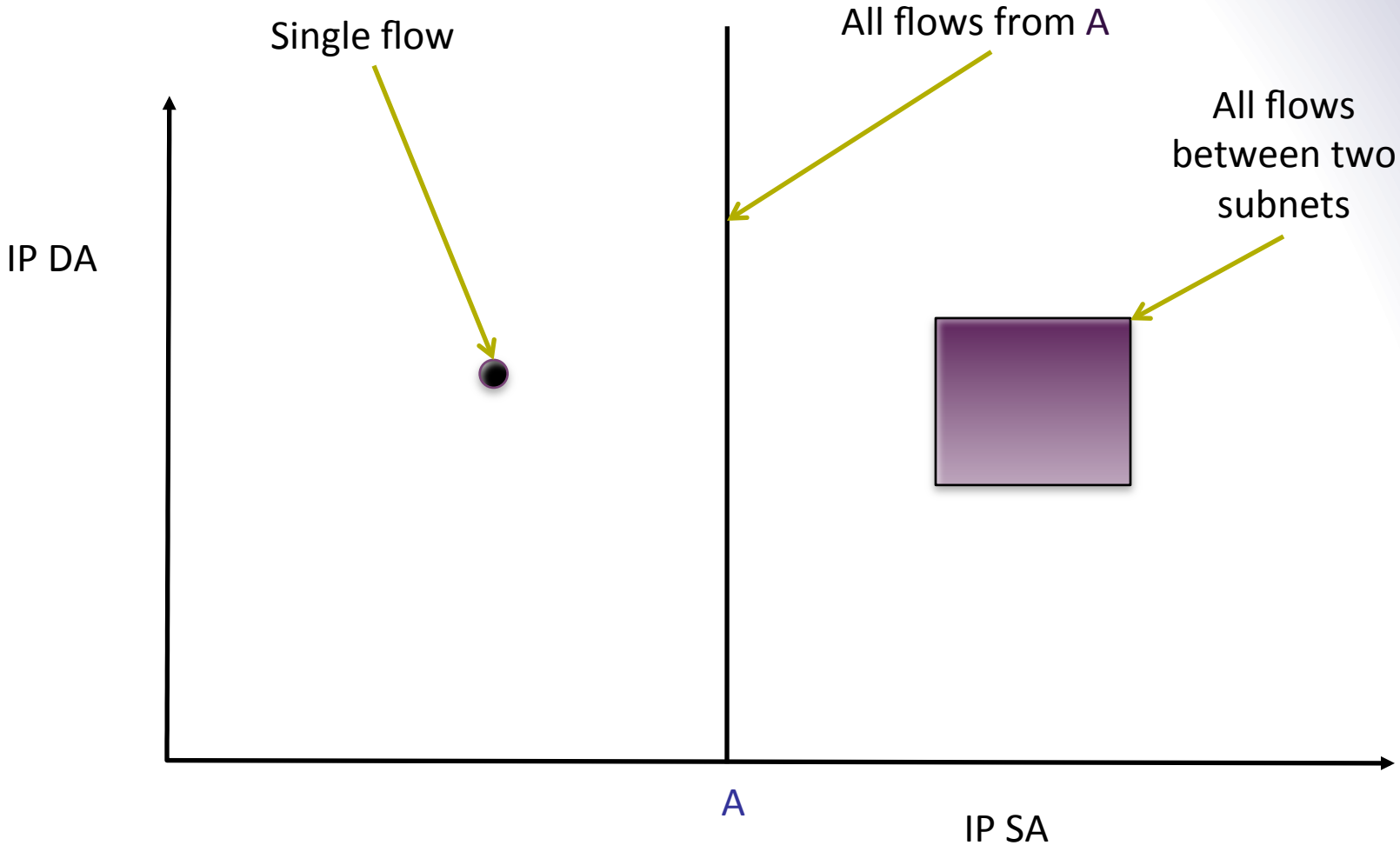
VLAN Switching

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

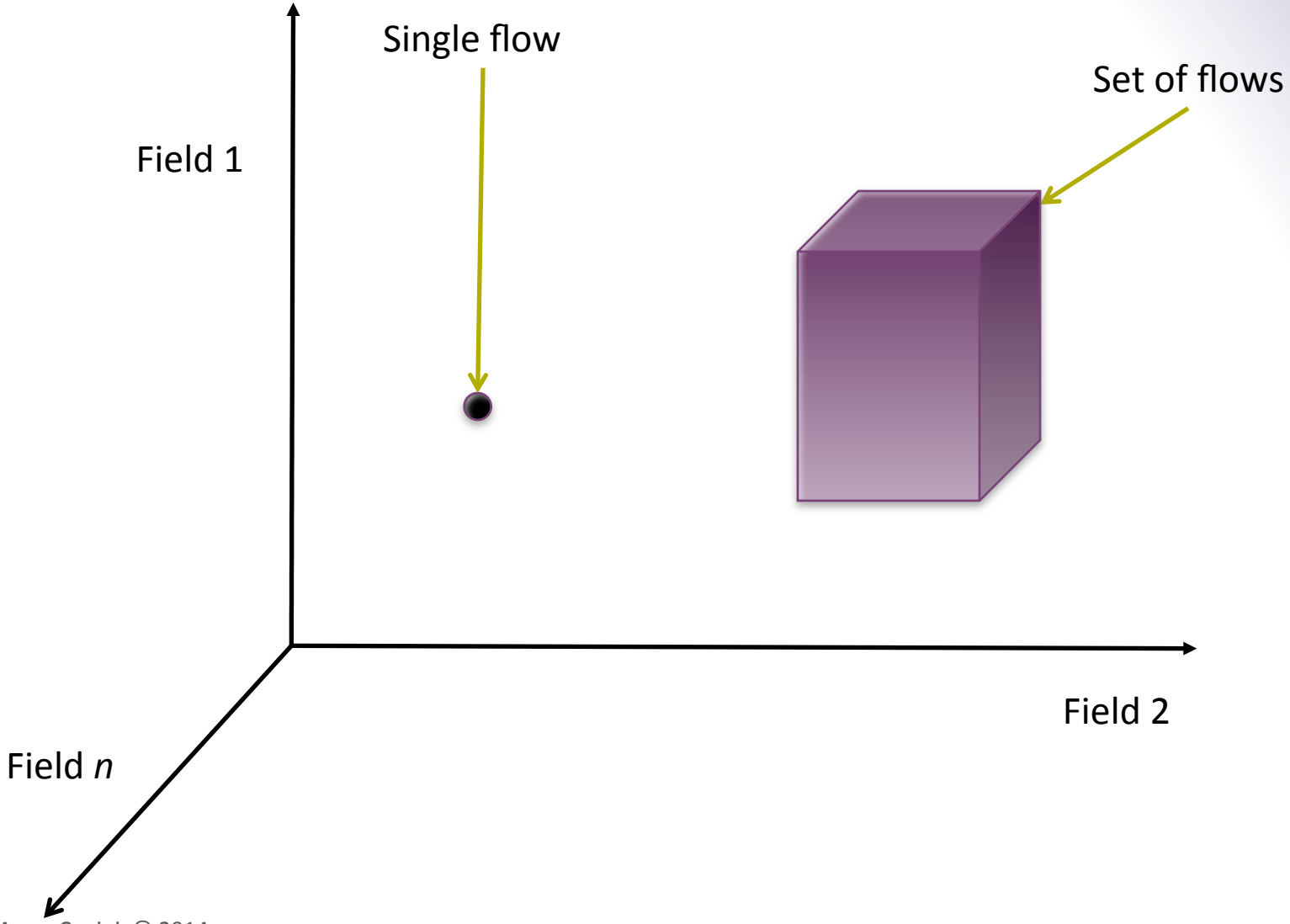
Headers as a protocol-agnostic collection of bits



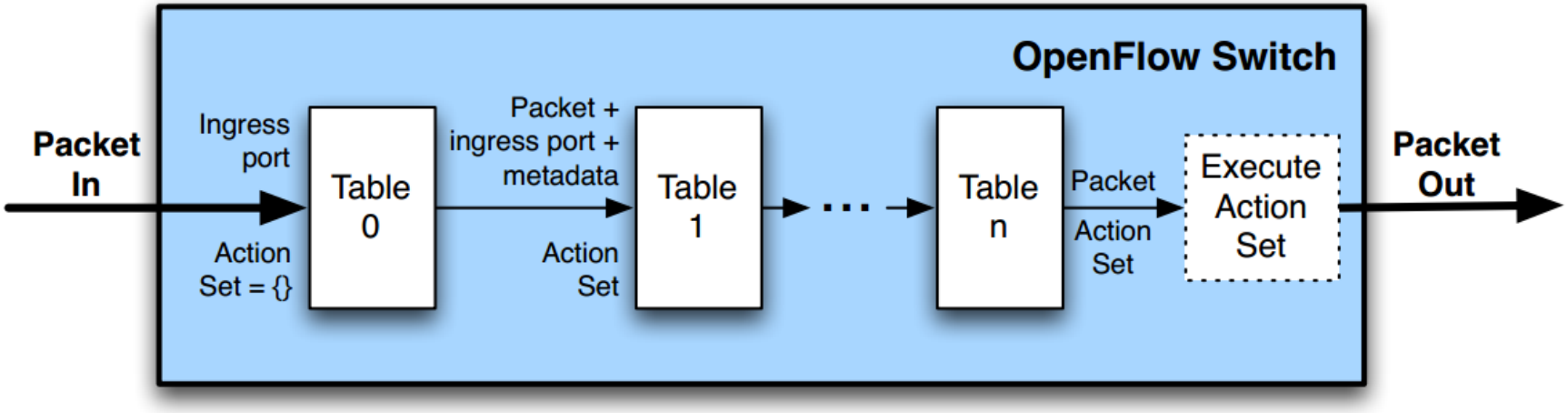
"Flowspace": A way to think about flows defined by match fields



Flowspace: Generalization



OpenFlow Pipeline



Packets are matched against multiple tables in the pipeline

OpenFlow Switch Specification Version > 1.1.0

Actions

- Forward on a port
- Flood
- Modify packet field
- Push/pop tag
- Forward to controller
- Drop (implicit if no actions)

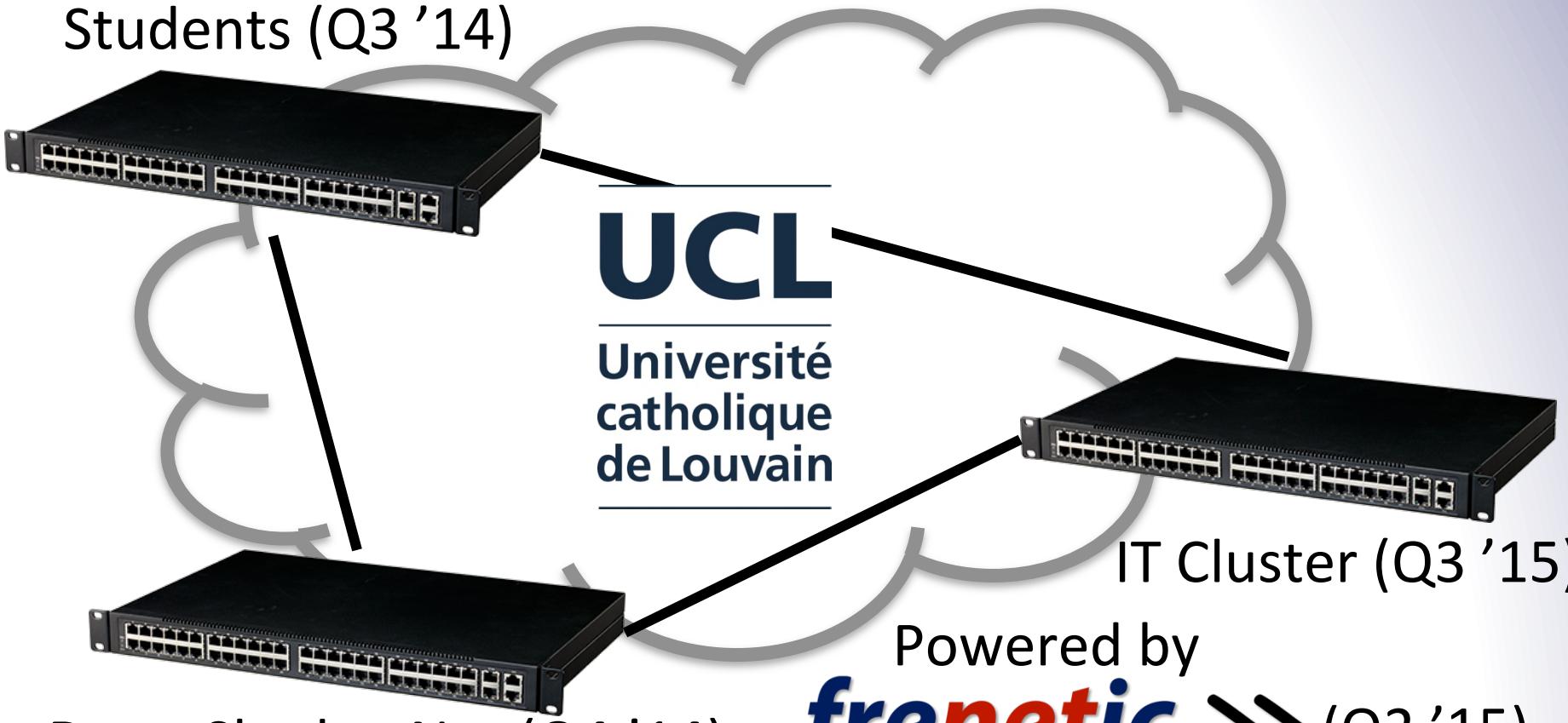
Table miss

- Packets for which no flow has been defined are normally sent to the controller
- The controller then defines a new flow for that packet and creates one or more flow table entries
- The packet is then processed as determined by the newly created flow entries
- By default packets unmatched by flow entries are dropped

OpenFlow key messages

Message	Direction	Description
Packet-In	Switch->Controller	Transfer the control of a packet to the controller. Packet-in events can be configured to buffer packets
Packet-Out	Controller->Switch	Instruct switch to send a packet out of a specified port. Send in response to Packet-in messages.
Modify-State	Controller->Switch	Add, delete and modify flow/group entries in the flow tables and to set switch port properties
Flow-Removed	Switch->Controller	Inform the controller about the removal of a flow entry from a flow table

UCL Deployment



Any questions?