



# Networking in Massachusetts Open Cloud (MOC)

Somaya Arianfar

July 30, 2015

# Clouds today

- Public and private cloud providers
  - Amazon, Microsoft,...
- Shared pools of resources
  - Compute, storage, and network
- Single provider acts as a single proxy
  - buys, controls and manages HW/SW

# Cloud provider: the main player

- For users
  - No easy way to go across clouds
  - No easy way to choose the HW/SW
  - Limited influence and control on managing allocated resources
- For cloud provider itself
  - Expansion could mean buying new HW
- For HW provider
  - HW needs to be bought by the cloud provider

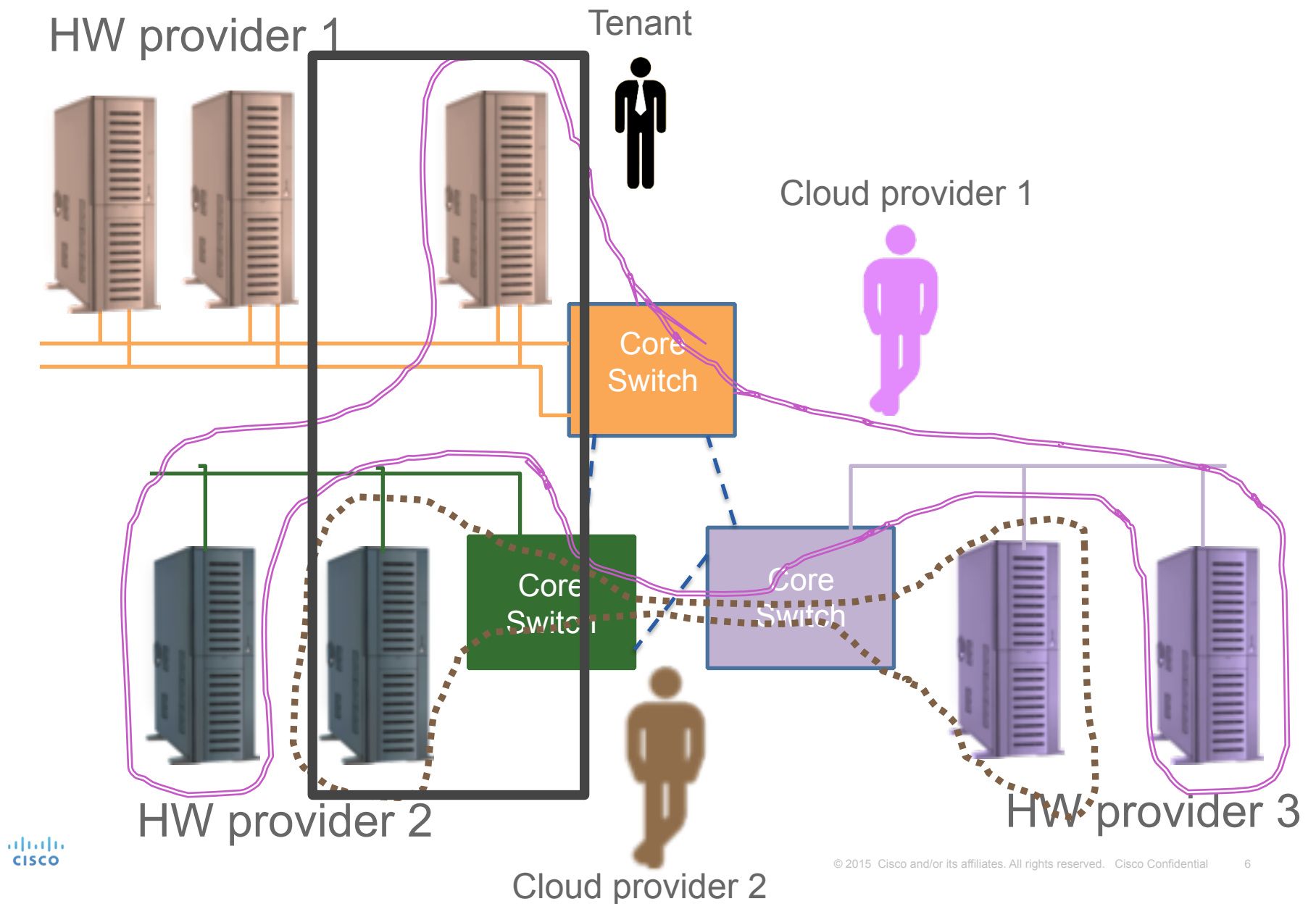
# Mass Open Cloud vision

- Providing more flexibility, openness, and control
- Removing single cloud dependencies and bindings
  - For the users
    - Across clouds, across administrative domains
  - For the HW/cloud providers
    - HW providers and HW admin **vs.** cloud providers **vs.** tenants
      - HW as an independent component should be **directly rented out** to the interested parties and be controlled by them (as opposed to be **sold**)
    - Recursive resource visibility and (probably) control

# Starting point

- Building up from the hardware
  - Hardware could be directly rented out to IaaS provider or to the users
- IaaS provider/user network could span multiple HW domains
- User network could span multiple IaaS provider domains

# MOC



# Networking

- MOC vision relatively easy to achieve for compute and storage
  - What about the network?
    - A set of shared and distributed resources
- Networking in today's cloud
  - Mainly used to connect storage and compute resources
    - Connecting VMs requires mapping/encapsulation/translation/authorization
    - Not necessarily an independent service, part of the infrastructure
  - Networking choices are made by the network/transit provider to serve their goals
    - Depending on how much control/flexibility they actually have over the network
  - Limited network service exposure to the tenant
    - E.g. firewall as a service (networking?), no redundancy elimination as a service

# Networking questions within MOC

- How can different networking providers participate in the market in a first class way, given:
  - The coupling of networking to compute and storage
  - The need to share networking resources
- How can we provide IaaS providers the same kinds of control (e.g., SDN) that they have when they own the HW?
- Can we provide networking services and SLAs to higher level providers and tenants when we have different HW providers?
- How can we support interoperability for tenants that span providers who use different networking technologies (e.g., VLAN, VXLAN, TRILL, SDN...)



Question:

How can we provide IaaS providers the same kinds of control (e.g., SDN) that they have when they own the networking HW?

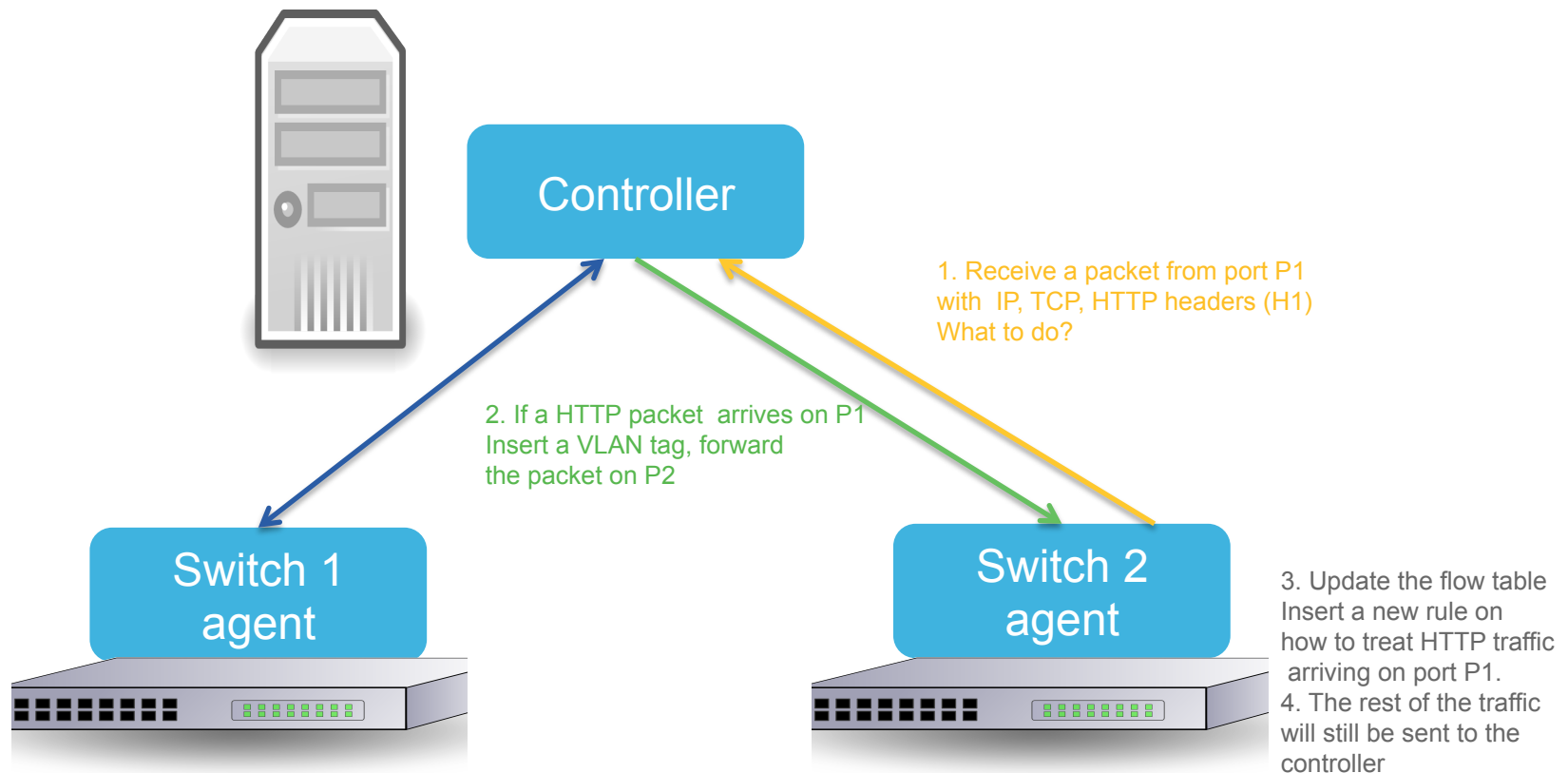
# Network control and data plane sharing

- Questions and concerns:
  - The same networking resources must be shared among/controlled by multiple SW or cloud providers/tenants (solved for links, but not for switches)
  - Who manages what
  - Who interacts with whom
  - What is the minimal infrastructure to support this
  - ...
- Outcome: offering network resources and their control as an independent service
- Means: full partitioning/virtualization of network resources

# Baseline: SDN

- SDN (Software Defined Networking)
  - A first step to look into full network partitioning/virtualization
    - Separation of control plane and data plane in the switches
      - Data plane is traditionally shared, but not the control plane
    - Could make it easier to share switches in addition to the links
- Centralized model
  - Easier configuration and management
    - E.g. reconfigure all the ports in all the switches
  - Easier change of control plane behavior
    - E.g. run a different routing algorithm other than the shortest path
  - More Flexibility (depending on the implementation)
    - E.g. make different decisions for HTTP traffic vs. the rest of the traffic

# SDN (existing work)



# SDN use case – B4

- Google's inter-datacenter WAN
- Motivation
  - Simplified management
  - Deploying customized routing and traffic engineering protocols
  - Leverage raw speed of newest HW (servers)
- Limitations
  - No HW fault tolerance, deep buffering, large routing tables
  - Requires pre-existing knowledge of traffic patterns
    - E.g. to avoid requiring deep buffers
  - No fate sharing between HW and SW

# Limited flexibility

- Mainly one administrative domain
  - One HW admin who is also the SW admin
    - E.g. in B4
      - HW: Google's merchant switch silicon
      - SW: Google's controller
    - E.g. most private clouds
      - HW: Existing OpenFlow switches (Brocade, Juniper, etc) bought by the cloud provider
      - SW: Cloud admin's controller

# Networking challenges in MOC (again)

- Expose the network (hardware and control) as an independently manageable first class service
  - Added flexibility
  - Much more complicated than compute and storage
    - Network by itself is not a single resource under single administration, it is a distributed resource
    - Many HW components in this distributed pool MUST be controlled and shared among many users
- Getting the connectivity logic to work across multiple administrative/control domains
  - Recursively
    - Across multiple pods
    - Across multiple clouds
    - Even higher in the hierarchy?

# SDN for network resource sharing in MOC

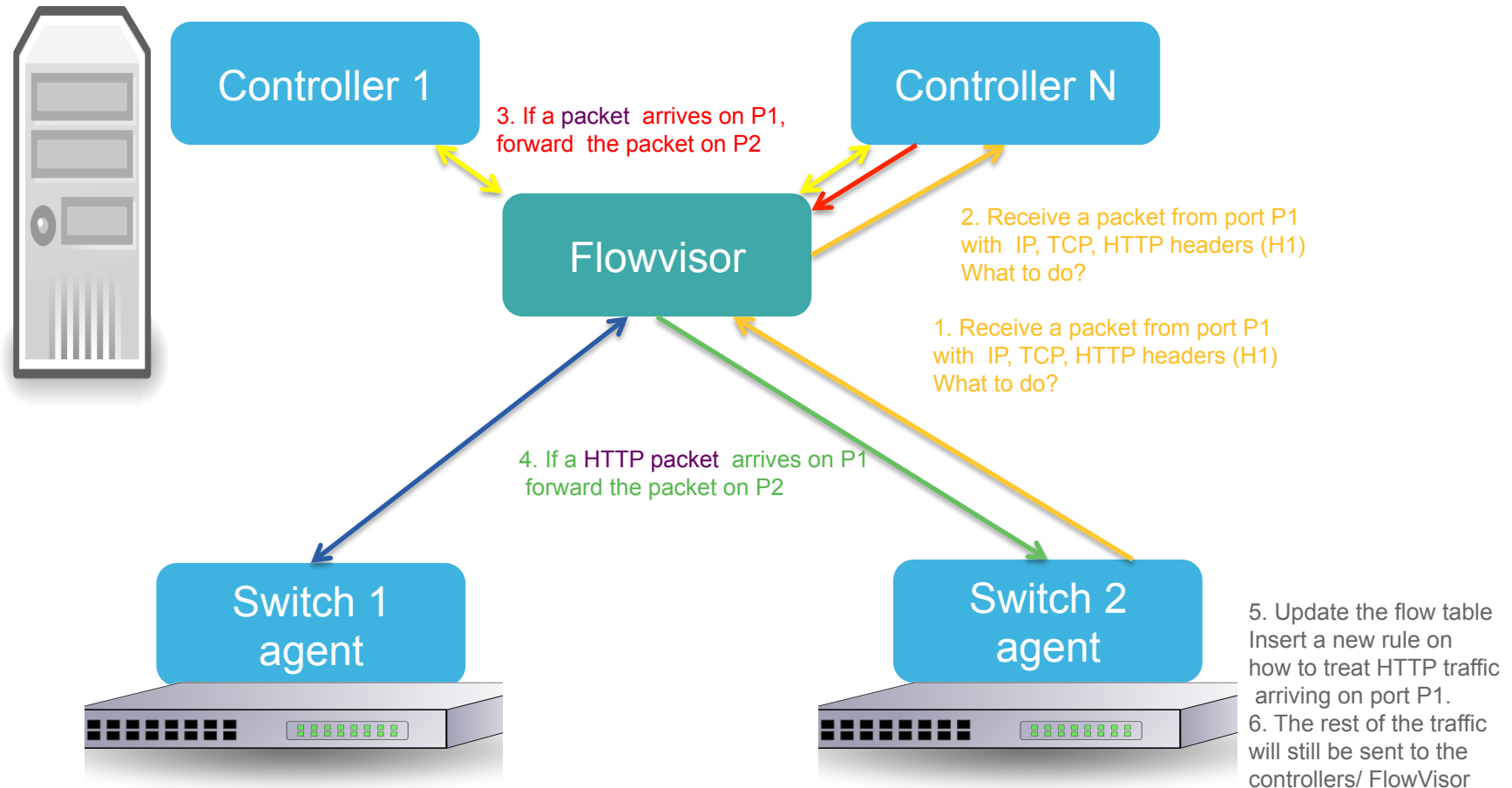
- Full flexibility and partitioning requirements
  - Sharing the control plane (**a modified form of SDN**)
- Crossing boundaries: Support for multiple control domains (**an extended form of SDN**)
  - Multiple HW domains
    - A SW control domain should be able to expand to multiple HW control domains
  - Multiple hierarchical SW domains
    - A tenant could have his own tenants
      - A SW control domain could be the host to another layer of a SW control domain and etc.



# Sharing the control plane: slicing/partitioning/ weak virtualization

- Slice: A subdivision of network resources (links, buffers, CPU,...)
- FlowVisor
  - A network slicing agent that sits between switches and controllers
  - Allows multiple SW controllers to share the same HW
  - Slices (stored in a local DB)
    - Switch based
    - Port based
    - Protocol based (VLAN, HTTP, ...), etc
  - Create slices (define the name and controller)
    - `fvctl add-slice slice1 tcp:192.168.56.101:5000 generic@email.com`
    - `fvctl add-slice slice2 tcp:192.168.56.101:5001 generic@email.com`
  - Define a flowspace for each slice
    - E.g. Assign switch number 3 port 1 to slice1 and port 2 to slice 2
      - `fvctl -f /dev/null add-flowspace dpid3 3 1 1 slice1=7`
      - `fvctl -f /dev/null add-flowspace dpid3 3 1 2 slice2=7`

# FlowVisor based slicing(existing work)



# MOC and control plane sharing with FlowVisor- problem 1

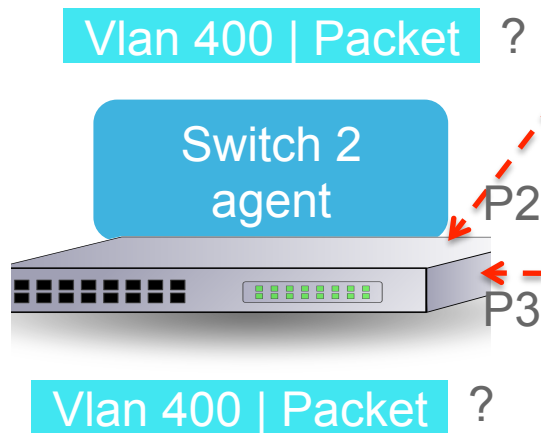
- Limited definition of slices to identify different tenants
  - E.g. Only VLAN based slices and no HTTP slices
- Problem
  - Bare-metal servers: no expectation from the servers to include the slice info to the packet header
- Solution
  - FlowVisor plays an active role in inserting the slice id into the packet header
    - As opposed to passively reading and mapping what is already in the header
    - Maps/translates/negotiates across multiple domains
  - Example slice definition and insertion
    - At the edge
      - Edge switches map different ports to different slices
        - **Insert a proper tenant/slice specific tag to each packet**

# MOC and control plane sharing with FlowVisor- problem 2

- Getting across multiple HW control domains (data centers/pods/...) for the same SW controller

## HWP1 FlowVisor instance view

- Tenant 20 :
  - Slice id 400
  - 50% BW on port 2 switch 2
  - 70% BW on port 3 switch 2



## HWP2 FlowVisor instance view

- Tenant 30:
  - Slice id 400
  - 30% BW on port 2 switch 2
  - 20% BW on port 3 switch 2



## HWP3 FlowVisor instance view

- Tenant 20:
  - Slice id 600
  - 10% BW on port 2 switch 5
  - 5% BW on port 3 switch 5



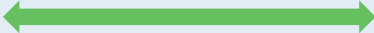
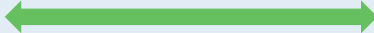

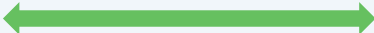


# Multi-domain slicing

- Multiple provider
  - Each provider (HW admin) runs its own instance of a FlowVisor
  - Each tenant (SW admin) can have a slice in each provider domain
- FlowVisor on its own is NOT sufficient
  - Crossing provider domain boundaries is difficult
    - Slicing (mainly FlowSpace definitions )
      - Done independently and manually in each domain
        - A dynamic mapping between tenant, slice, and VLAN might not be easy to enforce
      - Stored in administrator specific domains
        - Not directly negotiated/transferred/agreed across administrative domains
        - How to manage the traffic traversing the border between 2 administrative domains (2 different slice IDs map to the same tenant in each domain)
    - Figuring out the destination provider domain(s) for different tenants
      - Multicast/broadcast traffic

# Further thoughts and challenges

- Limitations and possibilities of a **recursive design in a hierarchy**
  - Adding another layer of tenant specific FlowVisors (tenants of tenants)
    - Traffic isolation
      - Might need more work on FlowVisor: e.g. OpenFlow 1.3 support in FlowVisor or even adding new protocol options
    - Flow table, BW divisions, **trust issues** ,...
    - HW admin creates class specific queues for each tenant, who would create queues for the tenants of the original tenant,.....
  - Scalability
- Going across multiple SW control domains
  - Cross cloud exchange
- Network view unification across multiple HW control domain
  - Expose an overall SW controller to a specific SW domain instead of exposing multiple SW controllers (one per each HW control domain)

# Summary of network resource sharing challenges in MOC

	Challenge 1	Challenge 2
Multiple HW providers/ multiple tenants (Orchestration)	Traffic isolation: Propagating tenant/slice mappings across administrative boundaries (through negotiation and etc.) 	Bandwidth (QoS) isolation: Agreeing on the case when same tenant gets different proportion of BW in different HW domains 
Network view unification	Expose a unified view of the network to the tenant instead of one view per HW administrative domain 	
Trust relationships	Who enforces what	
Recursive design	Bridging from cross pods to cross clouds  	Enforce a virtualization mechanism that allows tenants to have their own tenants 



# Tools and languages

- OpenFlow 1.0
  - Protocol spec
- FlowVisor
  - Available on github
- Mininet
  - SDN network simulator with one controller
  - Available on github (uses linux namespaces)
- Java
  - Jason
  - Jetty
- Python

# Conclusions

- An Open Cloud eXchange model like MOC introduces fundamental new challenges from the networking perspective
- We are starting new research focused on multi-tenant/provider SDN
- We are looking for collaborators...



*TOMORROW starts here.*