Sharif University of Technology

Computer Engineering Department

Software-Defined Networking

Ali Movaghar

Mohammad Hosseini

Network Virtualization Part 2

TA: Iman Rahmati & Farbod Shahinfar

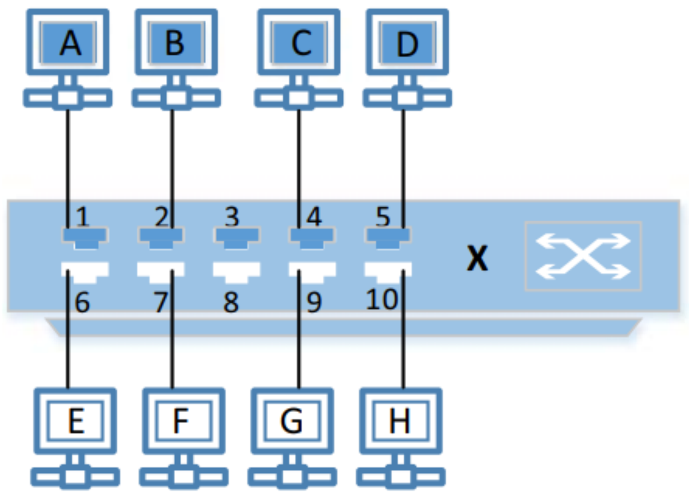
Virtual Networks

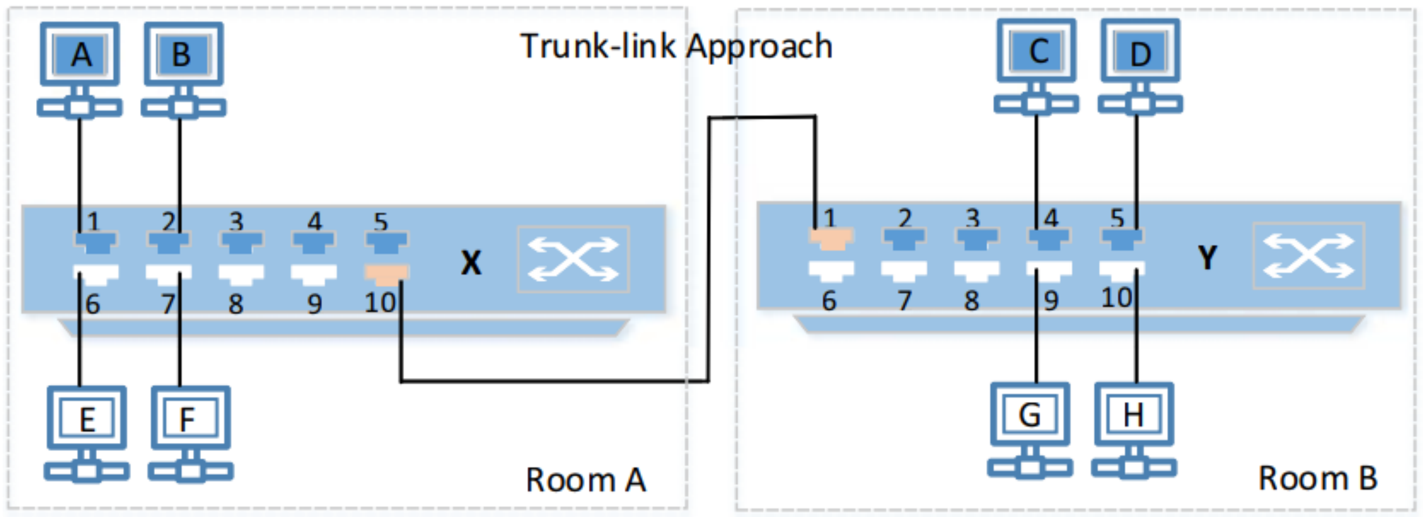
- Virtual networks have been a part of the Internet for years.

  - VLAN
  - VPN
  - MPLS
  - VRF

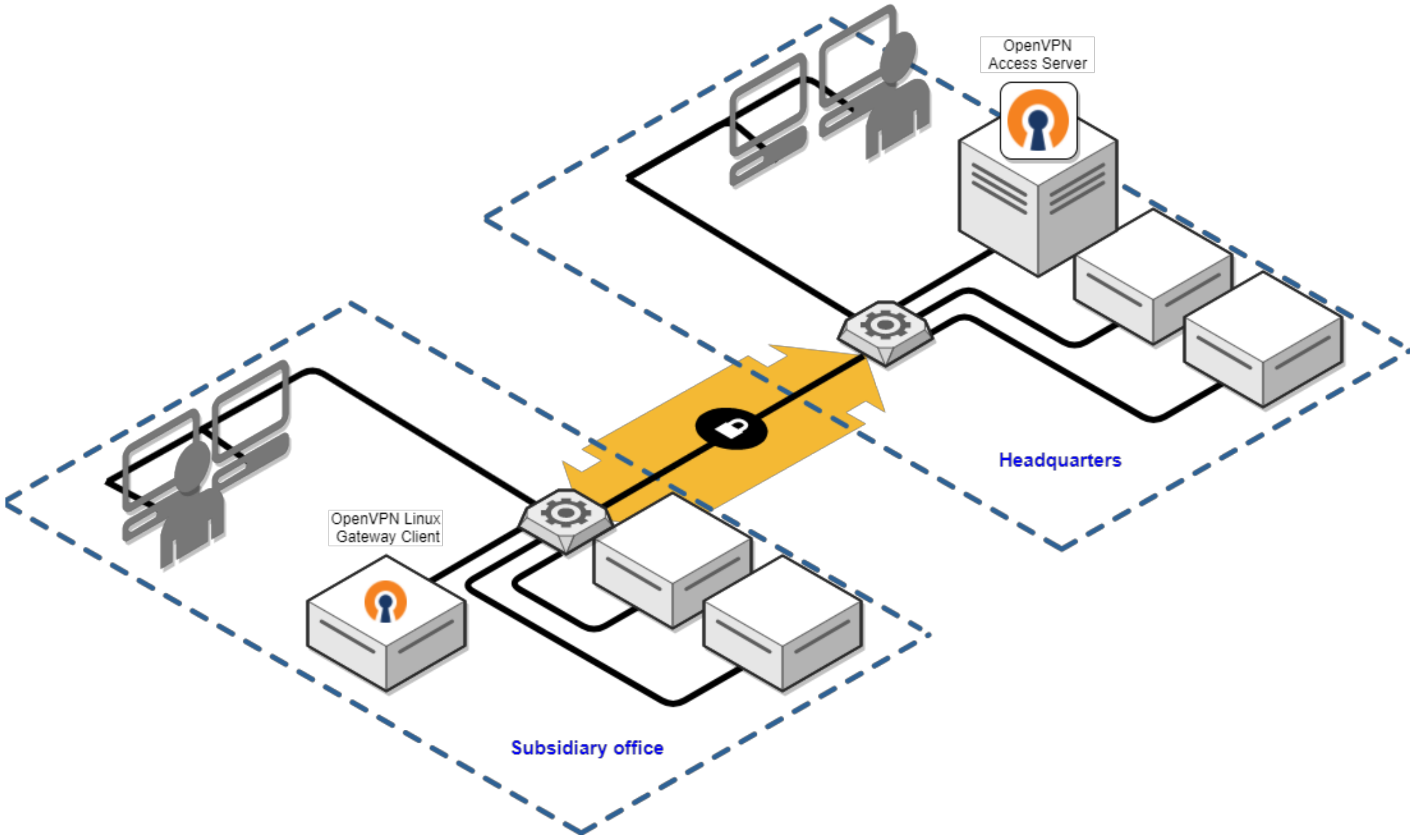
Virtual LAN (VLAN)

- VLANs are used whenever there is a need for traffic to be segregated at the link layer.
- For example, on Internet Protocol networks it is considered good practice to use a separate VLAN for each IP subnet. Reasons:
  - Preventing a machine assigned to one subnet from joining a different one by changing its IP address
  - Avoiding the need for hosts to process broadcast traffic originating from other subnets.
- Communication between two different VLANs is only possible through a router that has been connected to both VLANs.





Virtual Private Network (VPN)



Virtual Networks

- ❖ Virtual networks have been a part of the Internet for years.
  - ➢ VLAN, VPN, MPLS, VRF

- ➢ Each one of these virtualization primitives virtualize one aspect of network.
- ➢ Limited in scope.
- ➢ Traditionally configured by a box-by-box basis, with no single unifying abstraction that can be invoked in a more global manner.

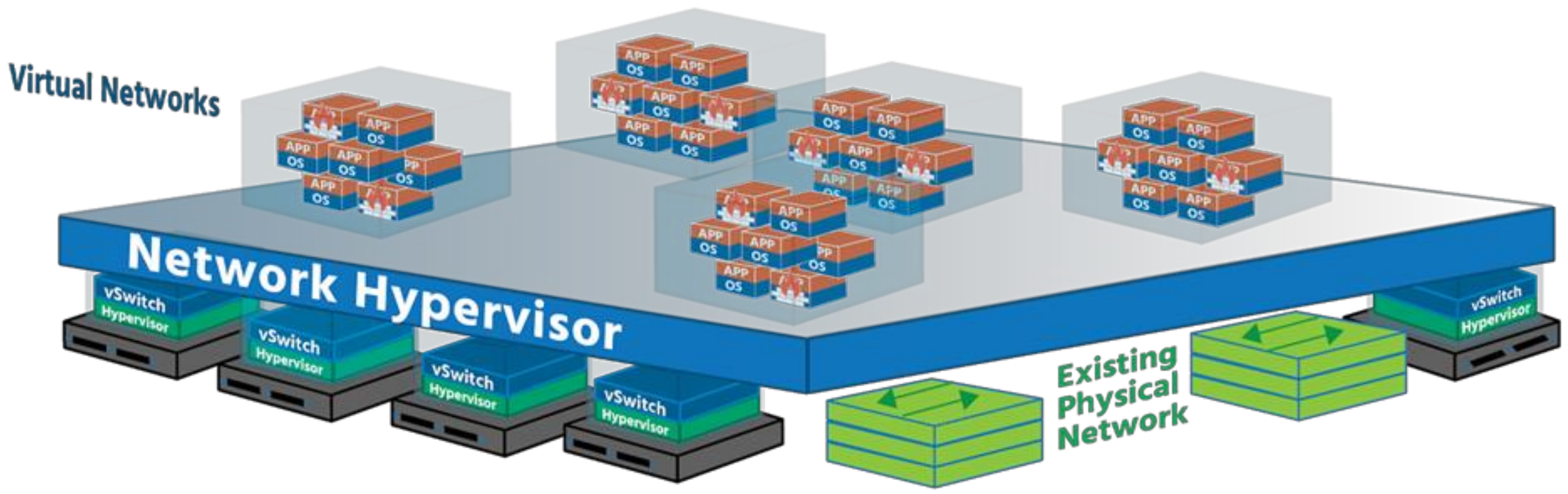
- ❖ There was no single technology or clean abstraction that virtualize the network as whole, automatically and programmatically.

SDN and Network Virtualization

- By separating the control plane from the data plane, and logically centralizing the control plane, it became possible to expose a single API entry point for the creation, modification, and deletion of virtual networks.

- This meant that the same **automation systems** that were being used to provision compute and storage capacity in a cloud (such as OpenStack) could now programmatically provision a virtual network with appropriate policies to interconnect those other resources.

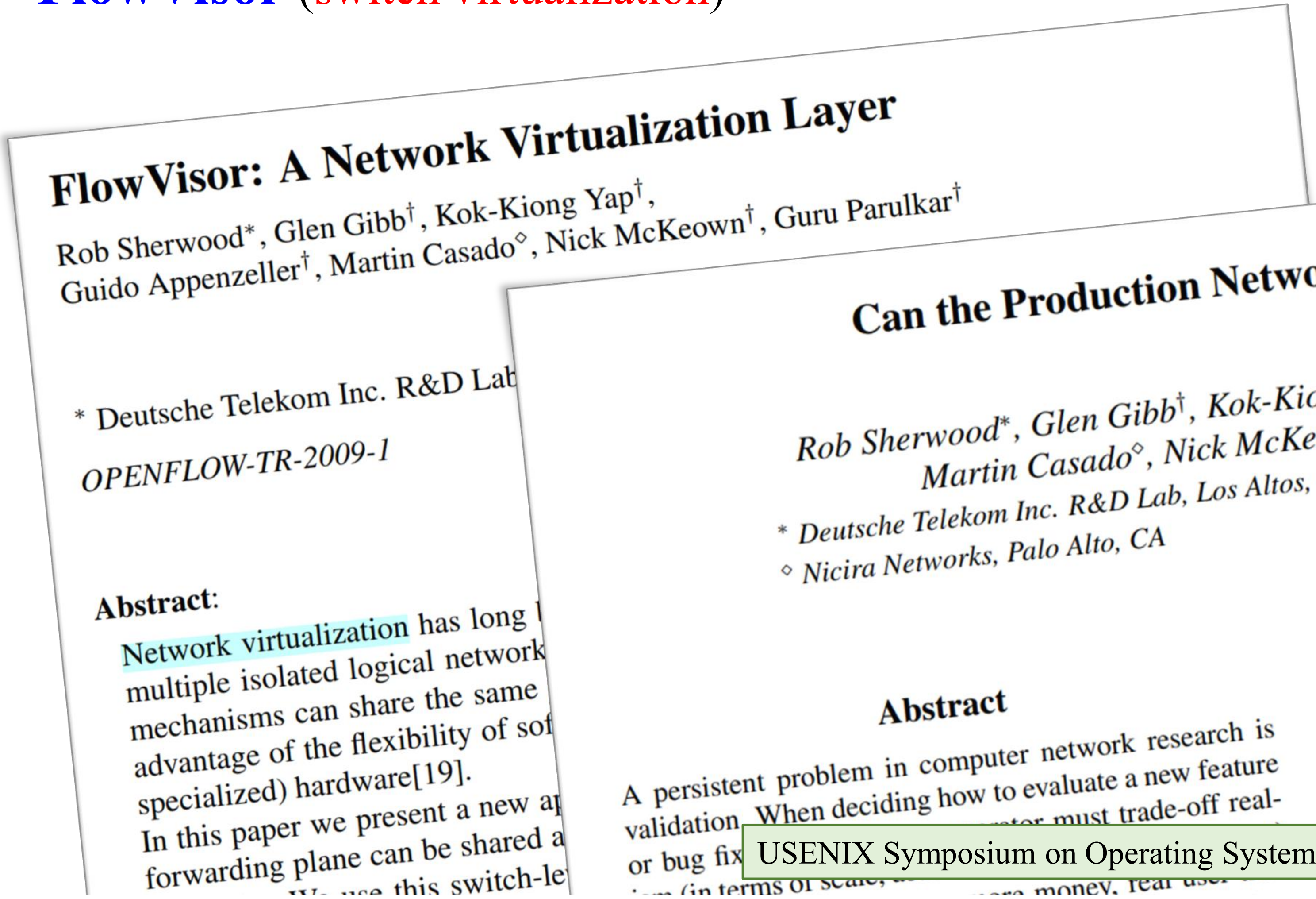
Network Virtualization



Multiple isolated logical networks each with potentially different topologies, addressing and forwarding mechanisms that share the same physical infrastructure.

The First Significant Work

FlowVisor (switch virtualization)



FlowVisor: A Network Virtualization Layer

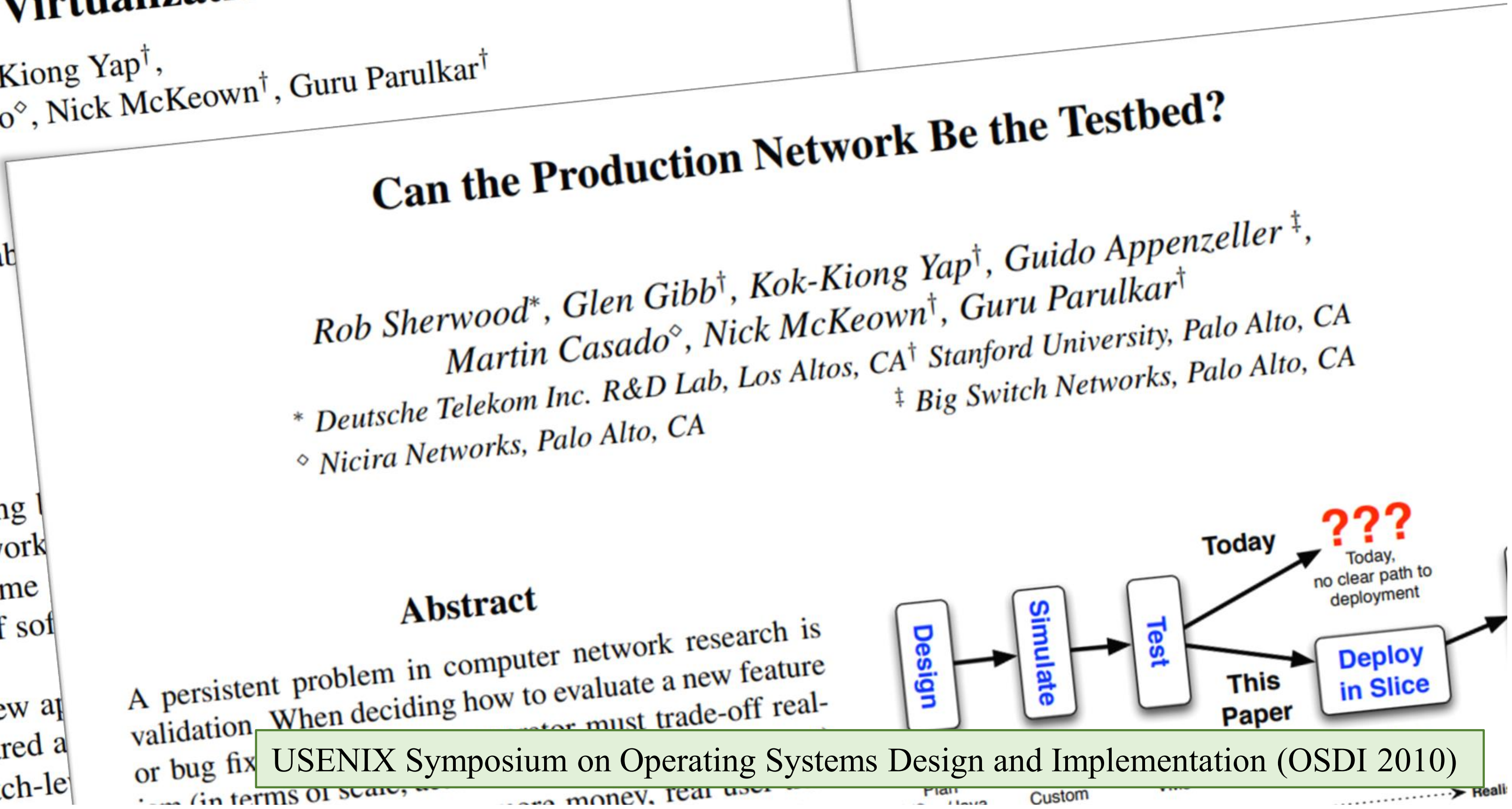
Rob Sherwood*, Glen Gibb†, Kok-Kiong Yap†, Guido Appenzeller†, Martin Casado◇, Nick McKeown†, Guru Parulkar†

* Deutsche Telekom Inc. R&D Lab

OPENFLOW-TR-2009-1

Abstract:

Network virtualization has long multiple isolated logical network mechanisms can share the same advantage of the flexibility of sof specialized) hardware[19]. In this paper we present a new ap forwarding plane can be shared a We use this switch-le



Can the Production Network Be the Testbed?

Rob Sherwood*, Glen Gibb†, Kok-Kiong Yap†, Guido Appenzeller‡, Martin Casado◇, Nick McKeown†, Guru Parulkar†

* Deutsche Telekom Inc. R&D Lab, Los Altos, CA † Stanford University, Palo Alto, CA ‡ Big Switch Networks, Palo Alto, CA ◇ Nicira Networks, Palo Alto, CA

Abstract

A persistent problem in computer network research is validation. When deciding how to evaluate a new feature or bug fix, a researcher must trade-off real- ism (in terms of scale, ... more money, real user ...

USENIX Symposium on Operating Systems Design and Implementation (OSDI 2010)

The Problem

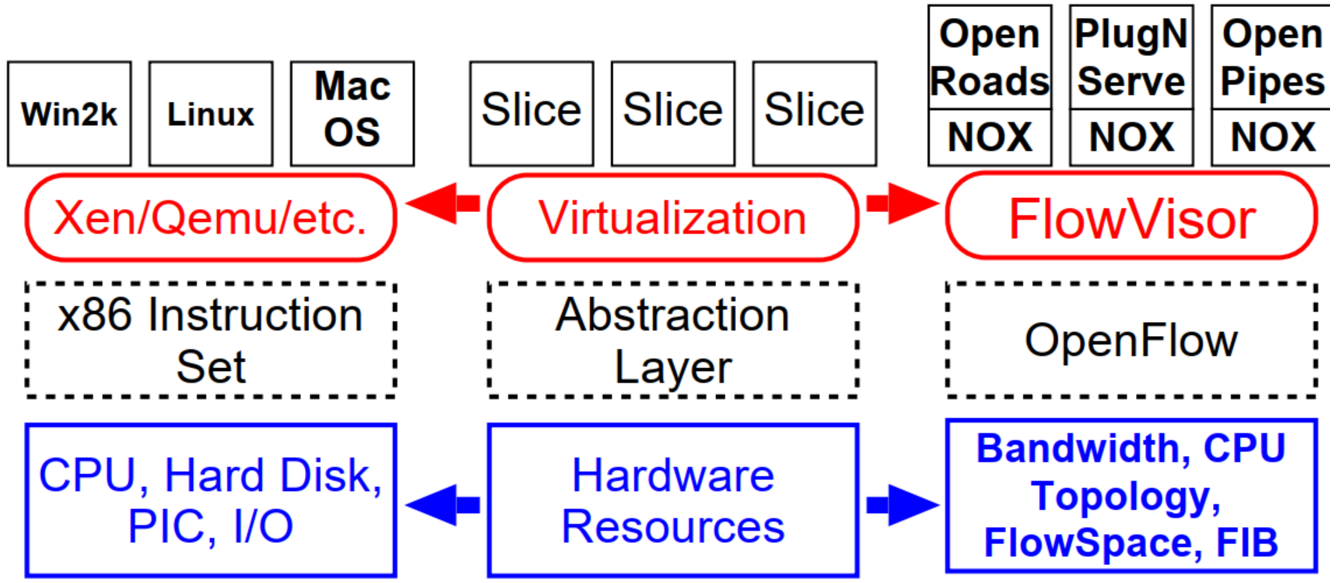
- Realistic evaluation and validation in computer network research
  - Real testbed
    - Real equipment
    - Scale
    - Actual user traffic
  - Trade-off between realism and cost
    - Real equipment requires vendor adoption
    - Larger scale costs more money
    - Real user traffic likely requires downtime
  - Goal: Realism
  - Ideally: Deploy in parallel with production without disrupting the production network

The Solution

❖ Virtualize the network and slice its resources

- The network itself should have a hardware abstraction layer.
- Multiple different networks can run simultaneously on top of the physical network without interfering with each other.
- Above the hardware abstraction layer, we want new protocols and addressing formats to run independently in their own isolated slice of the same physical network.
- Experimenters try out their ideas in an isolated slice.

The Solution



- **FlowVisor**: A network virtualization layer.
  - Based on SDN and OpenFlow
  - Slices the network hardware by placing a layer between the control plane and the data plane.

FlowVisor Design Goals

- The virtualization should be **transparent** to both data and control planes.
  - The control logic is unaware of the slicing layer.
  - Neither the OpenFlow switches nor the controllers need to be modified.

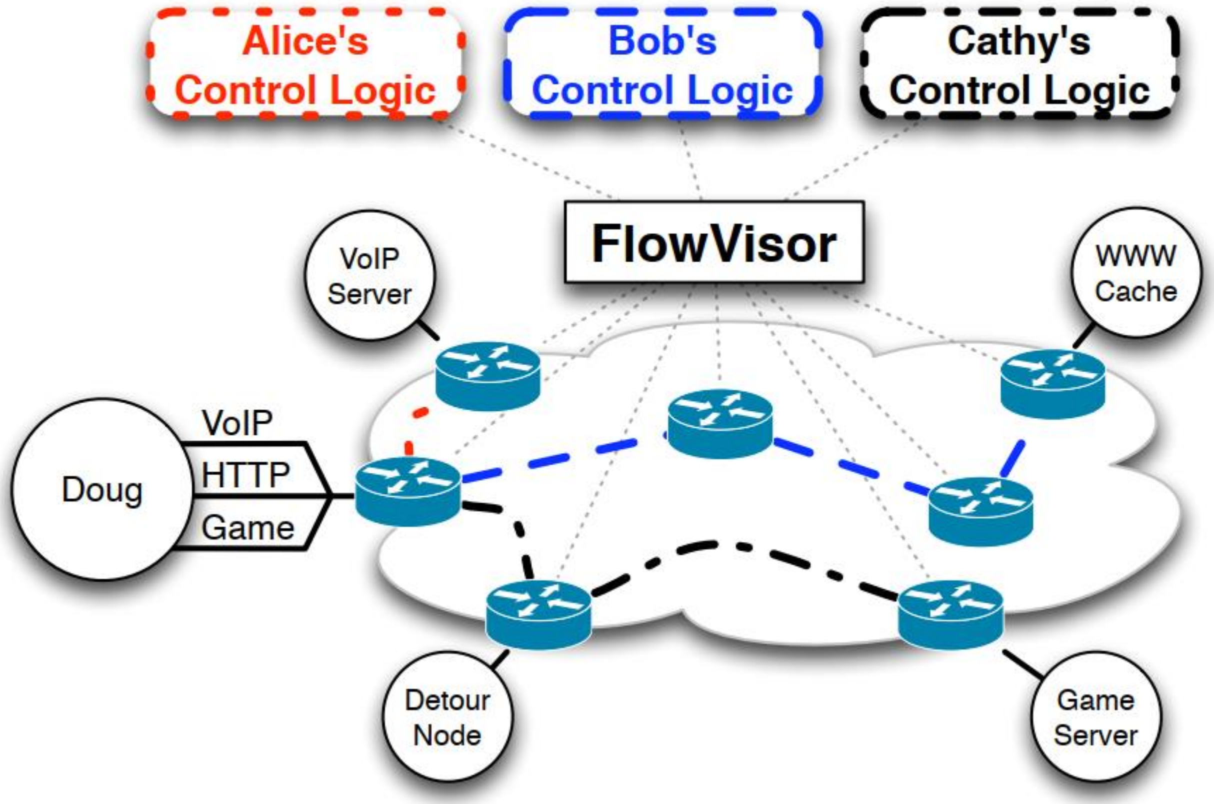
- There should be strong **isolation** between network slices.
  - Actions of one slice are prevented from affecting another.
  - FlowVisor blocks and rewrites control messages as they cross the slicing layer.

- The slice definition policy should be rich and extensible.

- Allowing real users to opt-in on a per-flow basis.
  - The policy language can map flows to slices

FlowVisor

- FlowVisor allows users to delegate control of subsets of their traffic to distinct researchers.
- Each research experiment runs in its own, isolated network slice.

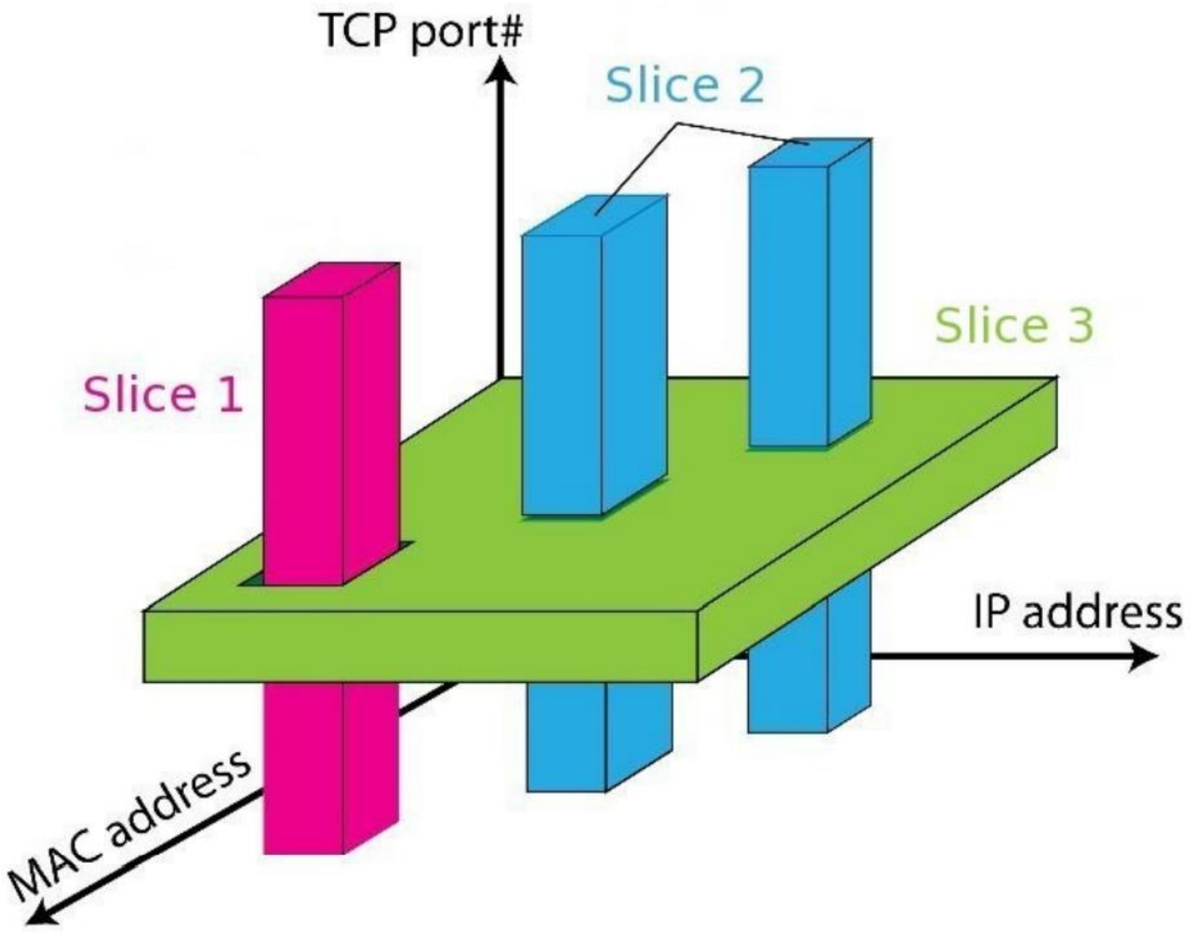


Slicing dimensions

- **Topology**. Each slice has its own view of network nodes (e.g., switches and routers) and the connectivity between them.
- **Bandwidth**. Each slice has its own fraction of bandwidth on each link.
- **Device CPU**. Each slice is limited to a fraction of each device’s CPU that it can consume.
- **Forwarding tables**. Each slice has a finite quota of forwarding rules.
- **FlowSpace**. Which packets does the slice control?

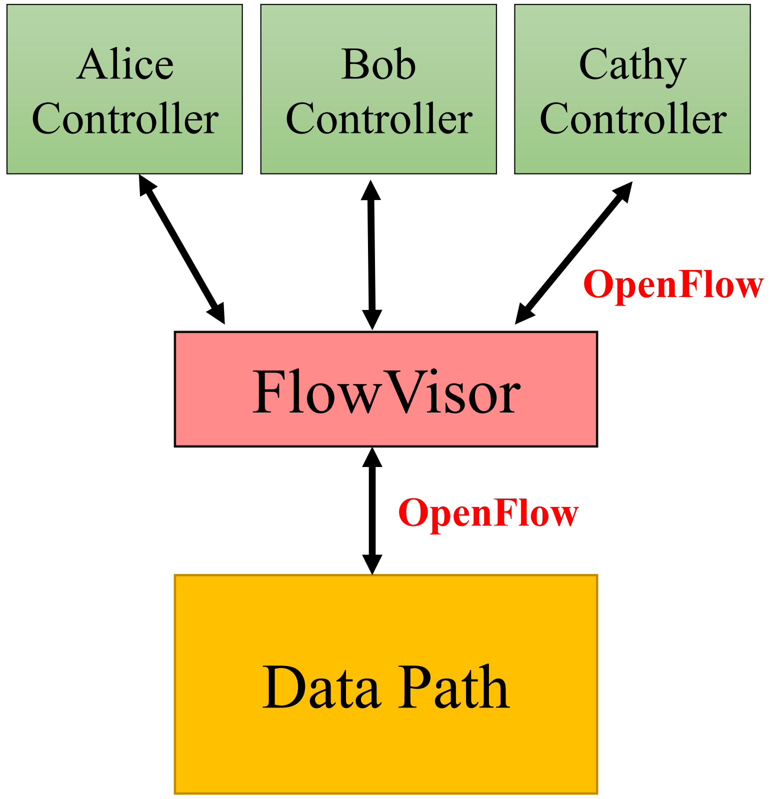
FlowSpace

❖ **FlowSpace**: Maps packets to slices



FlowVisor Design

- ❖ Network resources are sliced.
- ❖ Each slice has control over a set of flows, called its flowspace.
- ❖ Each slice has its own distinct, programmable control logic.
- ❖ FlowVisor acts as a **transparent proxy** between OpenFlow-enabled network devices and multiple OpenFlow slice controllers.
- ❖ All OpenFlow messages between the switch and the controller are sent through FlowVisor.
- ❖ FlowVisor uses the OpenFlow protocol to communicate upwards to the slice controllers and and downwards to OpenFlow switches.



FlowVisor Design

- FlowVisor enforces transparency and isolation between slices by inspecting, rewriting, and policing OpenFlow messages as they pass.
- Depending on the resource allocation policy, message type, destination, and content, the FlowVisor will forward a given message unchanged, translate it to a suitable message and forward, or bounce the message back to its sender in the form of an OpenFlow error message.
- For a message sent from slice controller to switch, FlowVisor ensures that the message acts only on traffic within the resources assigned to the slice.
- For a message sent from switch to controller, FlowVisor examines the message content to infer the corresponding slice(s) to which the message should be forwarded.

Messages to Controller

- FlowVisor only sends control plane messages to a slice controller if the source switch is actually in the slice's topology.
- FlowVisor rewrites OpenFlow feature negotiation messages so that the slice controller only sees the physical switch ports that appear in the slice.
- OpenFlow port up/port down messages are similarly pruned and only forwarded to the affected slices.

Messages to Forwarding Plane

- The most important messages to the forwarding plane are insertions and deletions to the forwarding table.
- FlowVisor rewrites both the flow definition and the set of actions so that they do not violate the slice's definition.
- Given a forwarding rule modification, the FlowVisor rewrites the flow definition to intersect with the slice's flowspace.

An Example

- A researcher, Bob, wants to create a new HTTP load-balancer to spread port 80 traffic over multiple web servers.
- He requests a network slice from Alice, his network administrator.
- He requests a slice: its topology should encompass the web servers, and its flowspace should include all flows with port 80.
- He is allocated a control plane where he adds his load balancing logic to control how flows are routed in the data plane.
- He may advertise his new service so as to attract users.
- Interested users “opt-in” by contacting their network administrator to add a subset of their flows to the flowspace of Bob’s slice.

An Example (cont.)

- For example, Bob’s flowspace gives him control over HTTP traffic for the set of users—e.g., users Doug and Eric.

- If Bob’s slice controller tried to create a rule that affected all of Doug’s traffic (HTTP and non-HTTP), then the FlowVisor would rewrite the rule to only affect the intersection, i.e., only Doug’s HTTP traffic.

- If the intersection between the desired rule and the slice definition is null, e.g., Bob tried to affect traffic outside of his slice, e.g.., Doug’s non-HTTP traffic, then the FlowVisor would drop the control message and return an error to Bob’s controller.

- If Bob tried to affect all traffic in the system in a single rule, the FlowVisor would transparently expand the single rule in to two rules: one for each of Doug’s and Eric’s HTTP traffic

Messages to Forwarding Plane (cont.)

- ❖ FlowVisor also rewrites the lists of actions in a forwarding rule.
  - ❖ For example, if Bob creates a rule to send out all ports, the rule is rewritten to send to just the subset of ports in Bob's slice.
  - ❖ If Bob tries to send out a port that is not in his slice, FlowVisor returns an "action is invalid" error.

Bandwidth Isolation

- The FlowVisor creates a per-slice queue on each port on the switch.
- The queue is configured for a fraction of link bandwidth, as defined in the slice definition.
- To enforce bandwidth isolation, FlowVisor rewrites all slice forwarding table additions from "*send out port X*" to "*send out queue Y on port X*", where *Y* is a slice-specific queue ID.

Device CPU Isolation

❖ Main sources of load on a switch CPU:

- Generating new flow messages
  - FlowVisor tracks the new flow message arrival rate for each slice, and if it exceeds some threshold, the FlowVisor inserts a forwarding rule to drop the offending packets for a short period (1 second).
- Handling requests from controller
  - For each slice, the FlowVisor limits CPU consumption by throttling the OpenFlow message rate to a maximum rate per second.

Flow Entry Isolation

- FlowVisor counts the number of flow entries used per slice and ensures that each slice does not exceed a preset limit.

- When a guest controller exceeds its flow entry limit, any new rule insertions receives a “table full” error message.

Network Virtualization

Recap the definition of network virtualization:

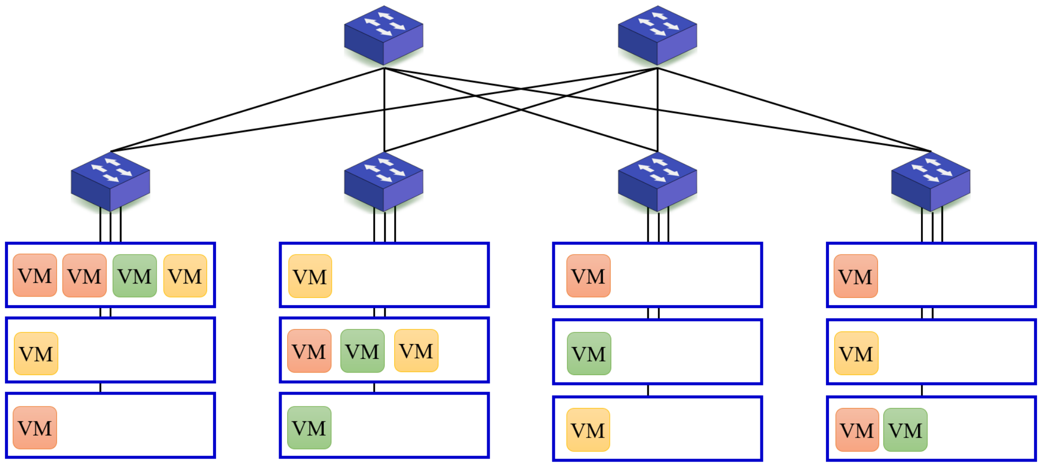
- "Multiple isolated logical networks each with potentially different topologies, addressing and forwarding mechanisms that share the same physical infrastructure."

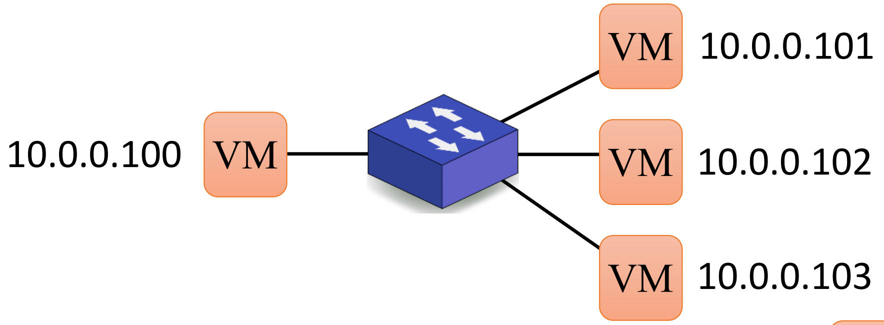
Does FlowVisor conform to the definition?

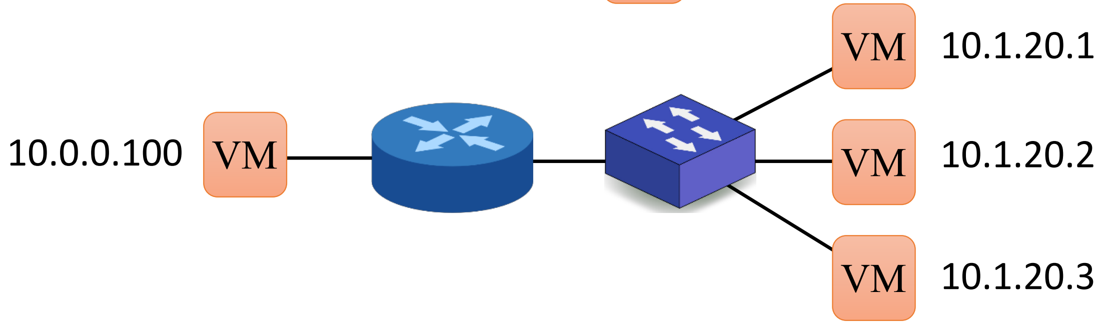
Network Virtualization in Multi-tenant Data Centers

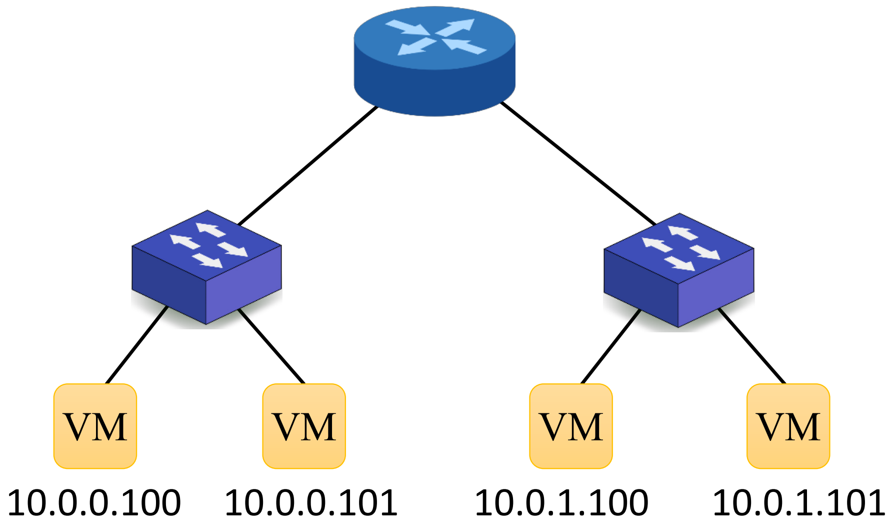
Imagine a multi-tenant data center.

- Creation of isolated virtual networks for each tenant
- Dynamic workload placement
- Workloads require different topologies
- Address overlap







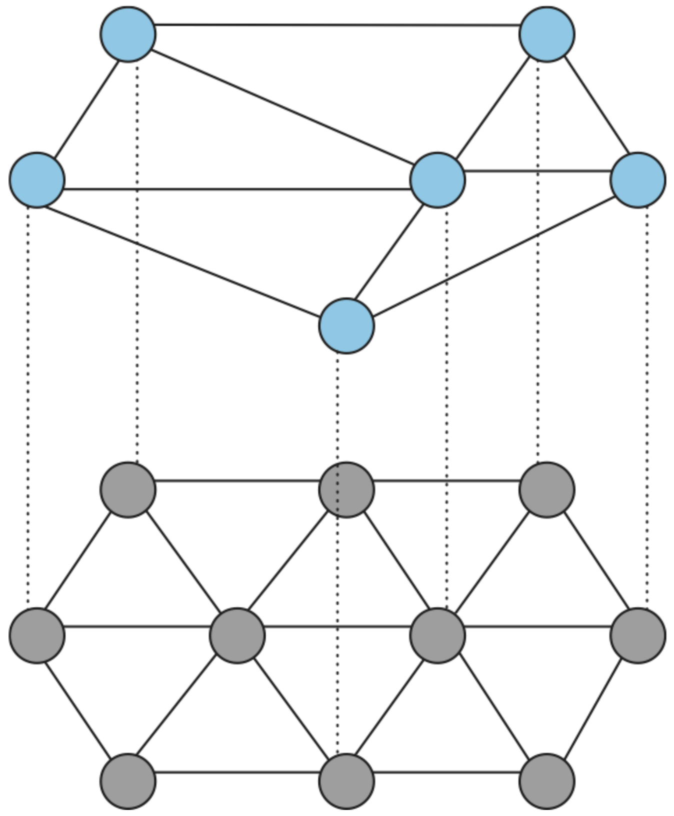


Solution

Network Virtualization Using
Overlay Networks
Implemented by Tunneling Protocols

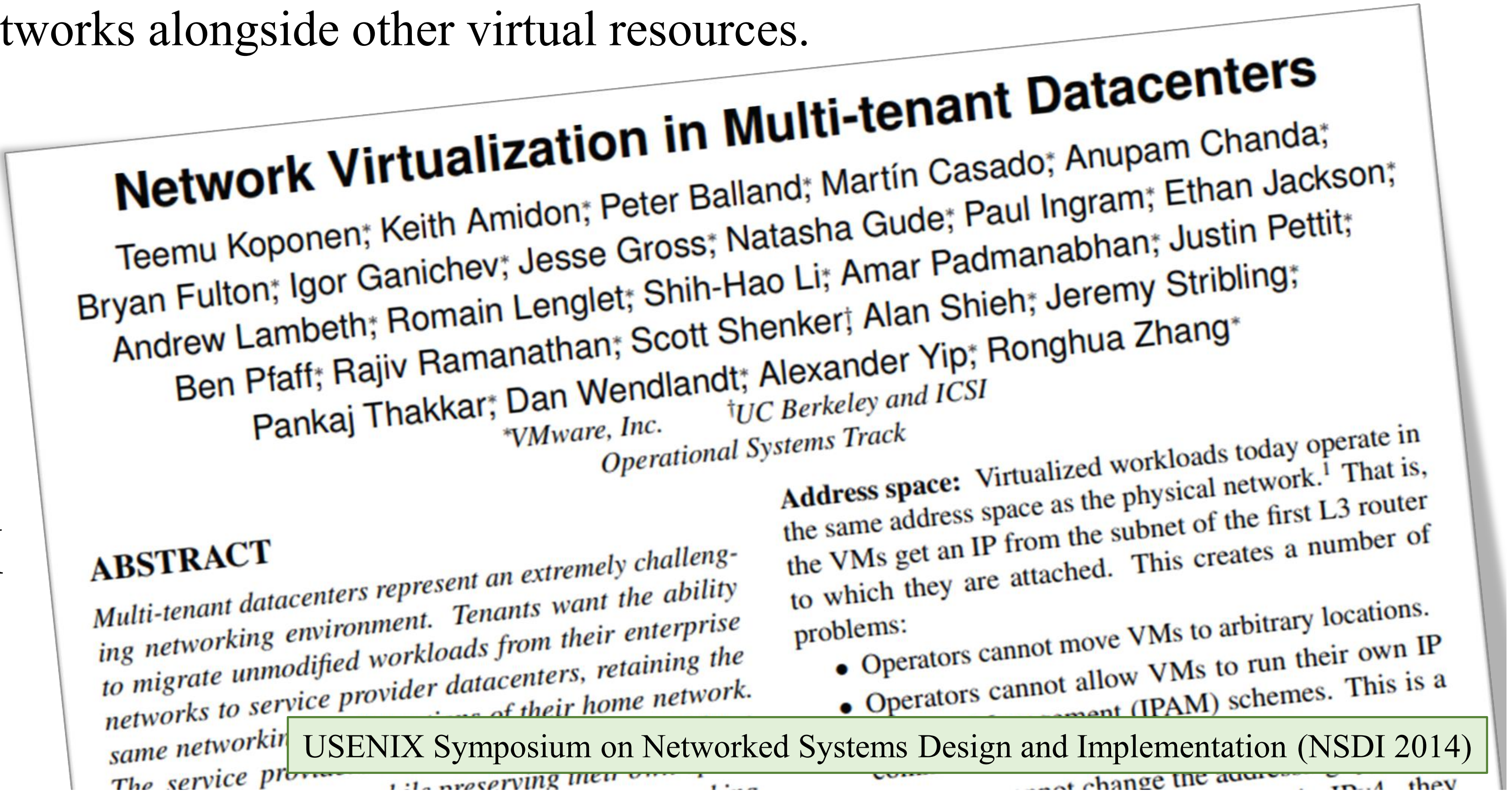
Overlay Network

- A logical network implemented on top of some underlying network
  - Each node in the overlay also exists in the underlying network
  - The links that connect the overlay nodes are implemented as tunnels through the underlying network.



Network Virtualization by Overlay Networks

- ❖ Overlay networks build the foundation of virtual networks and they run as independent virtual networks on top of a physical network infrastructure.
- ❖ Virtual network overlays allow resource providers, such as cloud providers, to provision and orchestrate networks alongside other virtual resources.



Network Virtualization in Multi-tenant Datacenters

Teemu Koponen[*], Keith Amidon[*], Peter Balland[*], Martín Casado[*], Anupam Chanda[*], Bryan Fulton[*], Igor Ganichev[*], Jesse Gross[*], Natasha Gude[*], Paul Ingram[*], Ethan Jackson[*], Andrew Lambeth[*], Romain Lenglet[*], Shih-Hao Li[*], Amar Padmanabhan[*], Justin Pettit[*], Ben Pfaff[*], Rajiv Ramanathan[*], Scott Shenker[†], Alan Shieh[*], Jeremy Stribling[*], Pankaj Thakkar[*], Dan Wendlandt[*], Alexander Yip[*], Ronghua Zhang[*]

[*]*VMware, Inc.* [†]*UC Berkeley and ICSI*

Operational Systems Track

ABSTRACT

Multi-tenant datacenters represent an extremely challenging networking environment. Tenants want the ability to migrate unmodified workloads from their enterprise networks to service provider datacenters, retaining the same networkin ... of their home network. The service pro ... hile preserving their own

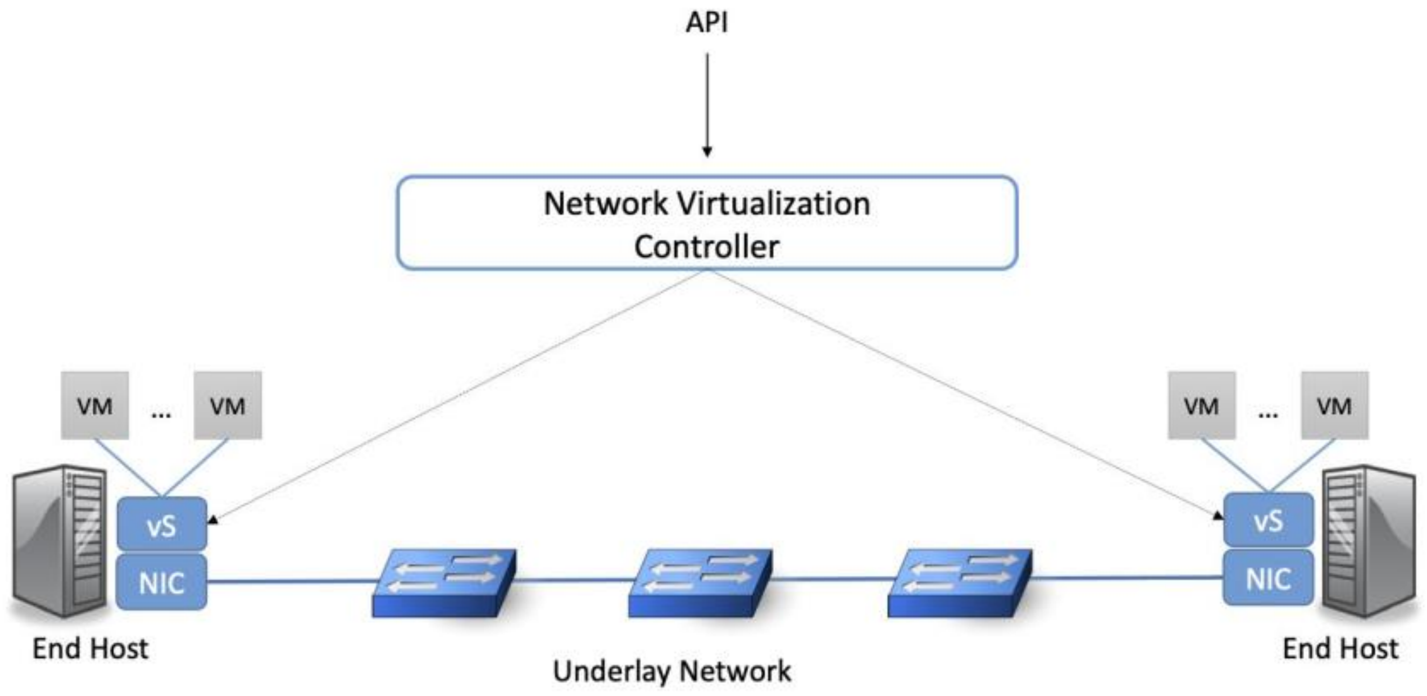
Address space: Virtualized workloads today operate in the same address space as the physical network.[1] That is, the VMs get an IP from the subnet of the first L3 router to which they are attached. This creates a number of problems:

- Operators cannot move VMs to arbitrary locations.
- Operators cannot allow VMs to run their own IP ... ment (IPAM) schemes. This is a ... cannot change the addre ... IPv4 ... they

USENIX Symposium on Networked Systems Design and Implementation (NSDI 2014)

- ➢ Nicira NVP
- ➢ VMware NSX
- ➢ OVN

A Basic Network Virtualization System

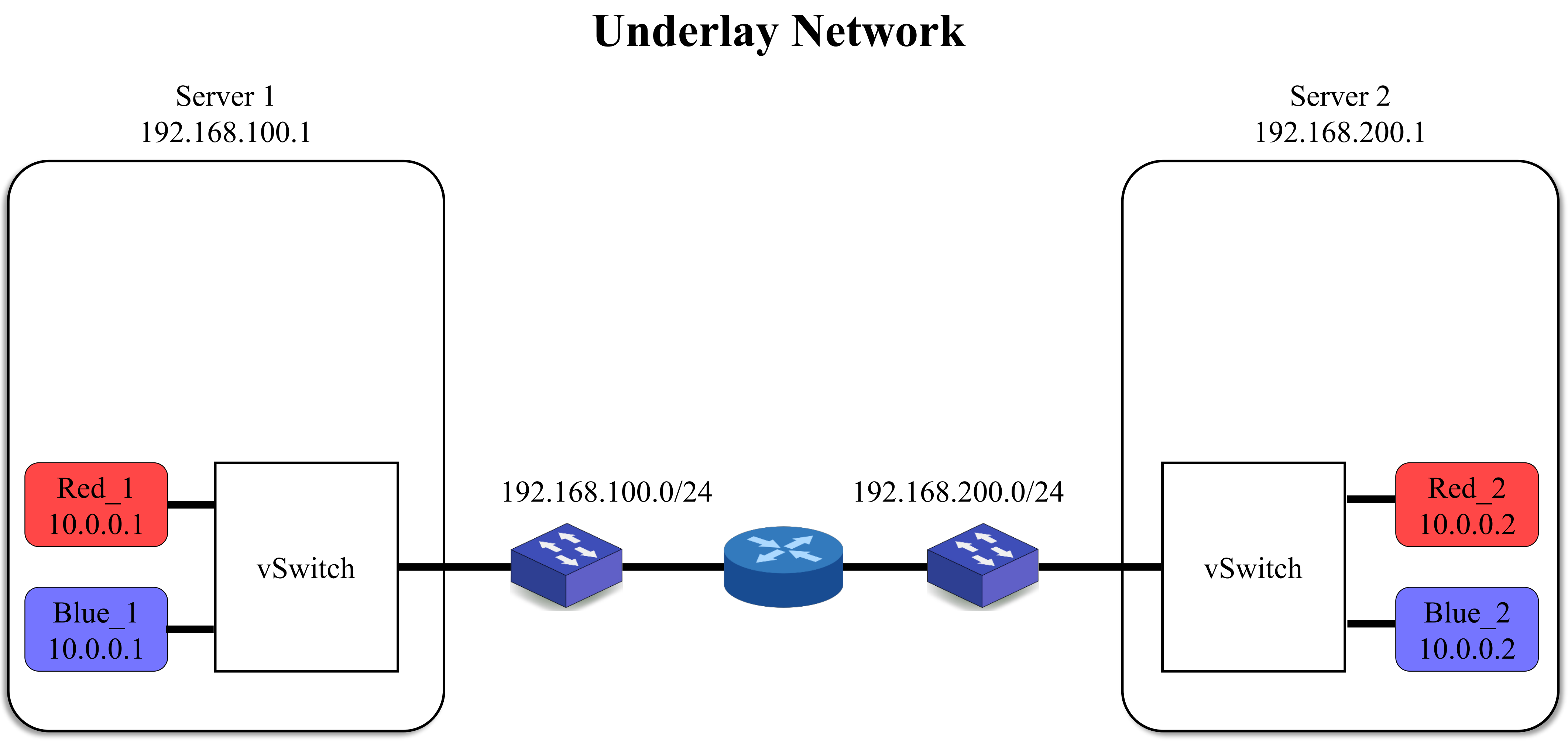


- For example, an API request could specify "*VM1 and VM2 should reside on the same virtual layer 2 subnet, network X*"
- The controller determines where those virtual machines are located, and then sends control commands to the appropriate virtual switches to create the virtual network abstraction that is required.

A Basic Network Virtualization System

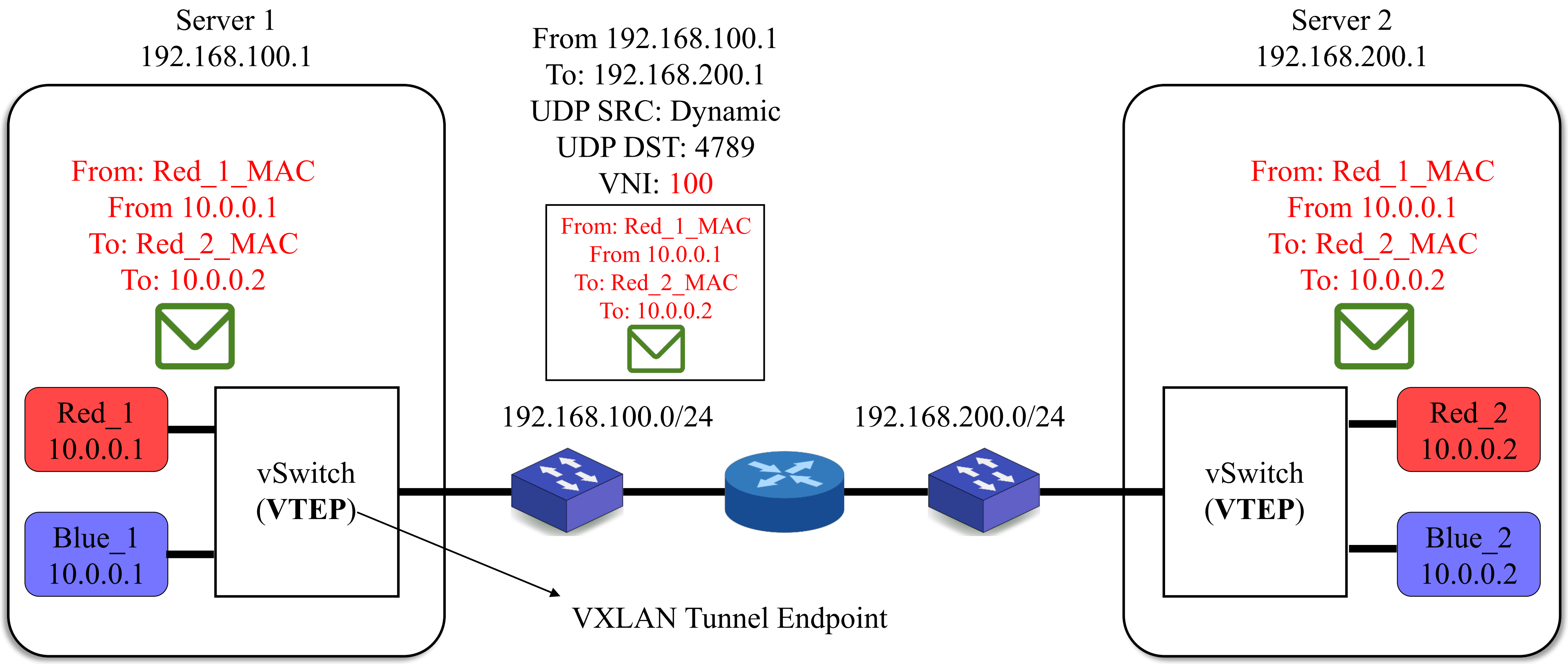
- The IP addresses of VMs need to be independent of the physical network topology.
- For this reason, network virtualization systems invariably make use of an overlay encapsulation such as VXLAN or NVGRE.
- Encapsulation decouples the address space of the virtual network from that of the physical network.

A Basic Network Virtualization System



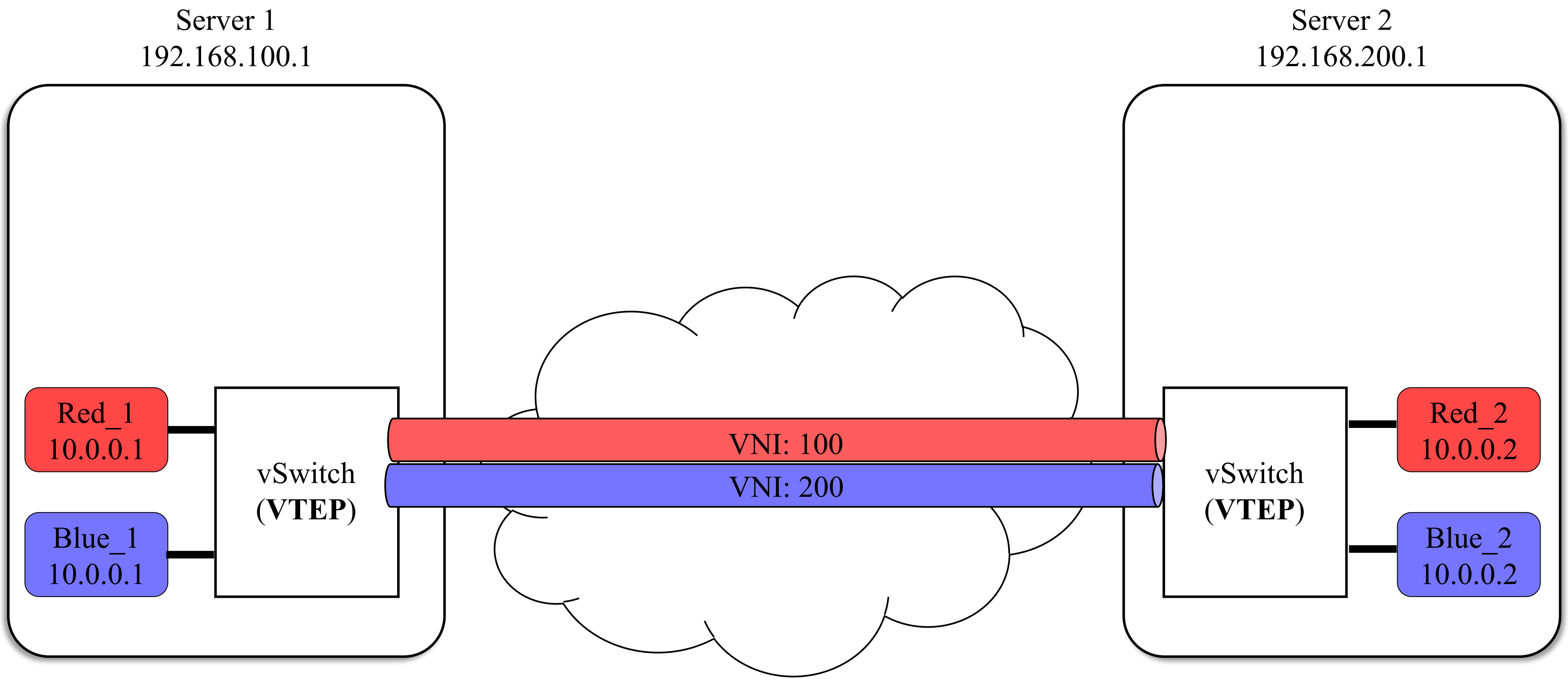
A Basic Network Virtualization System

VXLAN Encapsulation

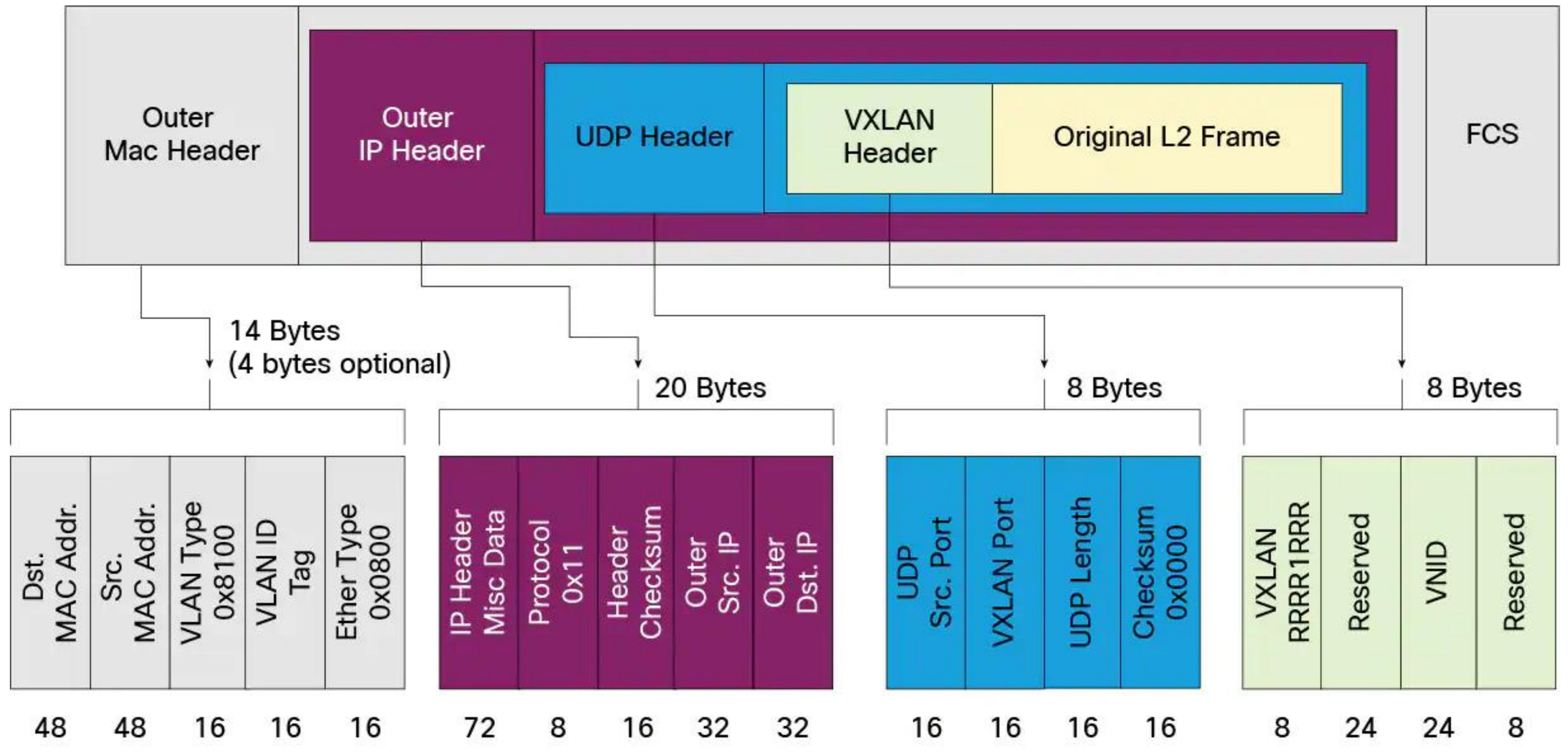


A Basic Network Virtualization System

VXLAN Tunnels



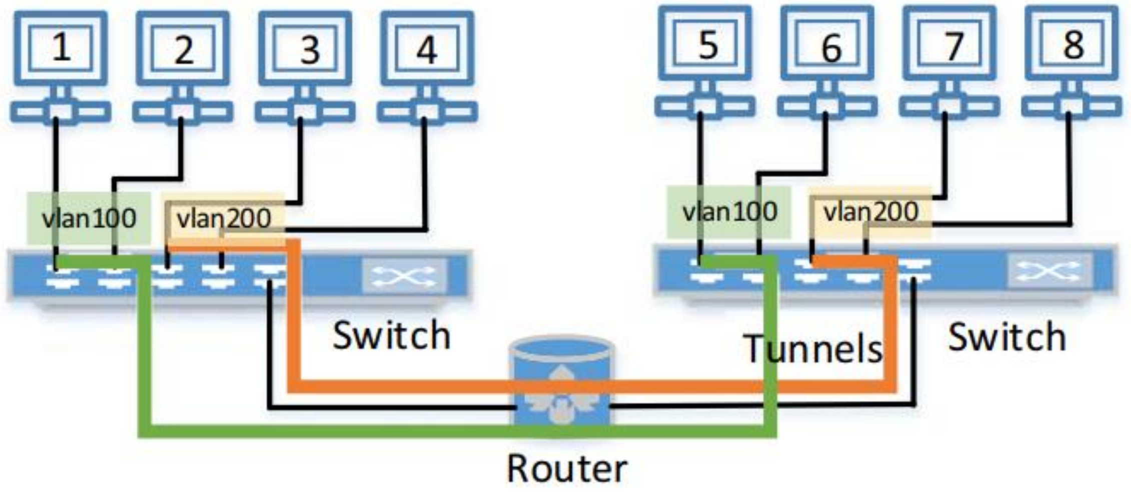
VXLAN Packet Format



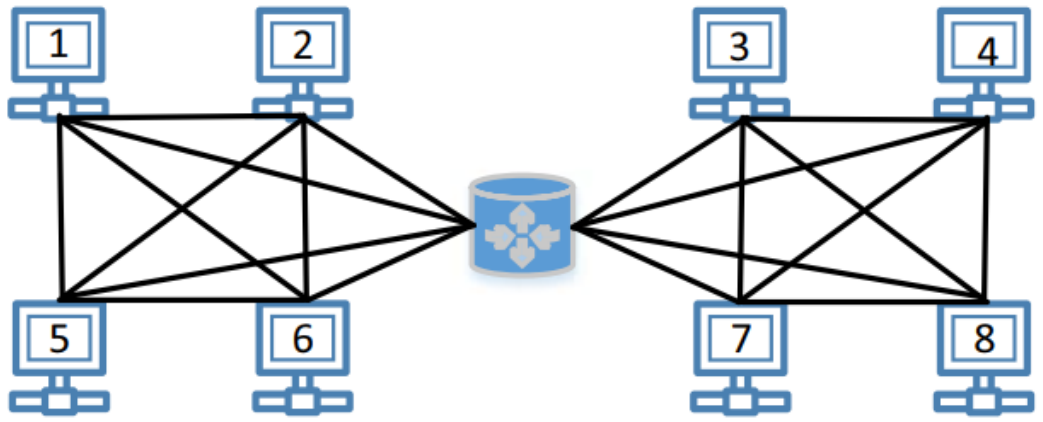
Other tunneling protocols: NVGRE, GENEVE

An Example of Virtual Networking by Tunneling

Physical Network with Layer-2 Tunnels (e.g., GRE/VxLAN)



Logical Network (Link-layer View)



VXLAN and VTEP

- Why VXLAN uses UDP instead of TCP?

- The effect of tunneling on the TCP/IP offloading resources (Large segment offload, TCP segmentation offload)?