

A tropical beach scene with palm trees, a bird of paradise flower in the foreground, and a blue sky with white clouds. The text is overlaid on the image.

# **VIRTUAL LABS ON SDN, OPEN VIRTUAL SWITCHES (OVS), CYBERSECURITY, AND OTHERS**

Jorge Crichigno

University of South Carolina



# Welcome!

I am a faculty member at the University of South Carolina (UofSC)  
This session will review labs for NETLAB developed at UofSC using open-source technology

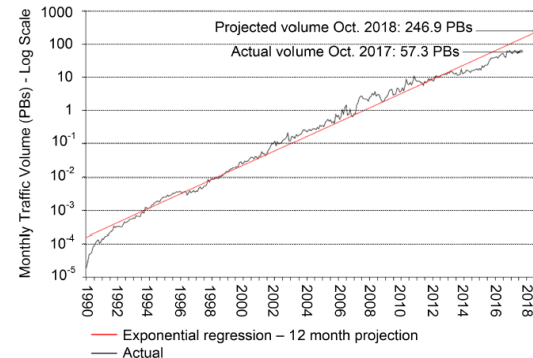
# Agenda

- Motivation, NETLAB environment
- Software-Defined Networking (SDN) labs
- Open Virtual Switch (OvS) labs
- Zeek Intrusion Detection Systems labs

# Motivation

- Science, engineering, mobile applications are generating data at an unprecedented rate
- From large facilities to portable devices, instruments can produce hundreds of terabytes in short periods of time
- Data must be typically transferred across high-throughput high-latency Wide Area Networks (WANs)
- The Energy Science Network (ESnet) is the backbone connecting U.S. national laboratories and research centers

Applications



ESnet traffic

# Motivation

---

- A biology experiment using the U.S. National Energy Research Scientific Computing Center (NERSC) resources

**SnapChat Data  
produced per day  
worldwide by millions  
of people  
= 38 TB**

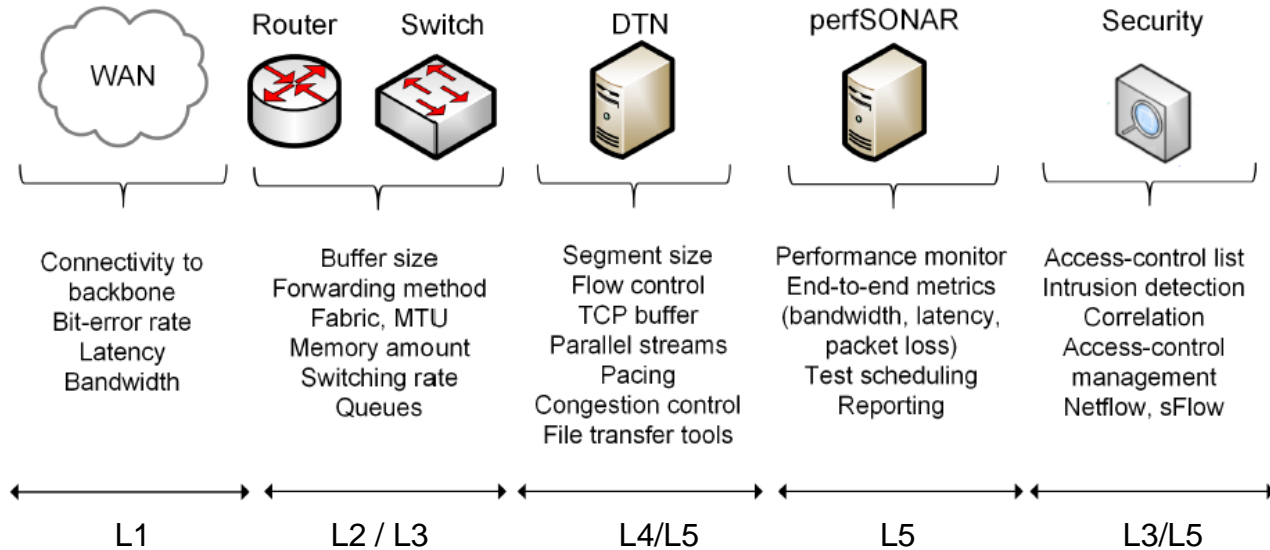
**One Biology experiment by  
a team of nine scientists:**

**= 114 TB**

**(Photosystem II X-Ray  
Study)**

# Motivation

There are features in network devices that are important for high performance



# Motivation

How can we teach these topics using a scalable environment?

Requirements

- High performance; speeds of ~50 Gbps
- Scalability; platform must be capable of cloning pods and expand capacity easily
- Real protocol stack, no simulation
- Free tools

# Motivation

## Partnering with the Network Development Group (NDG)

Feature	Private Cloud	Public Cloud
<b>Granularity to allocate physical resources</b>	Very granular	Not granular (access to the physical resources requires additional fees)
<b>Easy to create custom pods</b>	Easy	More difficult; hard to design complex topologies
<b>Cost</b>	Cost effective when used extensively	Cost effective for individual / small virtual machines; costly for large virtual machines over time
<b>Application layer for pedagogy, presentation of virtual scenarios</b>	Very flexible	Not flexible; limited to providers' interface, e.g., command-line interface
<b>Time-sharing compute resources</b>	The owner controls who can access resources. Easy to implement time-sharing policies	Cloud provider controls who can access resources (typically, a fee is required per user accessing resources)



# Environment: Mininet

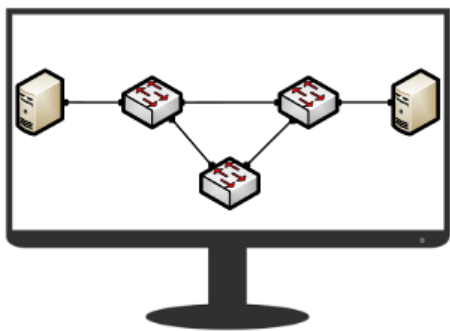
# Mininet

---

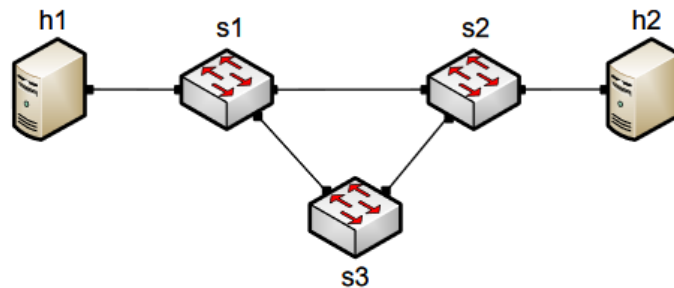
- Mininet is a virtual testbed for developing and testing network tools and protocols
- It creates a realistic virtual network on any type of machine (VM, cloud-hosted, or native)
- Inexpensive solution running in line with production networks
- Mininet offers the following features
  - Fast prototyping for new networking protocols
  - Simplified testing for complex topologies without the need of buying expensive hardware
  - Realistic execution as it runs real code on the Unix and Linux kernels
  - Open-source environment

# Mininet

- Mininet provides network *emulation* opposed to simulation, allowing all network software at any layer to be simply run as is
- Mininet's logical nodes can be connected into networks
- Nodes are sometimes called containers, or more accurately, *network namespaces*
- Containers consume sufficiently few resources that networks of over a thousand nodes have been created, running on a single laptop



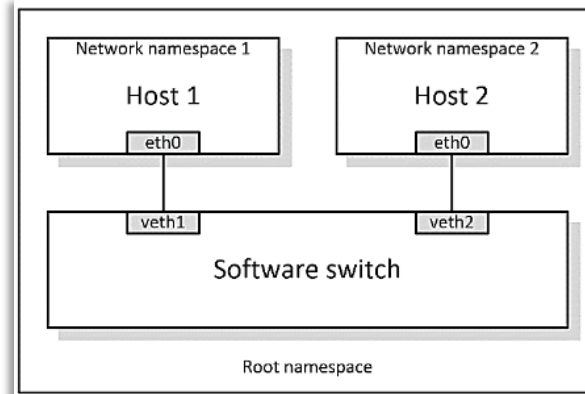
**Mininet Emulated Network**



**Hardware Network**

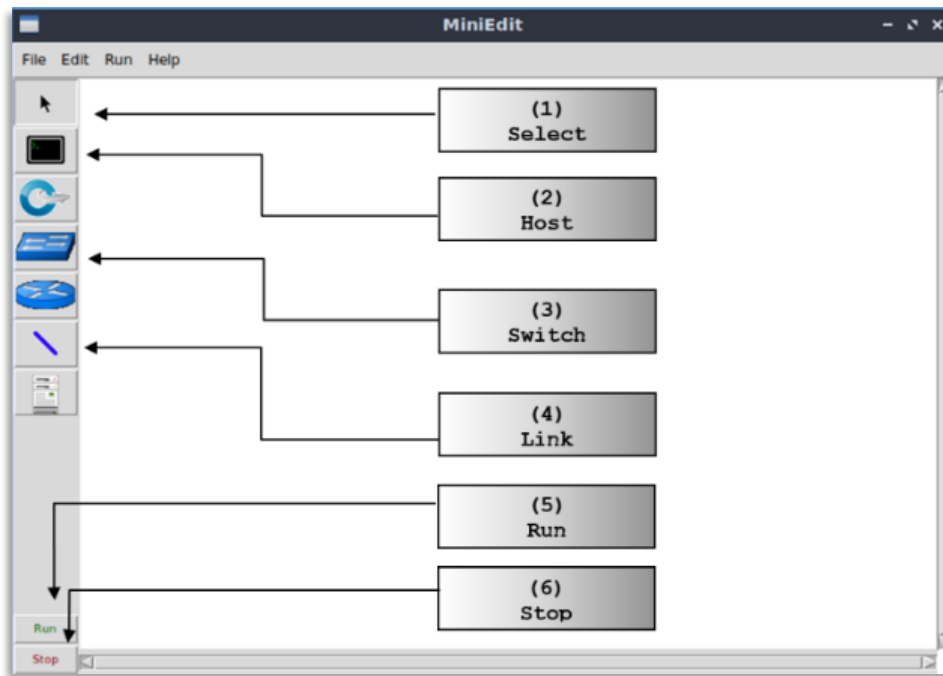
# Mininet Nodes

- A Mininet container is a process (or group of processes) that no longer has access to all the host system's native network interfaces
- Containers are then assigned virtual Ethernet interfaces, which are connected to other containers through a virtual switch
- Mininet connects a host and a switch using a virtual Ethernet (veth) link
- The veth link is analogous to a wire connecting two virtual interfaces



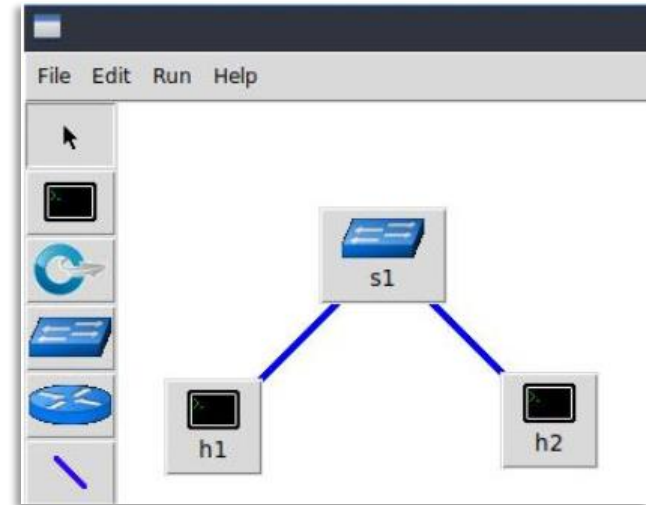
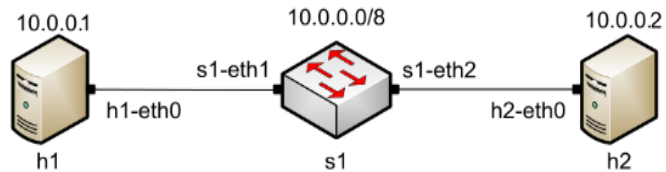
# MiniEdit

- MiniEdit is a simple GUI network editor for Mininet



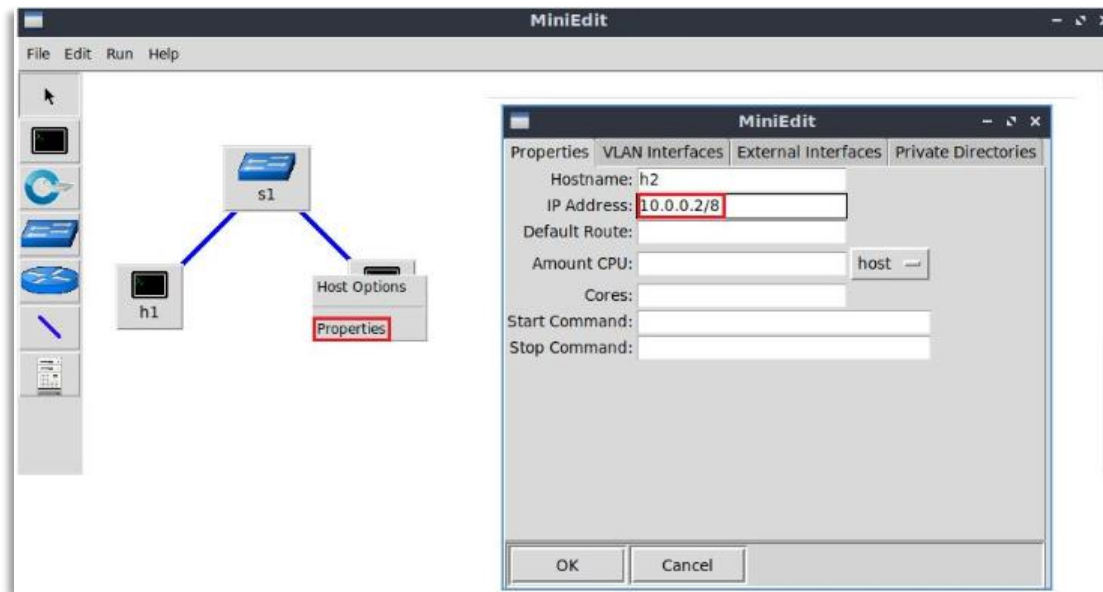
# MiniEdit

- To build Mininet's minimal topology, two hosts and one switch must be deployed



# Host Configuration

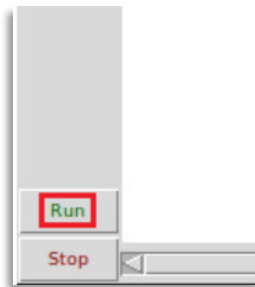
- Configure the IP addresses at host h1 and host h2
- A host can be configured by holding the right click and selecting properties on the device



# Starting Emulation

---

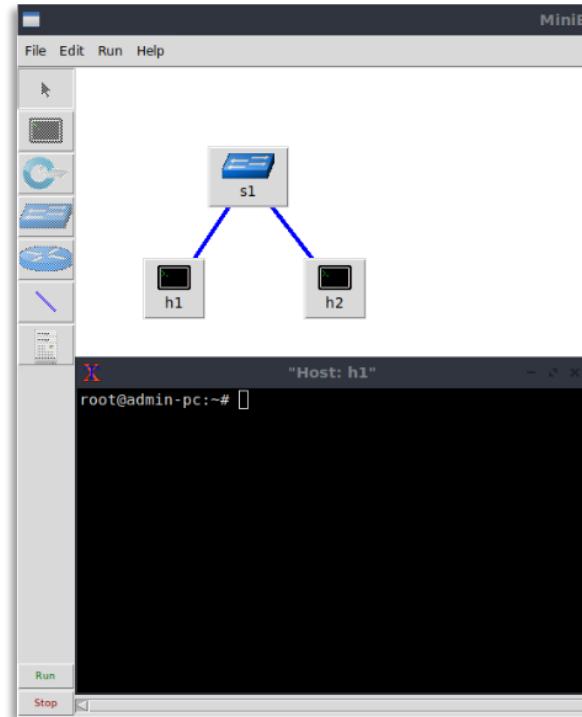
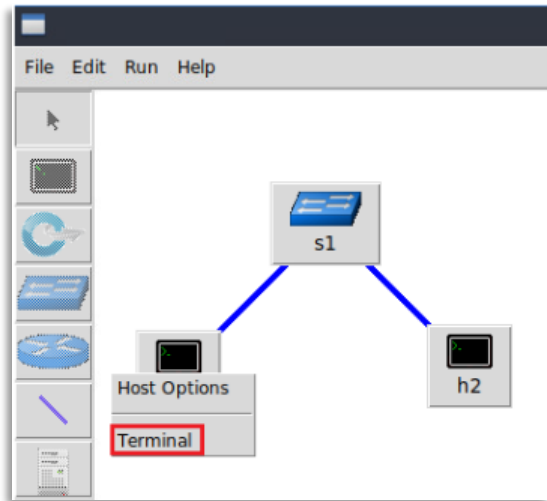
- Before testing the connection between host h1 and host h2, the emulation must be started
- Click on the Run button to start the emulation
- The emulation will start and the buttons of the MiniEdit panel will gray out, indicating that they are currently disabled





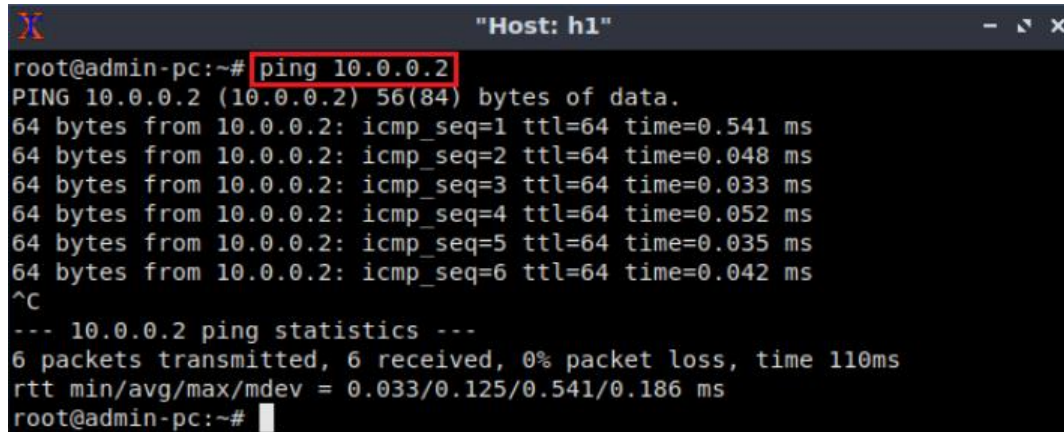
# Executing Commands on Hosts

- Open a terminal on host by holding the right click and selecting *Terminal*



# Testing Connectivity

- On host h1's terminal, type the command `ping 10.0.0.2`



```
Host: h1
root@admin-pc:~# ping 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.541 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.048 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.033 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.052 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.035 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.042 ms
^C
--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 110ms
rtt min/avg/max/mdev = 0.033/0.125/0.541/0.186 ms
root@admin-pc:~#
```

# Overview SDN Lab Series

# SDN Lab Series

---

The labs provide learning experiences on essential SDN topics

- Mininet
- Legacy networks, Border Gateway Protocol (BGP)
- FRR routing, an open routing implementation
- MPLS networks – early efforts toward SDN
- SDN fundamentals – controllers, switches
  - ONOS controller
  - Open Virtual Switch (OVS)
- Traffic isolation with VXLAN
- OpenFlow
- Interconnection between SDN and legacy Networks

# SDN Lab Series

---

## Lab experiments

Lab 1: Introduction to Mininet

Lab 2: Legacy Networks: BGP Example as a distributed system and autonomous forwarding decisions

Lab 3: Early efforts of SDN: MPLS example of a control plane that establishes semi-static forwarding paths

Lab 4: Introduction to SDN

Lab 5: Configuring VXLAN to provide network traffic isolation

Lab 6: Introduction to OpenFlow

Lab 7: SDN-routing within an SDN network

Lab 8: Interconnection between legacy networks and SDN networks

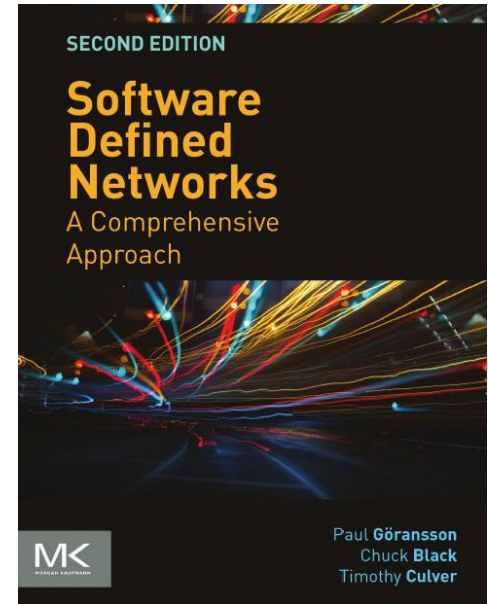
Lab 9: Configuring Virtual Private LAN Services (VPLS) with SDN networks

Lab 10: Applying Equal-Cost Multi-Path (ECMP) within SDN networks

# SDN Lab Series

---

- The goal of the SDN Lab Series is to provide a practical experience to students and IT practitioners
- The labs provide background information which is reinforced with hands-on activities
- A good book on SDN network (which matches the SDN Lab Series) is “Software Defined Networking, A Comprehensive Approach”



# Organization of Lab Manuals

---

Each lab starts with a section *Overview*

- Objectives
- Lab settings: passwords, device names
- Roadmap: organization of the lab

*Section 1*

- Background information of the topic being covered (e.g., fundamentals of TCP congestion control)
- Section 1 is optional (i.e., the reader can skip this section and move to lab directions)

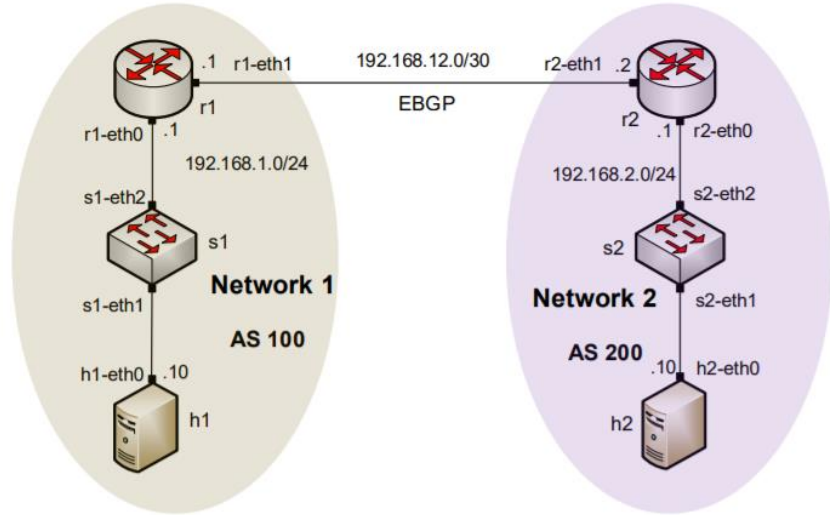
*Section 2... n*

- Step-by-step directions

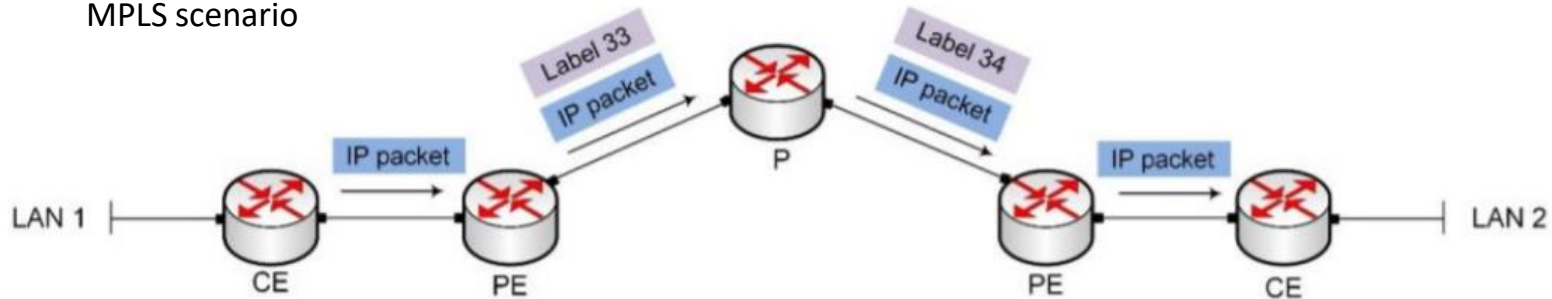
# Examples

## Legacy networks

BGP scenario



MPLS scenario

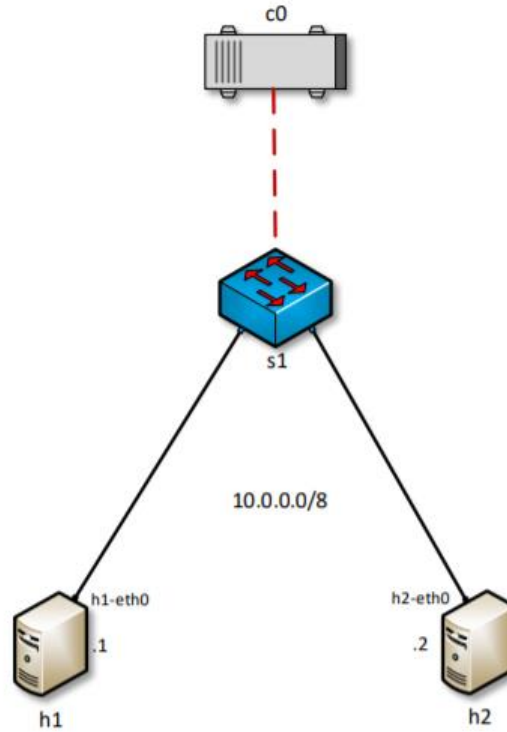




# Examples

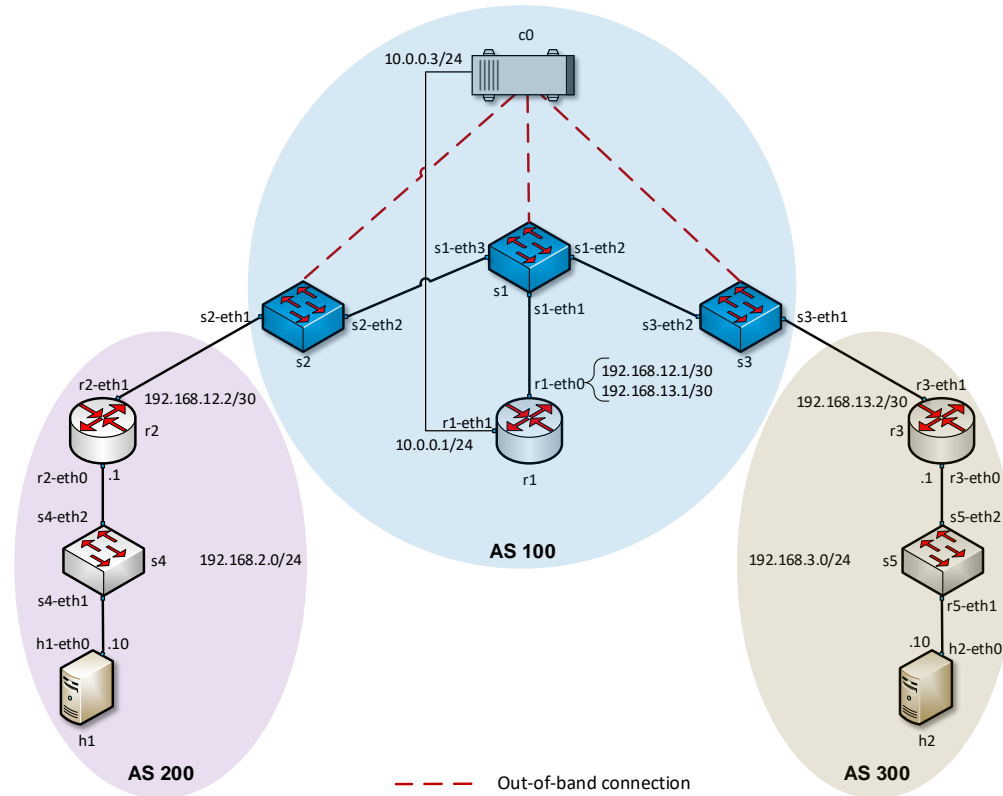
---

## SDN networks



# Examples

## Interconnection of SDN and legacy networks



# Overview Open Virtual Switch Lab Series

# Open vSwitch Lab Series

---

- Open vSwitch (OvS), is an open-source implementation of a distributed virtual multilayer switch
- The main purpose of OvS is to provide a switching stack for hardware virtualization environments, while supporting multiple protocols and standards
- The lab series provides a practical experience on Open vSwitch features
  - Linux Namespaces
  - OpenFlow
  - Traffic isolation with VLAN
  - Open vSwitch Kernel Datapath

# Open vSwitch Lab Series

---

## Lab experiments

Lab 1: Introduction to Linux namespaces and Open vSwitch

Lab 2: Introduction to Mininet

Lab 3: Open vSwitch Flow table

Lab 4: Introduction to Open vSwitch

Lab 5: Implementing VLANs in Open vSwitch

Lab 6: VLAN trunking in Open vSwitch

Lab 7: Implementing Routing in Open vSwitch

Lab 8: Open vSwitch Database Management Protocol (OVSDB)

Lab 9: Open vSwitch Kernel Datapath

Lab 10: Configuring Stateless Firewall using ACLs

Lab 11: Configuring Stateful Firewall using Connection Tracking

Lab 12: Configuring GRE Tunnel

# Organization of Lab Manuals

---

Each lab starts with a section *Overview*

- Objectives
- Lab settings: passwords, device names
- Roadmap: organization of the lab

*Section 1*

- Background information of the topic being covered (e.g., fundamentals of TCP congestion control)
- Section 1 is optional (i.e., the reader can skip this section and move to lab directions)

*Section 2... n*

- Step-by-step directions

# Overview Zeek Intrusion Detection Lab Series

# Zeek Lab Series

The lab series introduces learners to an emulated Intrusion Detection System (IDS) that actively monitors live networks for malicious traffic, policy violations and unidentified anomalies

It helps students to acquire hands-on skills on

- Understanding Network Intrusion Detection Systems
- Creating scripts to identify network traffic signatures
- Emulating scenarios to detect Denial of Service (DoS) attacks
- Developing Machine Learning classifiers for anomaly inference and classification



# Zeek Lab Series

The lab series can be partitioned into four parts

- Overview of the basic features of Zeek such as parsing, reading and organizing Zeek log files
- Generating, capturing and analyzing network traffic using open-source tools (e.g., nmap, tcpdump, Wireshak)
- Introduction to Zeek scripting
- Using Machine Learning features to infer and classify anomalies

# Zeek Lab Series

---

## Lab experiments

Lab 1: Introduction to the Capabilities of Zeek

Lab 2: An Overview of Zeek Logs

Lab 3: Parsing, Reading and Organizing Zeek Log Files

Lab 4: Generation, Capturing and Analyzing Network Scanner Traffic

Lab 5: Generation, Capturing and Analyzing DoS and DDoS-centric Network Traffic

Lab 6: Introduction to Zeek Scripting

Lab 7: Introduction to Zeek Signatures

Lab 8: Advanced Zeek Scripting for Anomaly and Malicious Event Detection

Lab 9: Profiling and Performance Metrics of Zeek

Lab 10: Application of the Zeek IDS for Real-Time Network Protection

Lab 11: Preprocessing of Zeek Output Logs for Machine Learning

Lab 12: Developing Machine Learning Classifiers for Anomaly Inference and Classification

# Organization of Lab Manuals

---

Each lab starts with a section *Overview*

- Objectives
- Lab settings: passwords, device names
- Roadmap: organization of the lab

*Section 1*

- Background information of the topic being covered (e.g., creating Zeek scripts for anomaly detection)
- Section 1 is optional (i.e., the reader can skip this section and move to lab directions)

*Section 2... n*

- Step-by-step directions

**Thank you**

