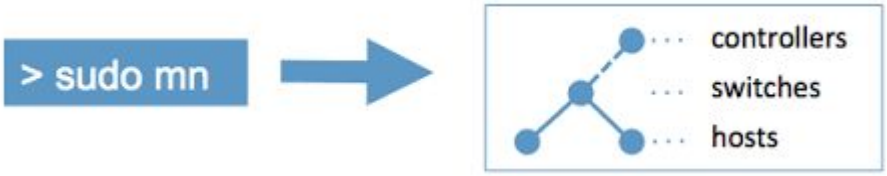


Mininet : An Instant Virtual Network on your Laptop (or other PC)

Ramon Fontes (PhD Candidate)
Danny Alex Lachos (PhD Candidate)
Prof. Christian Esteve Rothenberg
(INTRIG@DCA/FEEC/UNICAMP)

INTRIG
INFORMATION & NETWORKING TECHNOLOGIES RESEARCH & INNOVATION GROUP

Class Goals

- Learn how Mininet (and network emulation in general) works, and how it can be used for network experimentation
- Learn how Mininet-WiFi (under development by PhD candidate @ FEEC / UNICAMP) allows for realistic, wireless experiments

Source / References:

1. Introduction to Mininet - SIGCOMM 2014 Tutorial: https://docs.google.com/presentation/d/1Xtp05lLQTEFGICTxzV9sQl28wW_cAZz6B1q9_qZBR_8/edit#slide=id.g3942f5b2a_044
2. Mininet-WiFi: https://github.com/intrig-unicamp/mininet-wifi

Agenda

1. Introduction to Mininet

motivation, presentation, demos

2. Introduction to Mininet-WiFi

presentation, demos

Introduction to Mininet

Platforms for Network/Systems Experimentation

Network Emulator Architecture

Mininet: Basic Usage, CLI, API

Conclusion and Questions

Platforms for Network/Systems Experimentation

Platform	Advantages	Disadvantages
Hardware testbed	▪ Fast ▪ Accurate: “ground truth”	▪ Expensive ▪ Hard to reconfigure ▪ Hard to change ▪ Hard to download
Simulator	▪ Inexpensive, flexible ▪ Detailed ▪ Easy to download ▪ Virtual time	▪ May require app changes ▪ Might not run OS code ▪ May not be “believable” ▪ May be slow/non-interactive
Emulator	▪ Inexpensive, flexible ▪ Real code ▪ Reasonably accurate ▪ Easy to download ▪ Fast/interactive usage	▪ Slower than hardware ▪ Possible inaccuracy from multiplexing

Context: Platforms for Network Experimentation and Development

Container-based emulators: CORE, virtual Emulab, Trellis, Imunes, even ns-3 (in emulation mode), **Mininet, Mininet-WiFi**

VM-based emulators: DieCast

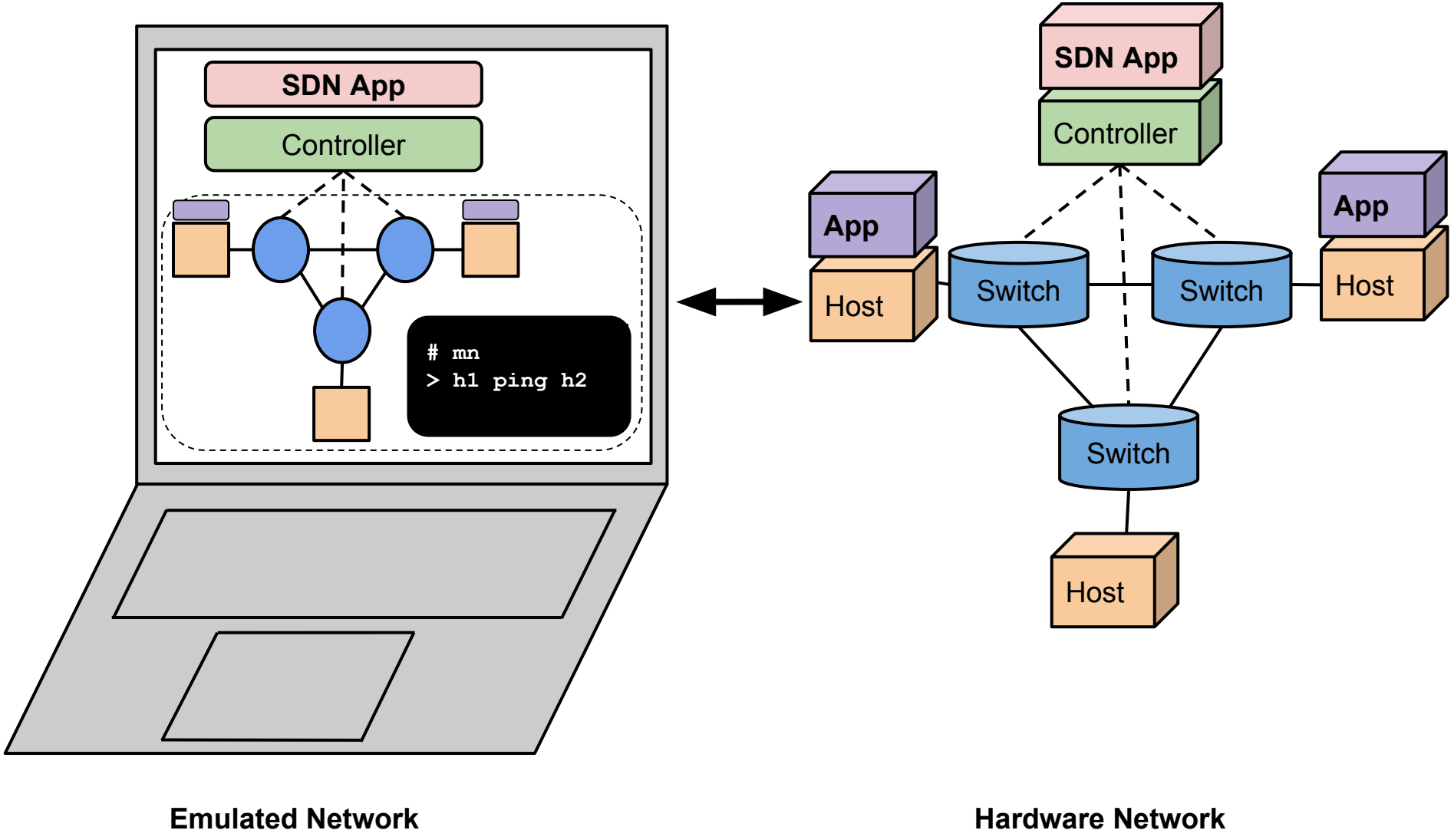
UML-based emulators: NetKit

Simulators: ns-3, OPNET

Testbeds: Emulab, GENI, PlanetLab, ORBIT

All of these are fine, but Emulators are particularly useful! Why? Because...

Apps move seamlessly to/from hardware



Introduction to Mininet

Platforms for Network/Systems Experimentation

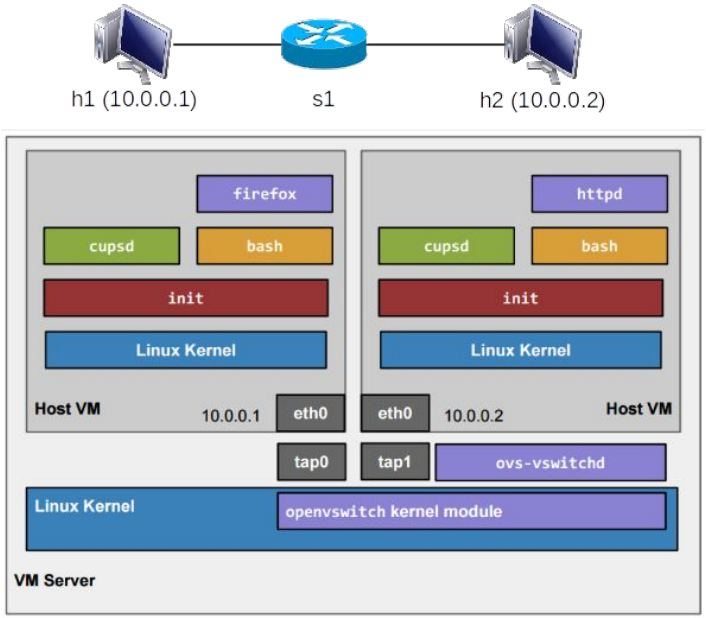
Network Emulator Architecture

Mininet: Basic Usage, CLI, API

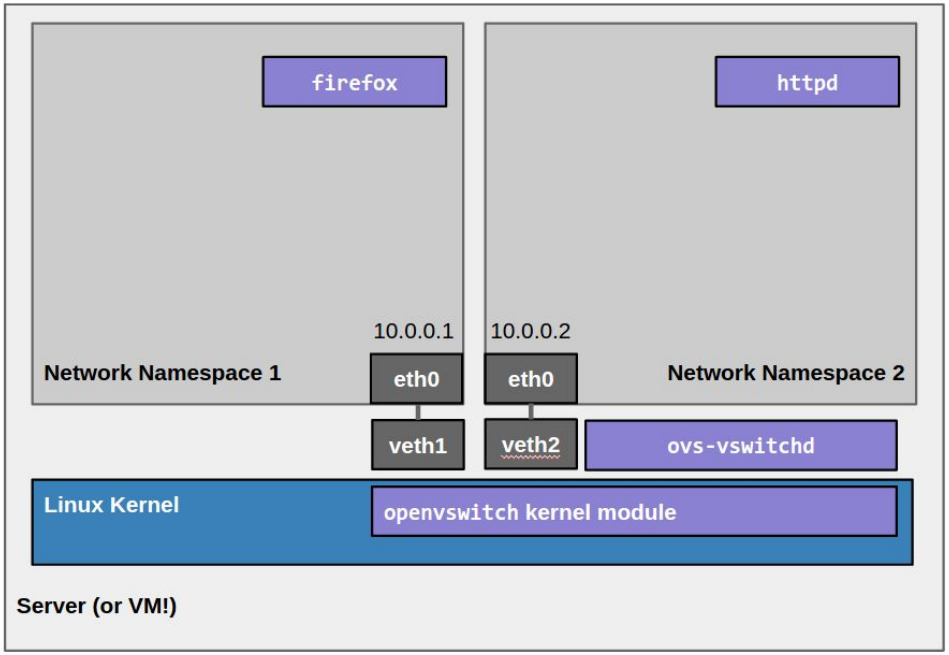
Conclusion and Questions

Hardware based Network Testbed (1/3)

Very Simple Network using
Full System Virtualization

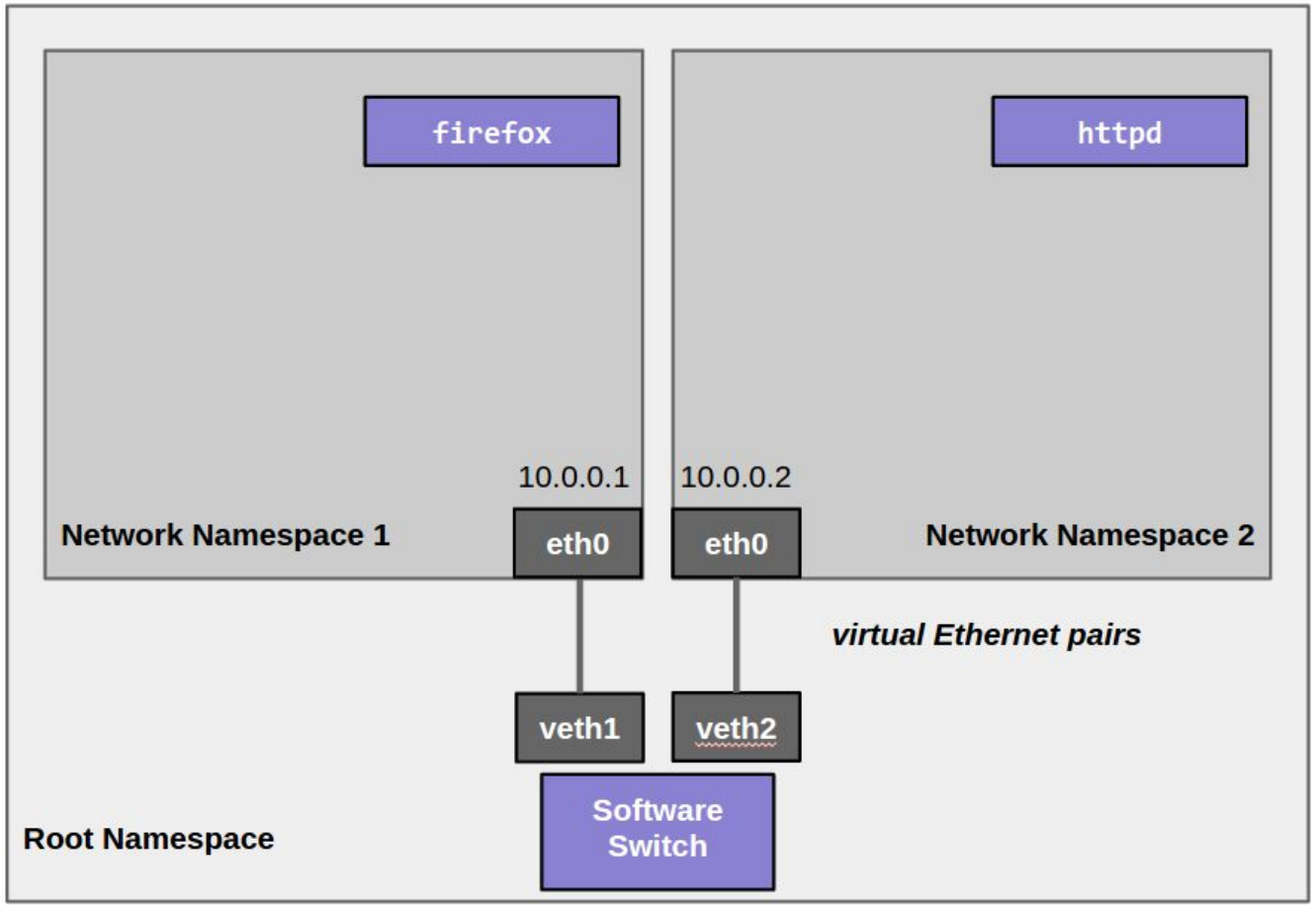


Very Simple Network using
Lightweight Virtualization



Hardware based Network Testbed (2/3)

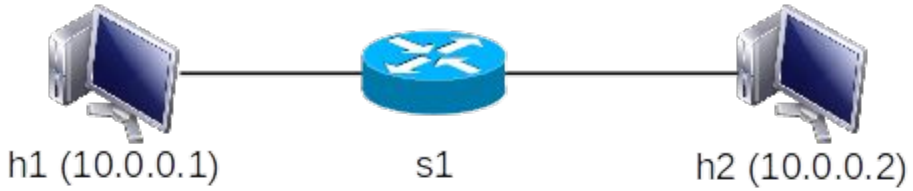
Mechanism: ***Network Namespaces and Virtual Ethernet Pairs***



Hardware based Network Testbed (3/3)

Problems:

- Too much work even for creating such a simple network topology
- Not programmable




```
sudo bash
# Create host namespaces
ip netns add h1
ip netns add h2
# Create switch
ovs-vsctl add-br s1
# Create links
ip link add h1-eth0 type veth peer name s1-eth1
ip link add h2-eth0 type veth peer name s1-eth2
ip link show
# Move host ports into namespaces
ip link set h1-eth0 netns h1
ip link set h2-eth0 netns h2
ip netns exec h1 ip link show
ip netns exec h2 ip link show
```

```
# Connect switch ports to OVS
ovs-vsctl add-port s1 s1-eth1
ovs-vsctl add-port s1 s1-eth2
ovs-vsctl show
# Set up OpenFlow controller
ovs-vsctl set-controller s1 tcp:127.0.0.1
ovs-controller ptcp: &
ovs-vsctl show
# Configure network
ip netns exec h1 ifconfig h1-eth0 10.1
ip netns exec h1 ifconfig lo up
ip netns exec h2 ifconfig h2-eth0 10.2
ip netns exec h1 ifconfig lo up
ifconfig s1-eth1 up
ifconfig s1-eth2 up
# Test network
ip netns exec h1 ping -c1 10.2
```

Wouldn’t it be great if...

We had a simple command-line tool and/or API that did this for us automatically?

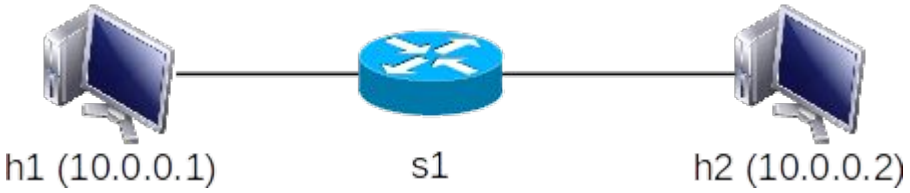
It allowed us to easily create topologies of varying size, up to hundreds of nodes, and run tests on them?

It was already included in Ubuntu?

Emulator based Network Testbed

What We Want is:

- A simple command-line tool / API which can ease the work
- The solution should allow us to easily create topologies for varying size, up to hundreds and thousand of nodes




```
h1 = net.addHost( 'h1' )
h2 = net.addHost( 'h2' )
s1 = net.addSwitch( 's1' )
c0 = net.addController( 'c0' )
net.addLink( h1, s1 )
net.addLink( h2, s1 )
net.start()
CLI( net )
```

Topology Generation using Mininet API

Mininet: An Instant Virtual Network on your Laptop (or other PC) - Mininet

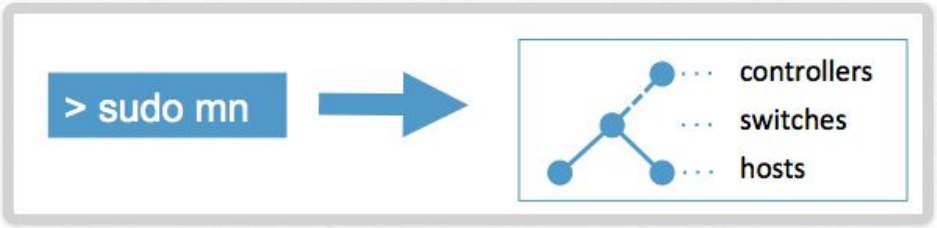
mininet.org

Reader

Mininet

An Instant Virtual Network on your Laptop (or other PC)

Mininet creates a **realistic virtual network**, running **real kernel, switch and application code**, on a single machine (VM, cloud or native), in seconds, with a single command:



Because you can easily interact with your network using the Mininet CLI (and API), customize it, share it with others, or deploy it on real hardware, Mininet is useful for development, teaching, and research.

Mininet is also a great way to develop, share, and experiment with OpenFlow and Software-Defined Networking systems.

Mininet is actively developed and supported, and is released under a permissive BSD Open Source license. We encourage you to contribute code, bug reports/fixes, documentation, and anything else that can improve the system!

Get Started

Download a Mininet VM, do the walkthrough and run the OpenFlow tutorial.

Support

Read the FAQ, read the documentation, and join our mailing list, mininet-discuss.

Contribute

File a bug, download the source, or submit a pull request - all on GitHub.

News

Mininet Tutorial at SIGCOMM

Announcing Mininet 2.1.0 !

Nick Feamster's SDN Course

Automating Controller Startup

Display a menu

Mininet is a *Network Emulator*

In this talk, ***emulation*** (or running on an **emulator**) means running *unmodified* code *interactively* on *virtual hardware* on a *regular PC*, providing convenience and realism at low cost – with some limitations (e.g. speed, detail.)

This is in contrast to running on a **hardware testbed** (fast, accurate, expensive/shared) or a **simulator** (cheap, detailed, but perhaps slow and requiring code modifications.)

Introduction to Mininet

Platforms for Network/Systems Experimentation

Network Emulator Architecture

Mininet: Basic Usage, CLI, API

Conclusion and Questions

Mininet's Python API

Core of Mininet!! Everything is built on it.

Python >> JSON/XML/etc.

Easy and (hopefully) fun

Python is used for *orchestration*, but emulation is performed by compiled C code (Linux + switches + apps)

api.mininet.org

docs.mininet.org

Introduction to Mininet

Mininet command line Interface Usage 1/3

Interact with hosts and switches

Start a minimal topology

```
$ sudo mn
```

Start a minimal topology using a remote controller

```
$ sudo mn --controller=remote,ip=[IP_ADDDR],port=[listening port]
```

Start a custom topology

```
$ sudo mn --custom [topo_script_path] --topo=[topo_name]
```

Display nodes

```
mininet> nodes
```

Display links

```
mininet> net
```

Dump information about all nodes

```
mininet> dump
```

Mininet command line Interface Usage 2/3

Interact with hosts and switches

Check the IP address of a certain node

```
mininet> h1 ifconfig -a
```

Print the process list from a host process

```
mininet> h1 ps -a
```

Test connectivity between hosts

Verify the connectivity by pinging from h1 to h2

```
mininet> h1 ping -c 1 h2
```

Verify the connectivity among all hosts

```
mininet> pingall
```

Mininet command line Interface Usage 3/3

Run a regression test

Traffic receive preparation

```
mininet> iperf -s -u -p [port_num] &
```

Traffic generation from client

```
mininet> iperf -c [IP] -u -t [duration] -b [bandwidth] -p [port_num] &
```

Link variations

```
$ sudo mn -link tc,bw=[bandwidth],delay=[delay_in_millisecond]
```

Python Interpreter

Print accessible local variables

```
mininet> py locals()
```

Execute a method through invoking mininet API

```
mininet> py [mininet_name_space].[method]
```

Mininet Application Programming Interface Usage 1/4

Low-level API: nodes and links

- **mininet.node.Node**
  A virtual network node, which is a simply in a network namespace
- **mininet.link.Link**
  A basic link, which is represented as a pair of nodes

Method	Description
MAC/setMAC	Return/Assign MAC address of a node or specific interface
IP/setIP	Return/Assign IP address of a node or specific interface
cmd	Send a command, wait for output, and return it
terminate	Send kill signal to Node and clean up after it
Link	Create a link to another node, make two new interfaces

```
h1 = Host( 'h1' )
h2 = Host( 'h2' )
s1 = OVSSwitch( 's1', inNamespace=False )
c0 = Controller( 'c0', inNamespace=False )
Link( h1, s1 )
Link( h2, s1 )
h1.setIP( '10.1/8' )
h2.setIP( '10.2/8' )
```

```
c0.start()
s1.start( [ c0 ] )
print h1.cmd( 'ping -c1', h2.IP() )
s1.stop()
c0.stop()
```

Mininet Application Programming Interface Usage 2/4

Middle-level API: network object

- **mininet.net.Mininet**
  Network emulation with hosts spawned in network namespaces

Class	Method	Description
Net	addHost	Add a host to network
	addSwitch	Add a switch to network
	addLink	Link two nodes into together
	addController	Add a controller to network
	getNodeByName	Return node(s) with given name(s)
	start	Start controller and switches
	stop	Stop the controller, switches and hosts
	ping	Ping between all specified hosts and return all data

```
net = Mininet()
h1 = net.addHost( 'h1' )
h2 = net.addHost( 'h2' )
s1 = net.addSwitch( 's1' )
c0 = net.addController( 'c0' )
net.addLink( h1, s1 )
net.addLink( h2, s1 )
```

```
net.start()
print h1.cmd( 'ping -c1', h2.IP() )
CLI( net )
net.stop()
```

Mininet Application Programming Interface Usage 3/4

High-level API: topology templates

- **mininet.topo.Topo**
  Data center network representation for structured multi-trees

Class	Method	Description
Topo	Methods similar to net	E.g., addHost, addSwitch, addLink,
	addNode	Add node to graph
	addPort	Generate port mapping for new edge
	switches	Return all switches
	Hosts/nodes/switches/links	Return all hosts
	isSwitch	Return true if node is a switch, return false otherwise

```
class SingleSwitchTopo( Topo ):
    "Single Switch Topology"
    def build( self, count=1):
        hosts = [ self.addHost( 'h%d' % i )
            for i in range( 1, count + 1 ) ]
        s1 = self.addSwitch( 's1' )
        for h in hosts:
            self.addLink( h, s1 )
```

```
net = Mininet(
topo=SingleSwitchTopo( 3 ) )
net.start()
CLI( net )
net.stop(
```

Mininet Application Programming Interface Usage 4/4

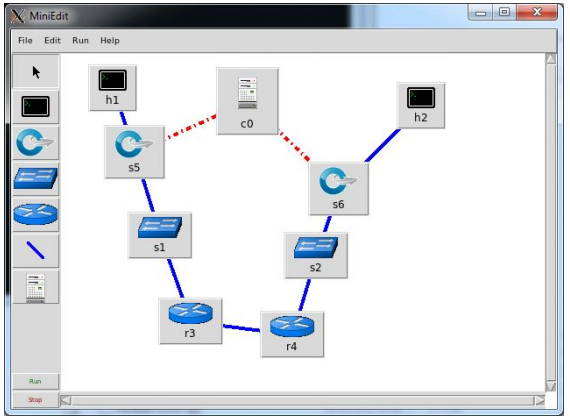
Customized topology

Examples can be found here:
https://github.com/mininet/mininet/tree/master/examples

Mininet Applications

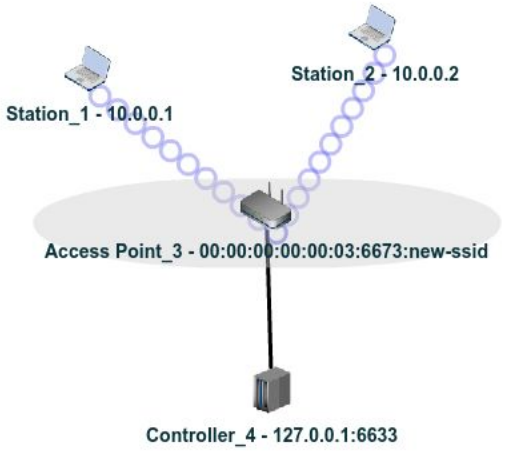
MiniEdit (python API)

- A GUI application which eases the Mininet topology generation
- Either save the topology or export as a Mininet python script



Visual Network Description (VND) (ramonfontes.com)

- A GUI tool which allows automate creation of Mininet and OpenFlow controller scripts



Introduction to Mininet

Platforms for Network/Systems Experimentation

Network Emulator Architecture

Mininet: Basic Usage, CLI, API

Conclusion and Questions

Conclusion and Questions

***Network Emulators* can facilitate teaching networking via realistic live demos, interactive labs and course assignments**

- inexpensive, interactive, real apps and OS, reasonably accurate
- downloadable, fast setup

***Mininet* is a lightweight virtualization/container based emulator**

- modest hardware requirements, fast startup, hundreds of nodes
- command line tool, CLI, simple Python API
- SDN as well as Ethernet/IP networking
- install using VM, Ubuntu package, or source

mininet.org: Tutorials, walkthroughs, API documentation and examples

teaching.mininet.org: Mininet-based **course assignments and labs**

open source: hosted on github, permissive BSD license

Tutorial Agenda

1. Introduction to Mininet

motivation, presentation, demos

2. Introduction to Mininet-WiFi

motivation, presentation, demos

Mininet-WiFi

Ramon Fontes and Christian Esteve Rothenberg
(FEEC/UNICAMP)

Main Goal

Aims at providing high fidelity emulation of wireless networks enabling real network analysis in fully controlled environments in support of research on Wireless and SDWN.

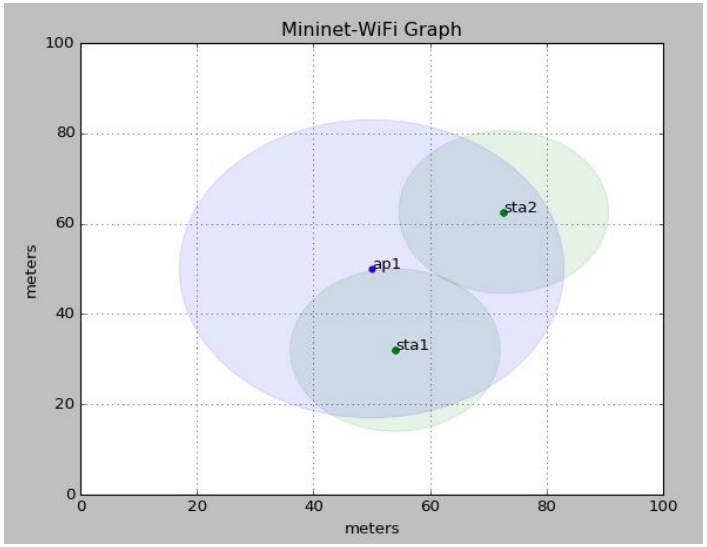
Resources available

Infra and adhoc

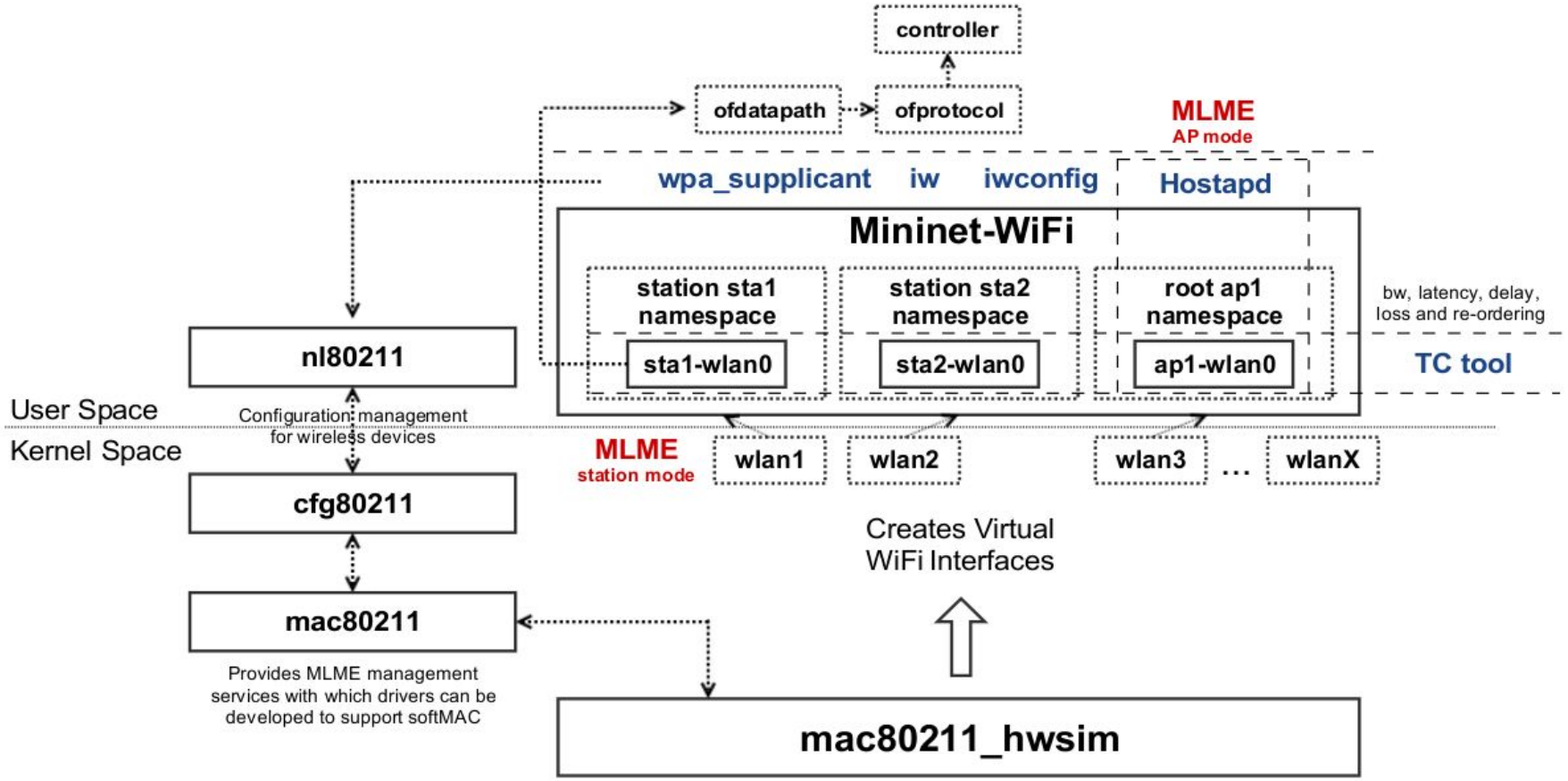
Mobility

Propagation Models

Replaying Network Conditions



Main Components



Mininet-WiFi command line Interface Usage

Interact with hosts and switches

Start a minimal topology

```
$ sudo mn --wifi
```

Verify the connectivity by pinging from sta1 to sta2

```
mininet-wifi> sta1 ping sta2
```

Verify what access points are available

```
mininet-wifi> sta1 iw dev sta1-wlan0 scan
```

Verify the position of station 1

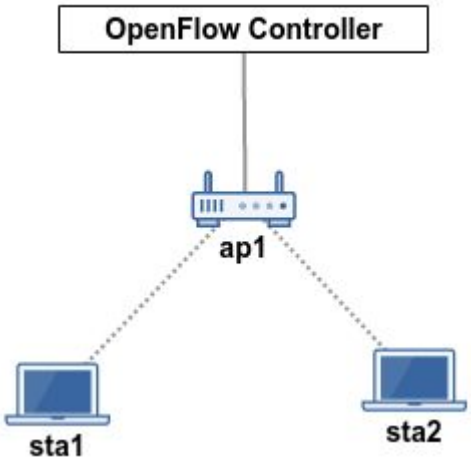
```
mininet-wifi> py sta1.params['position']
```

Verify the distance between sta1 and sta2

```
mininet-wifi> distance sta1 sta2
```

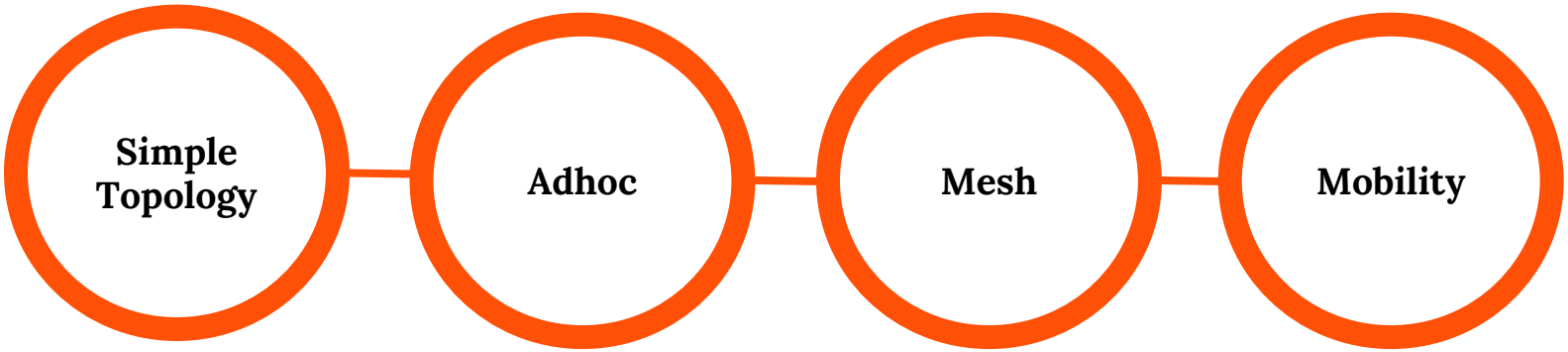
Further information of station 1

```
mininet-wifi> py sta1.params
```



Many Examples are Available

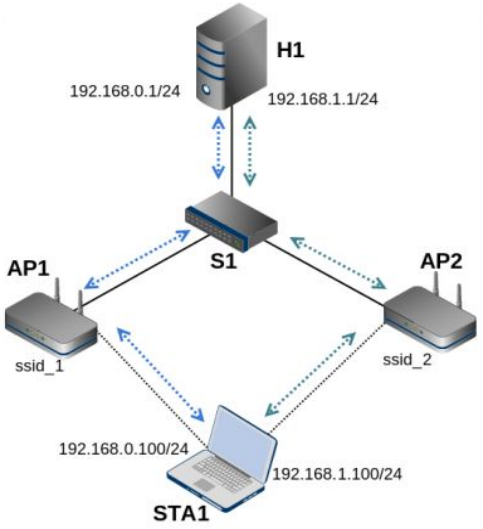
examples/

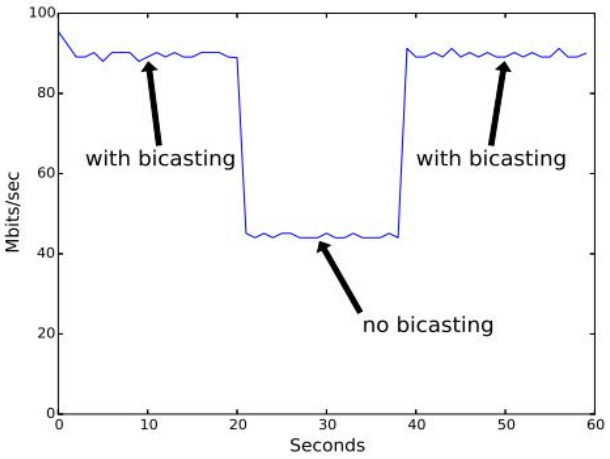


and others...

Case Studies

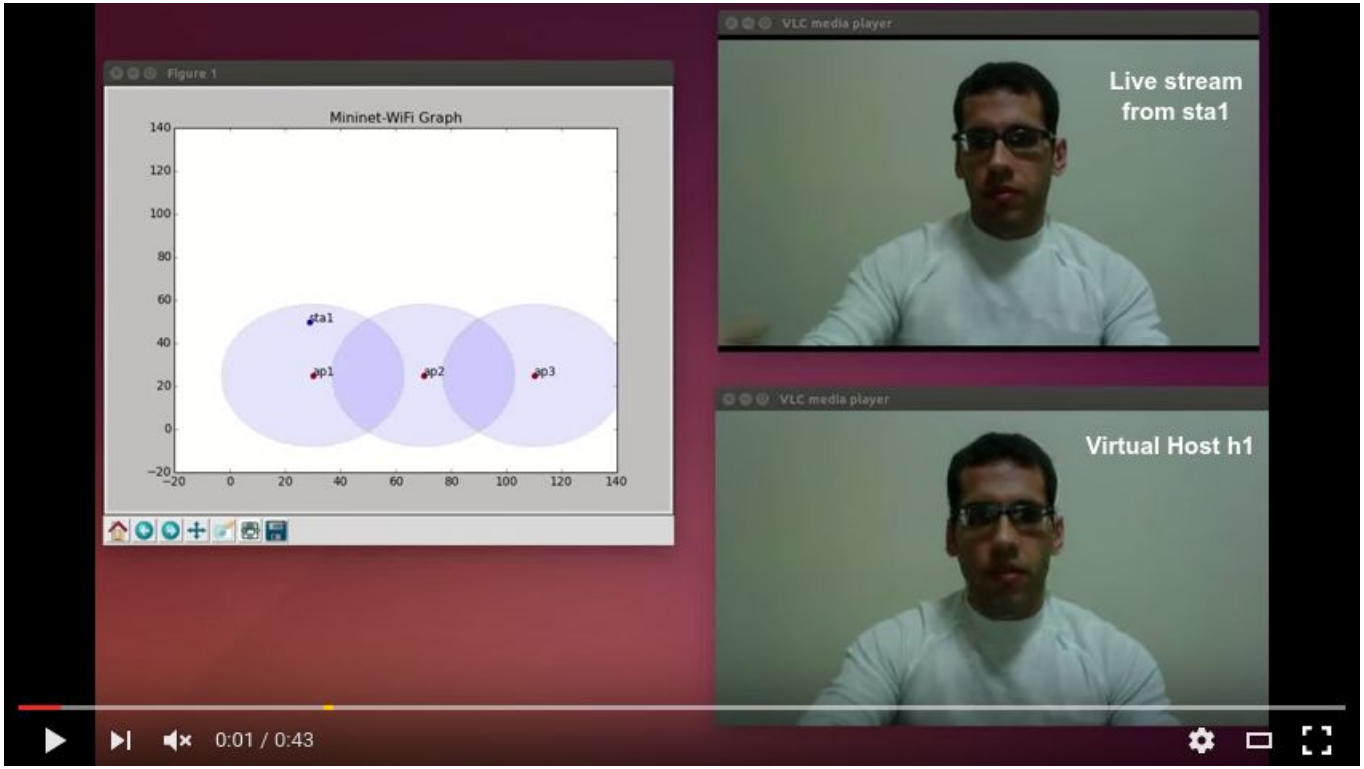
#1 Wireless Bicasting





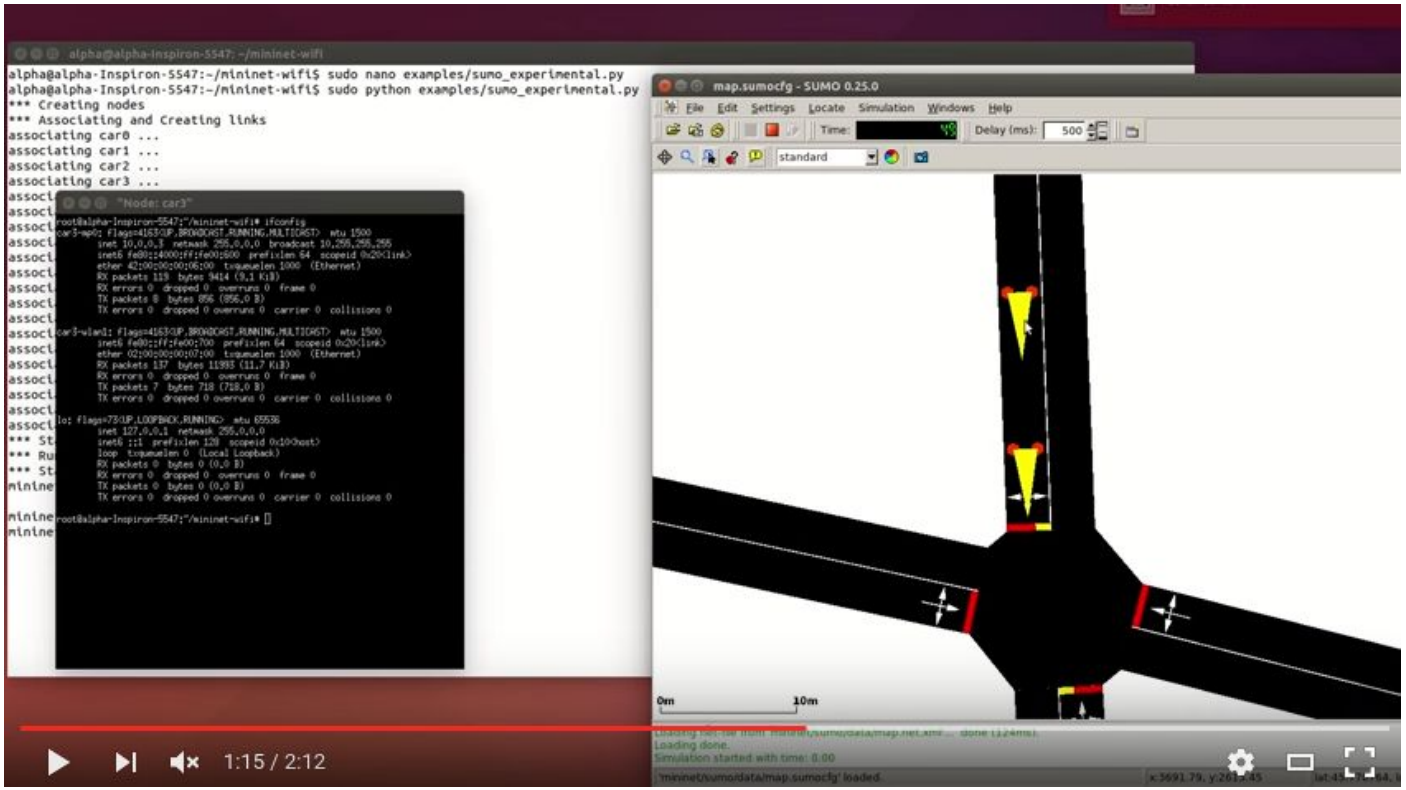
Case Studies

#2 Using all the wireless networks around us



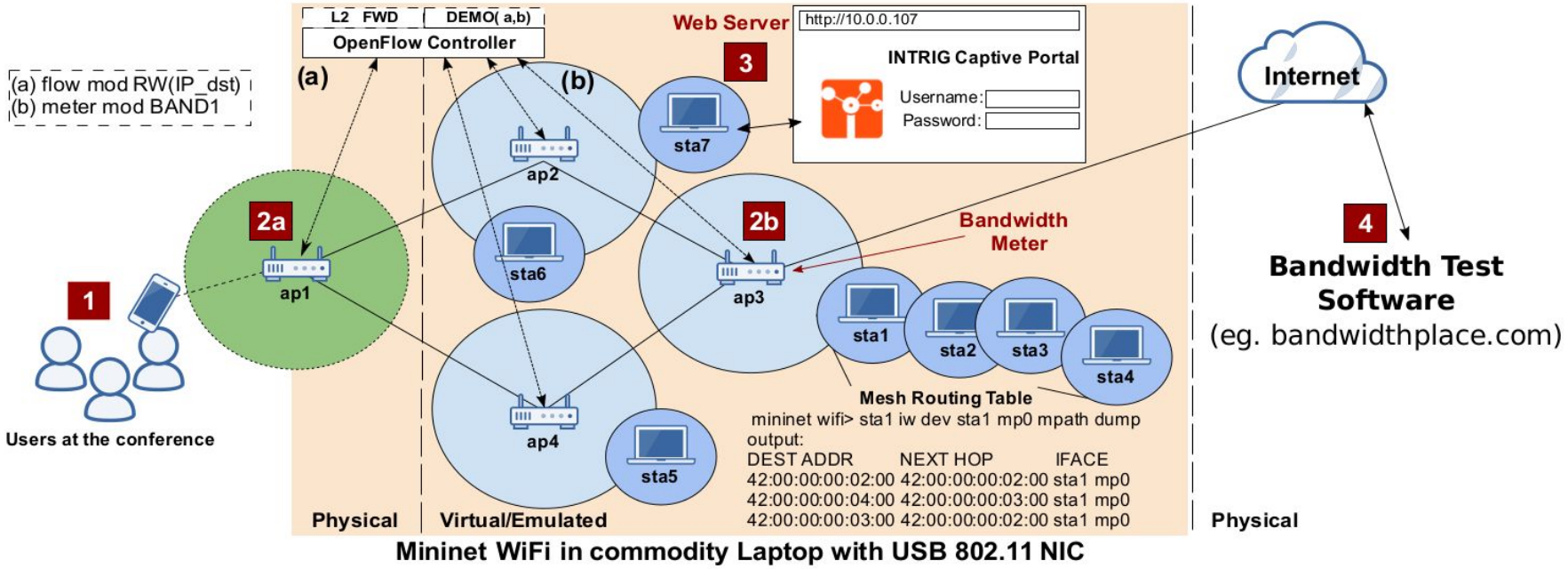
Case Studies

#3 Urban Mobility



Case Studies

#4 Integration between physical/virtual environment



Mininet WiFi in commodity Laptop with USB 802.11 NIC

Available at:

https://github.com/**intrig-unicamp/mininet-wifi**

Join us:

https://groups.google.com/forum/#!forum/mininet-wifi-discuss

Backup/Supplementary Slides