

Deploying OpenFlow experiments on the Virtual Wall testbed



Maxim Claeys
Jeroen Famaey
Niels Bouten

Ghent University
iMinds

Outline

1. The Virtual Wall network emulation testbed

2. Fed4FIRE: Federation of Future Internet testbeds

3. Executing a basic Virtual Wall experiment using jFed

4. OpenFlow protocol basics

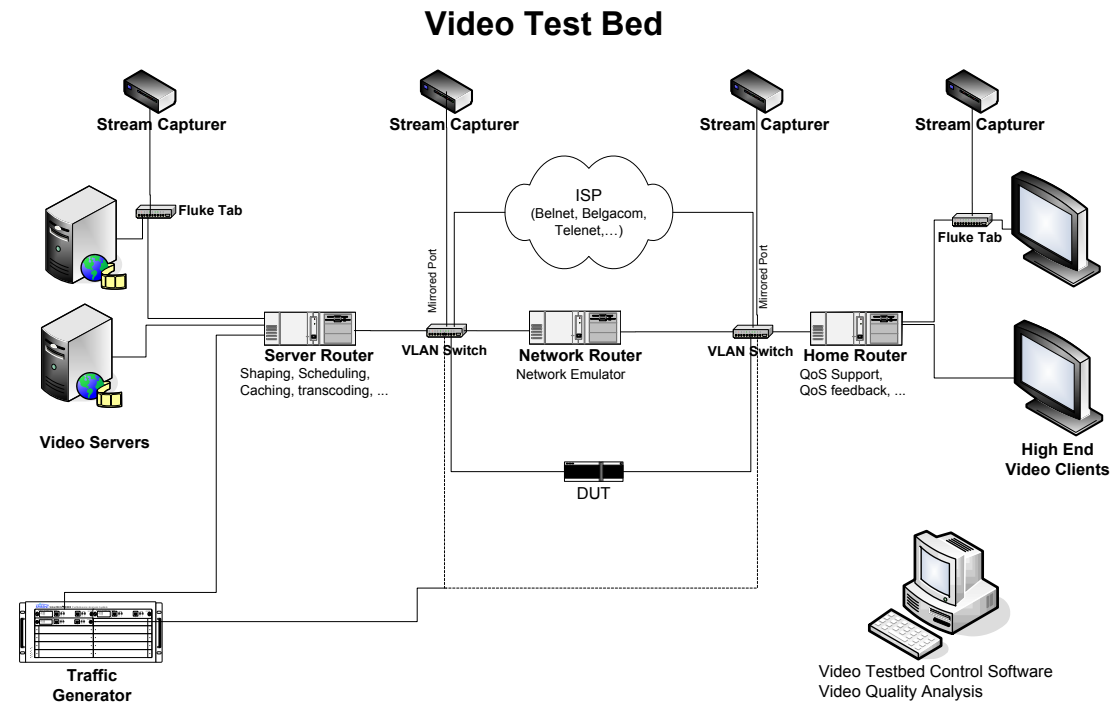
5. Running OpenFlow experiments on the Virtual Wall

The Virtual Wall network emulation testbed

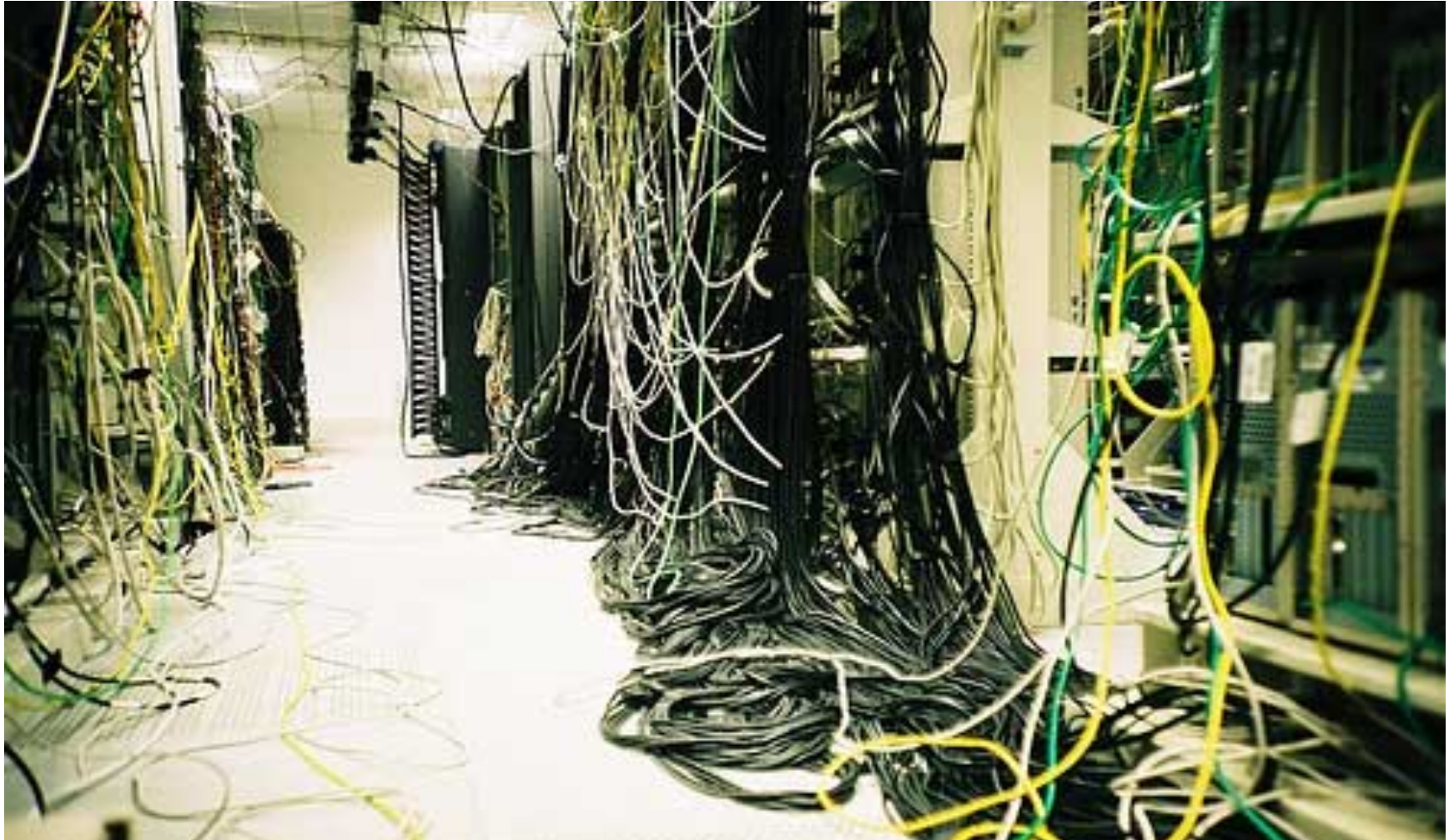
Doing large test setups – The hard way

Creating large test setups

- Can be very time consuming
- Requires a lot of hardware
- No shared usage of hardware



After a few years (or months?)



The iMinds Virtual Wall



- ✓ Automated topology creation and device configuration
- ✓ Full automatic install of OS & other software
- ✓ Fast swap-in and swap-out of experiments
- ✓ Experiment management tools

The Virtual Wall

Displays for visualization & demos



Server nodes

Emulab

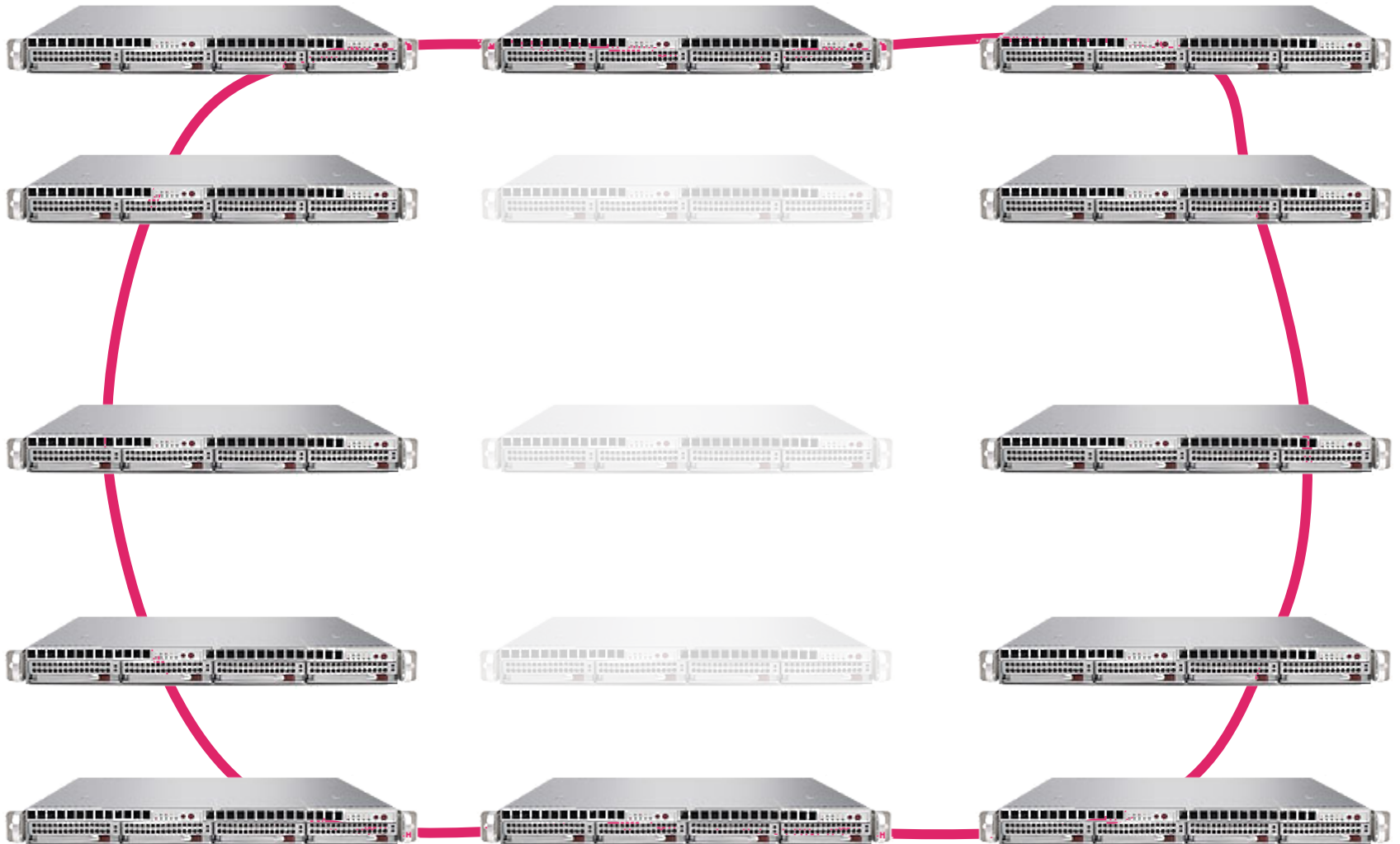
- University of Utah testbed (hard- and software)
- Predecessor of the Virtual Wall
- Testbed management software freely available



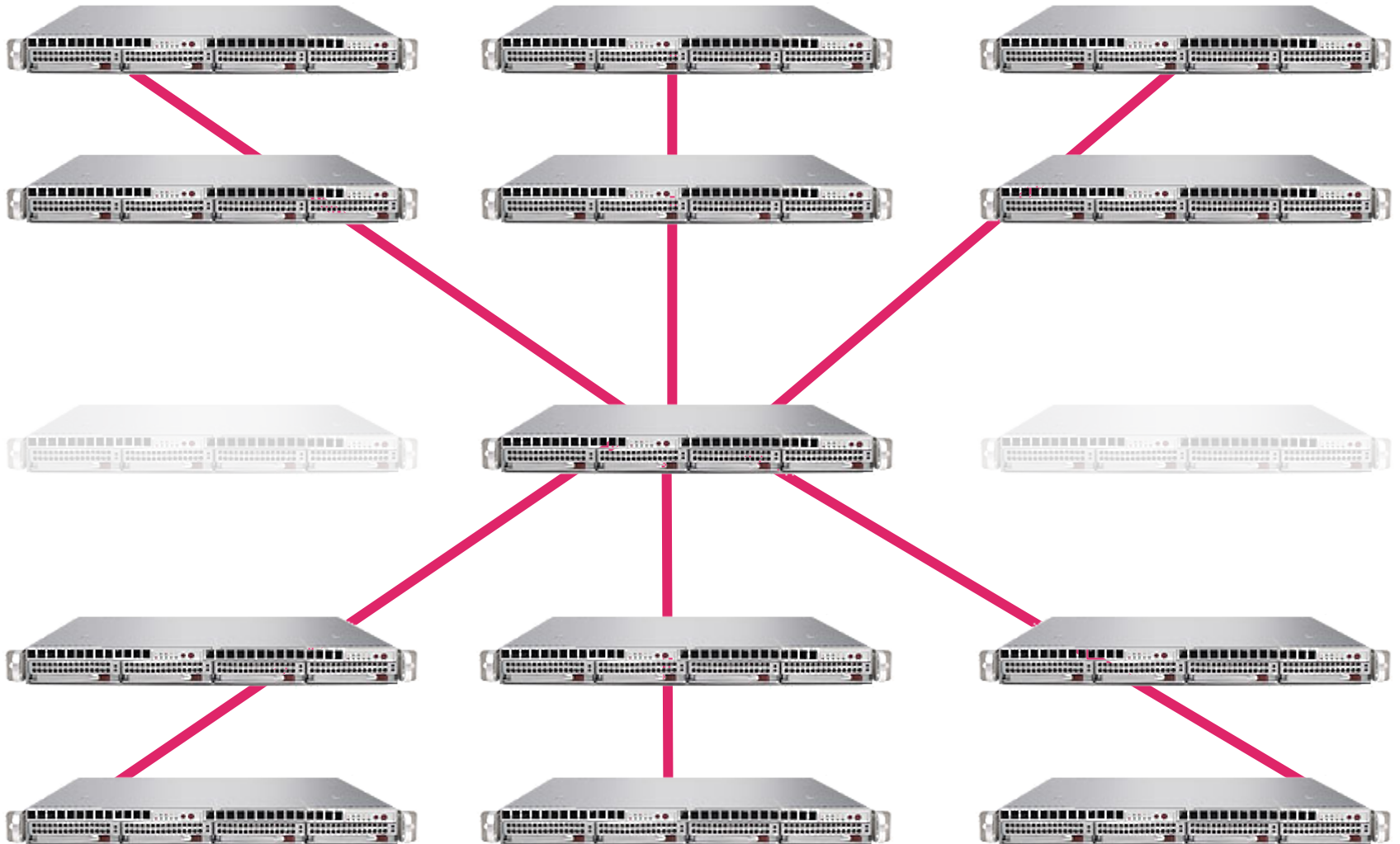
The Virtual Wall Concept



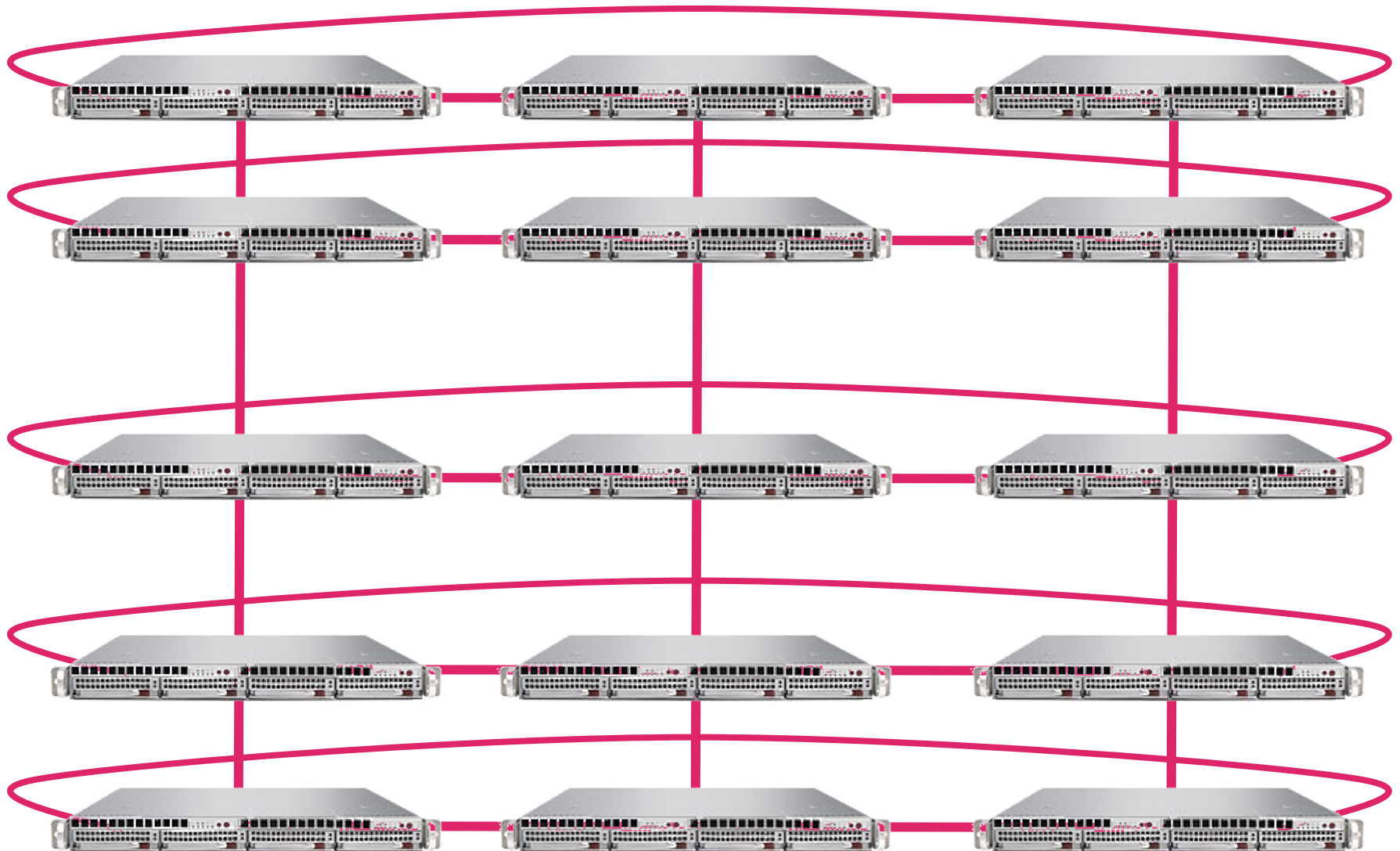
Virtual Wall: Topology Control



Virtual Wall: Topology Control



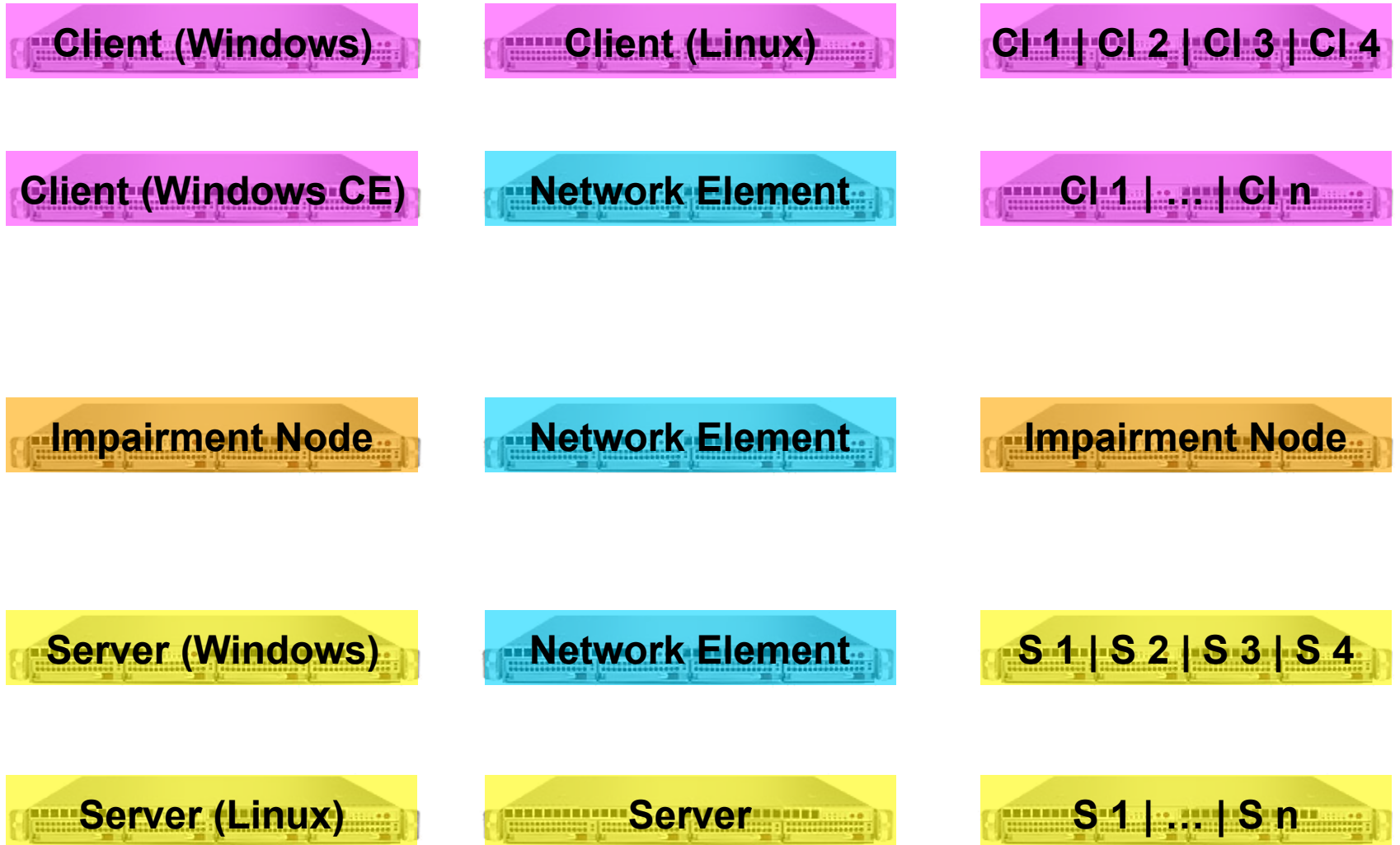
Virtual Wall: Topology Control



Virtual Wall: Node Functionality



Virtualization



Two Wall Setups: Virtual Wall 1

- 200 servers, “older” nodes
 - Dual CPU, quad or eight cores
 - 4 – 12 GB RAM
 - 2 – 6 network interfaces
- Central switch: Force 10 networks
 - 336 x Gb/s port
 - 8 x 10 Gb/s port
 - 1.53 Tb/s backplane



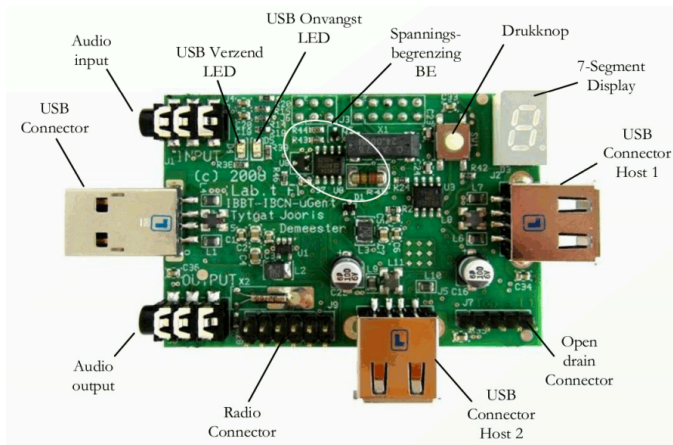
Two Wall Setups: Virtual Wall 2

- 100 servers, “newer” nodes
 - 12 cores @ 2.40GHz
 - 24GB RAM
 - 2-5 network interfaces
- 1 Super computational node
- 1 Super graphical node
- Central switch: Force 10 networks



Connection to the iMinds wilab.t

- Virtual Wall can be linked to Wireless testbed
- Limited external interference
- 60 fixed nodes and 20 mobile node carriers
 - Zotac embedded PC
 - Rmon sensor node
 - Bluetooth
 - Subset equipped with Webcam



Wrap-up: Virtual Wall

- Automate time-consuming manual testbed setup
 - Automatic topology, network and device configuration
 - Reuse of hardware
- Connected to
 - iMinds wireless wilab.t testbed
 - Other European FIRE facilities
- Publicly available
 - For iMinds project partners
 - Through FIRE initiatives (BonFIRE, Fed4FIRE)

Fed4FIRE: Federation of Future Internet testbeds

Fed4FIRE – general info

- IP project coordinated by iMinds
- 10/2012 - 9/2016
- Total budget: 7.75 MEUR
- 28 partners



University of Thessaly



National
Technical
University of
Athens



University of
BRISTOL



NIA



OÉ Gaillimh
NUI Galway

LANCASTER
UNIVERSITY



WOOX
INNOVATIONS



University of
Kent

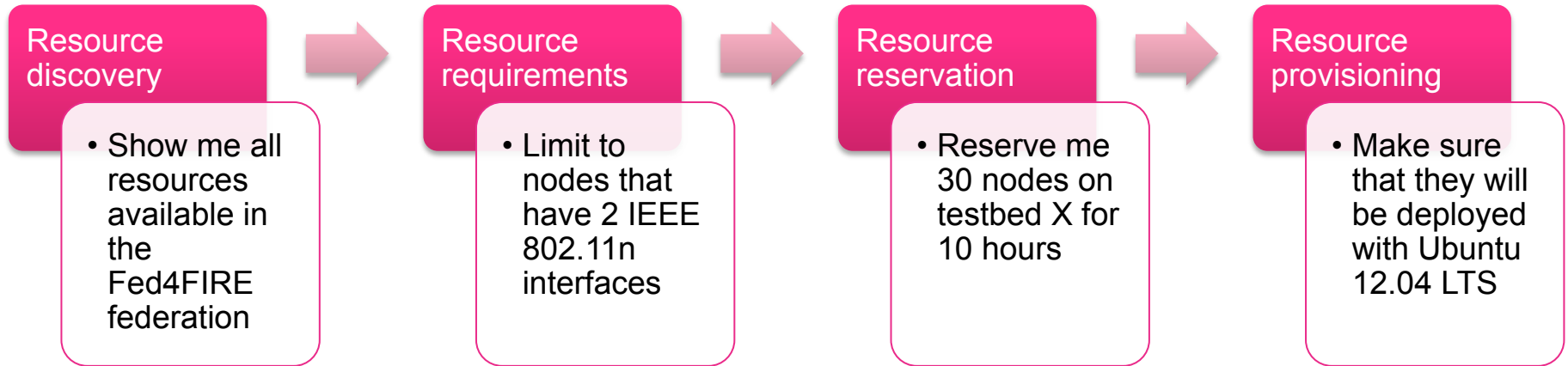


Televés

What's in it for the experimenter?

- Access a wide range of FIRE testbeds.
- Create experiments that break the boundaries of the different individual FIRE domains (wireless, wired, OpenFlow, cloud computing, smart cities, services, etc.)
- Easily access all the required resources with a single account.
- Focus on your core task of experimentation, instead of on practical aspects such as learning to work with different tools for each testbed, requesting accounts on each testbed separately, etc.

Example of the experiment lifecycle



Experiment control

- After 10 s, start data stream of 10 Mbps with source node 1, after 30 s start second data stream of 5 Mbps with source node 5.

Monitoring

- Facility monitoring: crucial servers up and running? → testbed up and running
- Infrastructure monitoring: CPU load, number of transmit errors
- Experiment measurement: measure end-to-end throughput, delay and jitter.

Permanent storage

- Store measurements on the storage server of testbed X for later analysis

Resource release

- I'm done with them after 5h already, release my resources so they can be used by other experimenters.

How to configure a Wall experiment



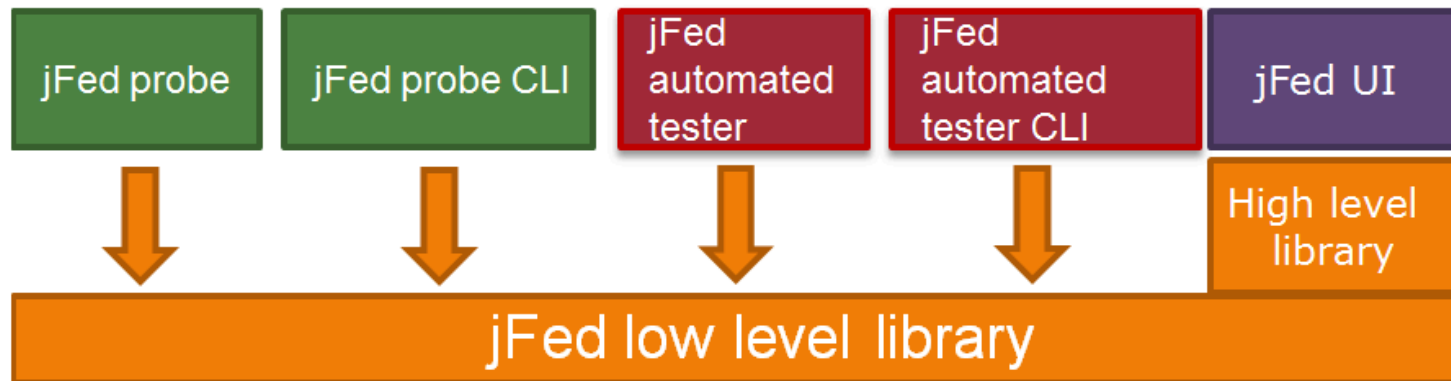
Emulab configuration
(AIMS 2013)

jFed Experimenter
(Fed4FIRE)



Common tool: jFed

- Java based framework supporting SFA testbed federation client tools
- Includes automated testing tools and an experimenter tool



Resource Specifications (RSpec)

- RSpecs are XML documents that describe resources
 - Machines, VMs, links, etc.

RSpec for a physical machine with one interface:

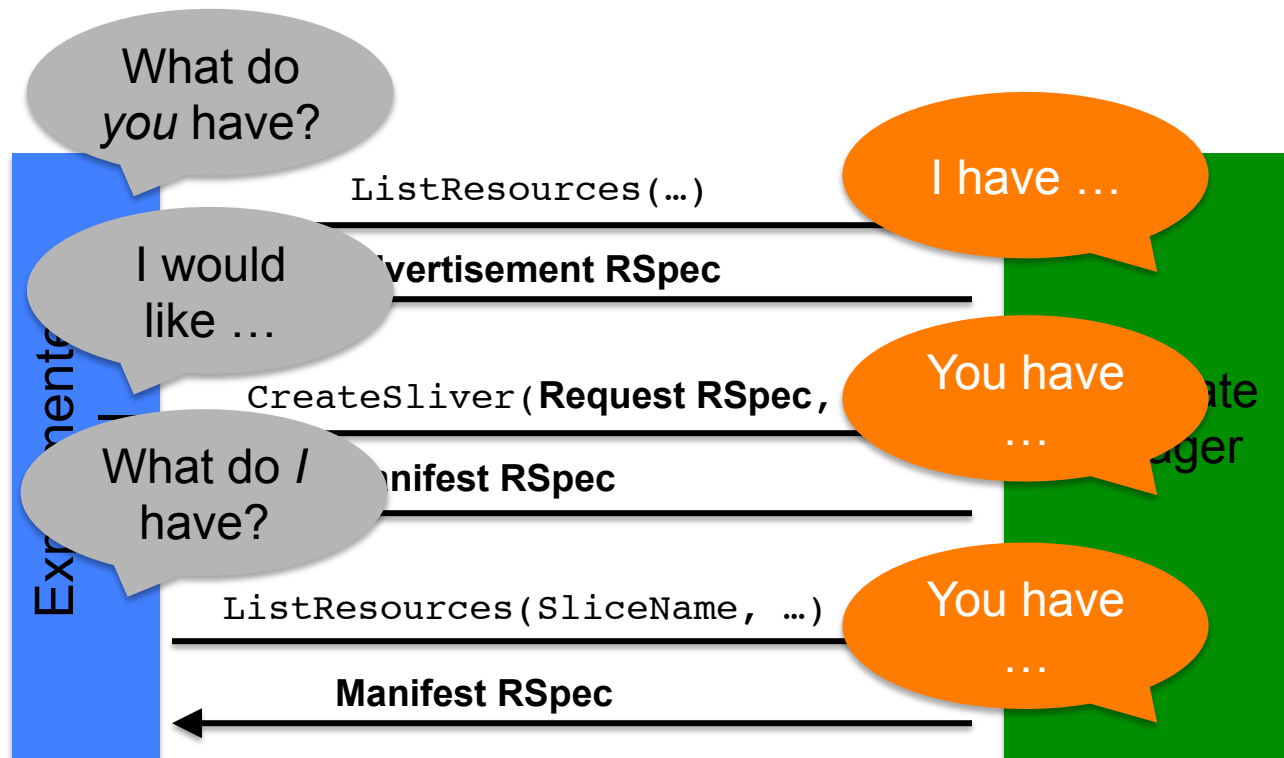
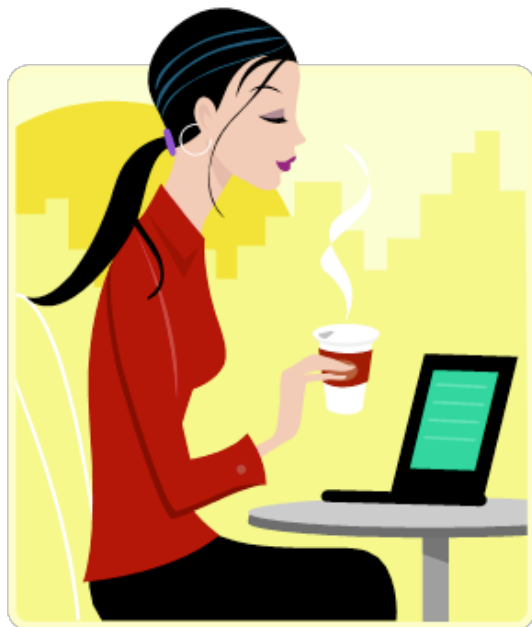
```
<?xml version="1.0" encoding="UTF-8"?>
<rspec type="request" xsi:schemaLocation="http://www.geni.net/
resources/rspec/3 ... xmlns="http://www.geni.net/resources/rspec/3">
  <node client_id="node0" component_manager_id="urn:publicid:IDN
+wall1.ilabt.iminds.be+authority+cm" exclusive="true">
    <sliver_type name="raw-pc"/>
  </node>
</rspec>
```

RSpecs

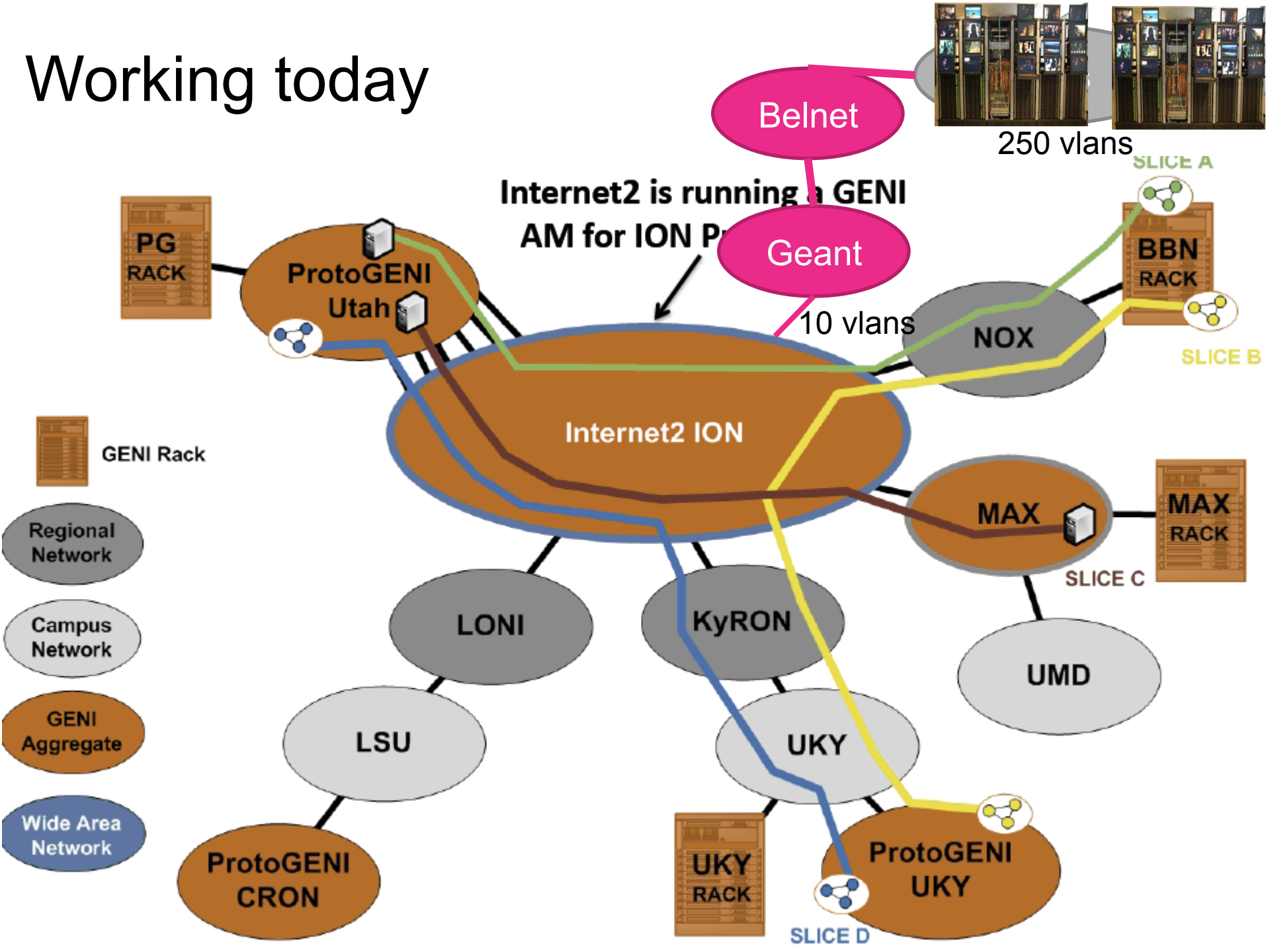
- RSpec documents are exchanged by experimenter tools (e.g. jFed) and testbeds
 - Testbeds use RSpecs to describe what they have – **Advertisement RSpecs**
 - Experimenters use RSpecs to describe the resources they want – **Request RSpecs**
 - Testbeds use RSpecs to describe the resources allocated to an experimenter – **Manifest RSpecs**

The AM API

- Experimenter tools and testbeds talk to each other using the Aggregate Manager API (**AM API**)



Working today



Conclusions

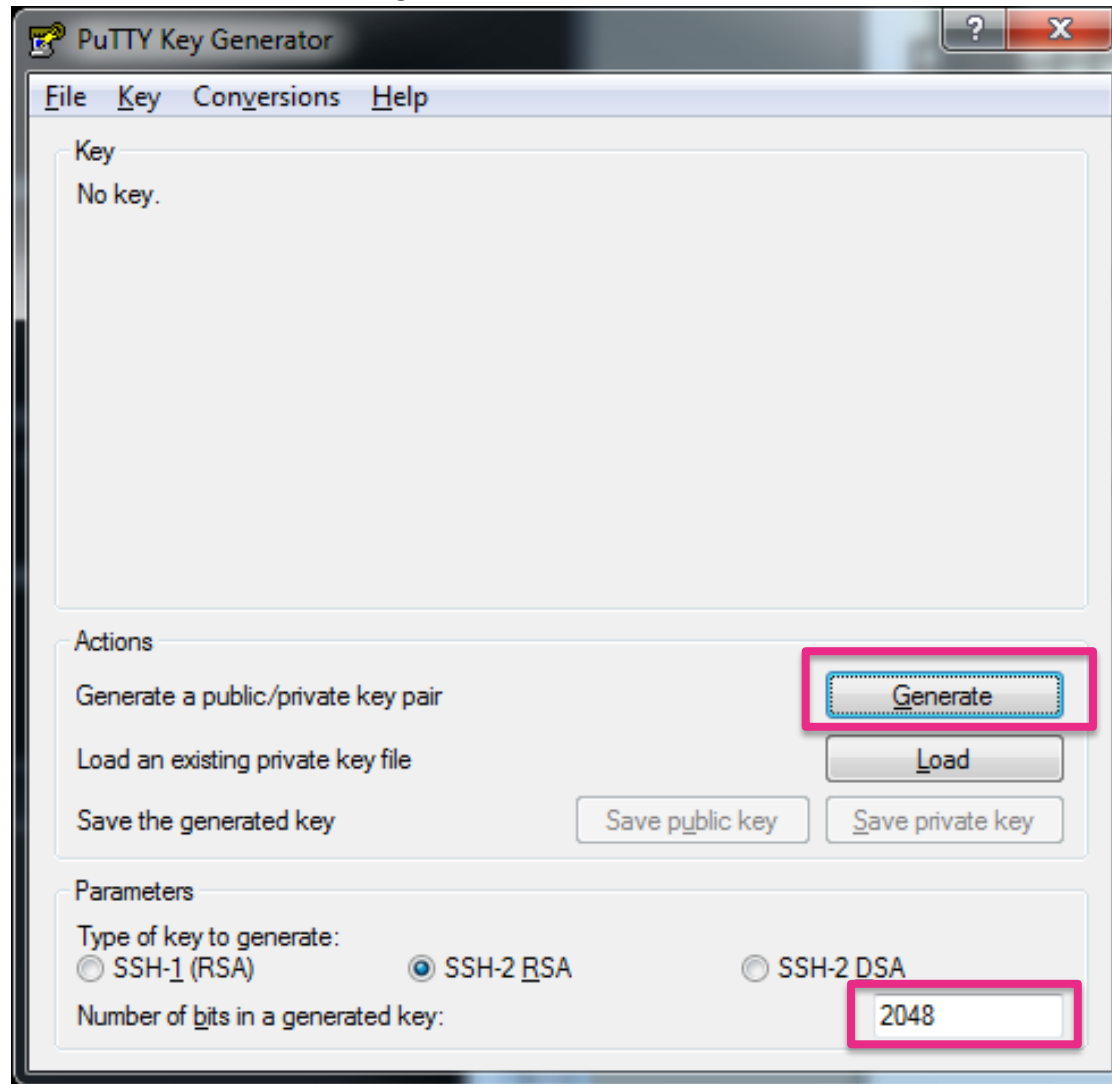
- A wide range of available European testbed facilities circumvent need for time-consuming manual testbed configuration and setup
- Virtual Wall available for iMinds project partners (or get in touch with us for other options)
- Federation of European testbeds: Fed4FIRE: Wide range of wired and wireless testbeds

Executing a basic Virtual Wall experiment using jFed

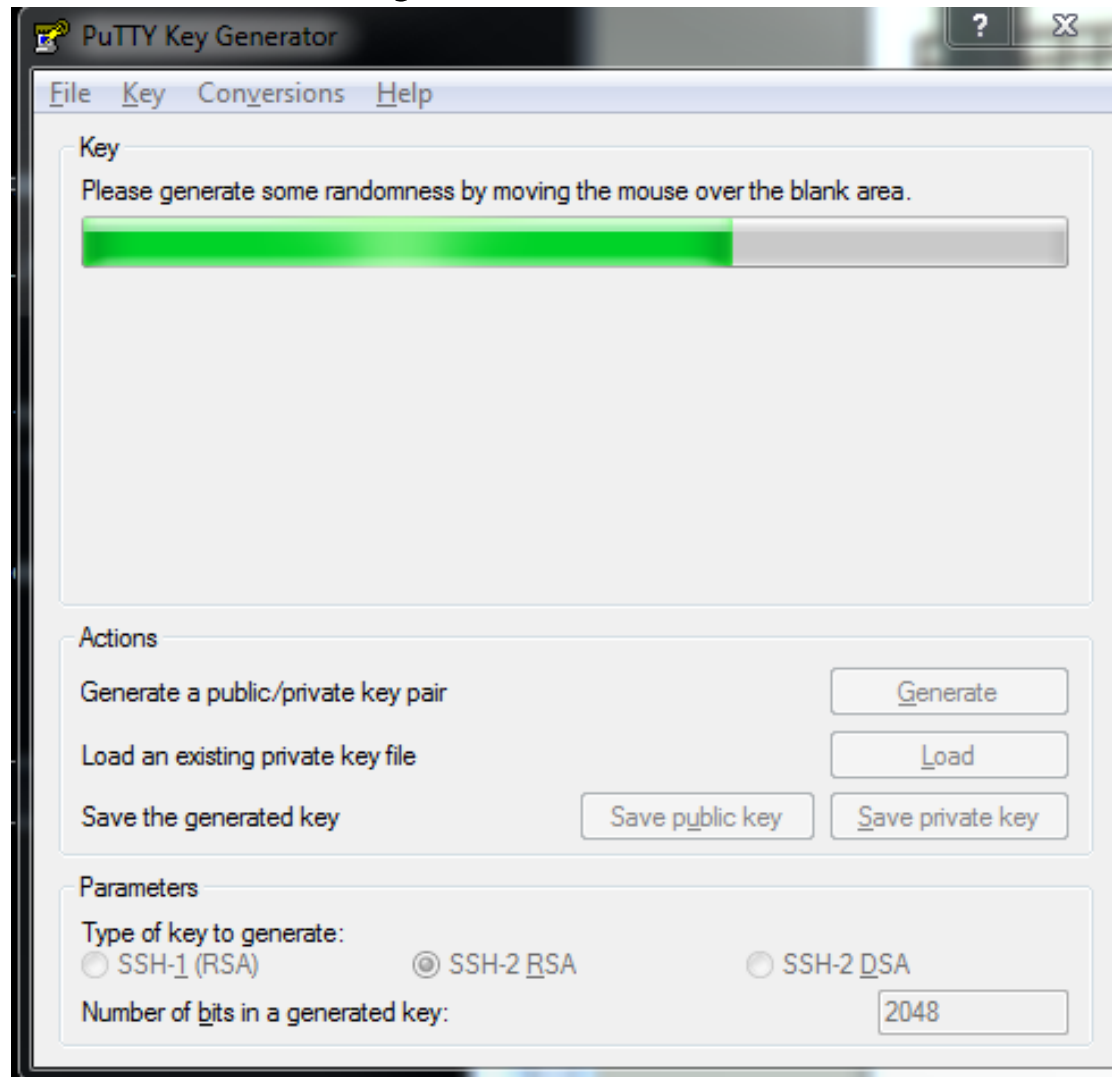
Create SSH key

- Linux/Unix
 - Create SSH key using `ssh-keygen -t rsa`
- Windows
 - Use PuTTYgen to create SSH keys
 - Download from: <http://the.earth.li/~sgtatham/putty/latest/x86/putty-0.63-installer.exe>

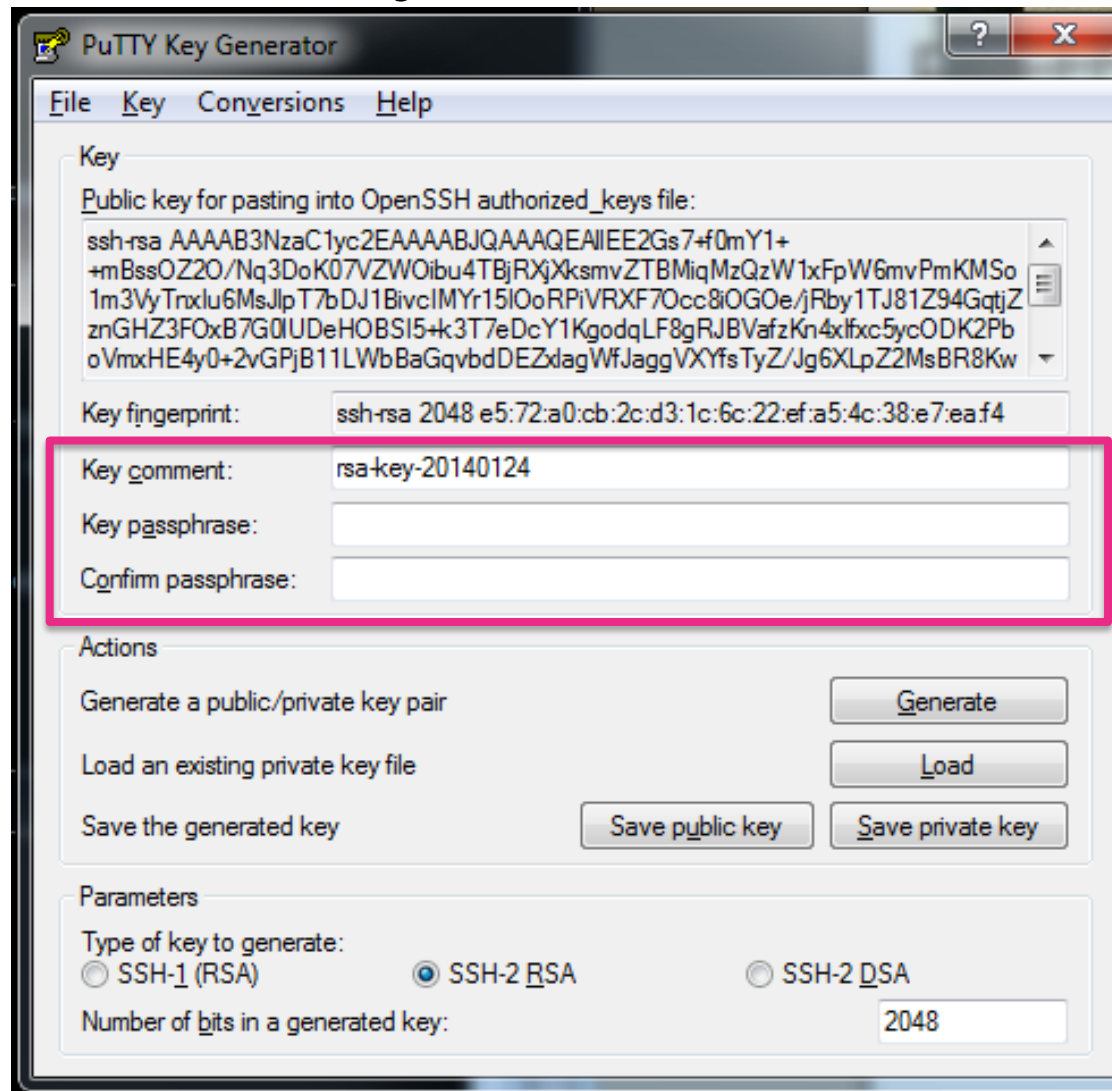
Create SSH key



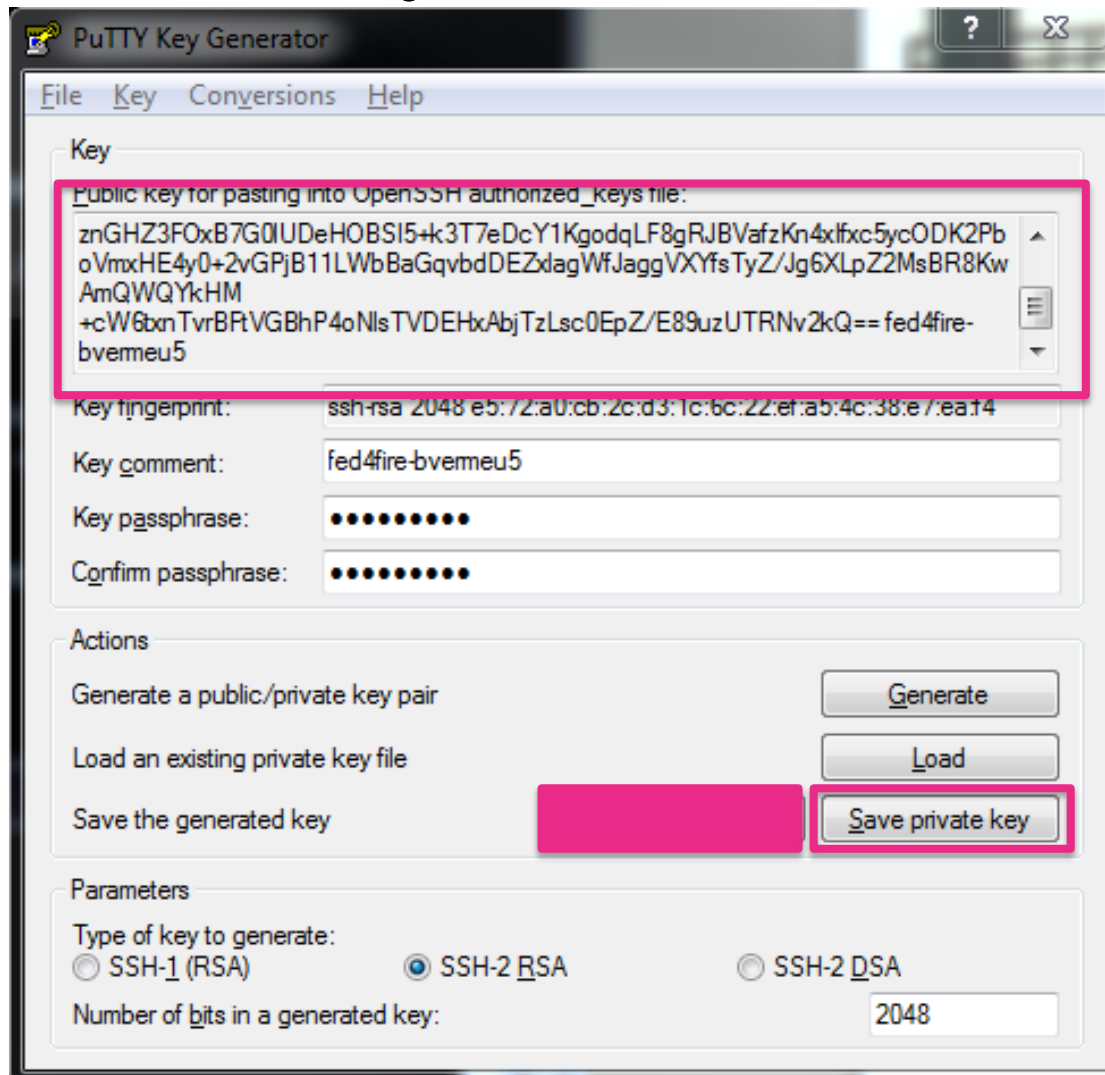
Create SSH key



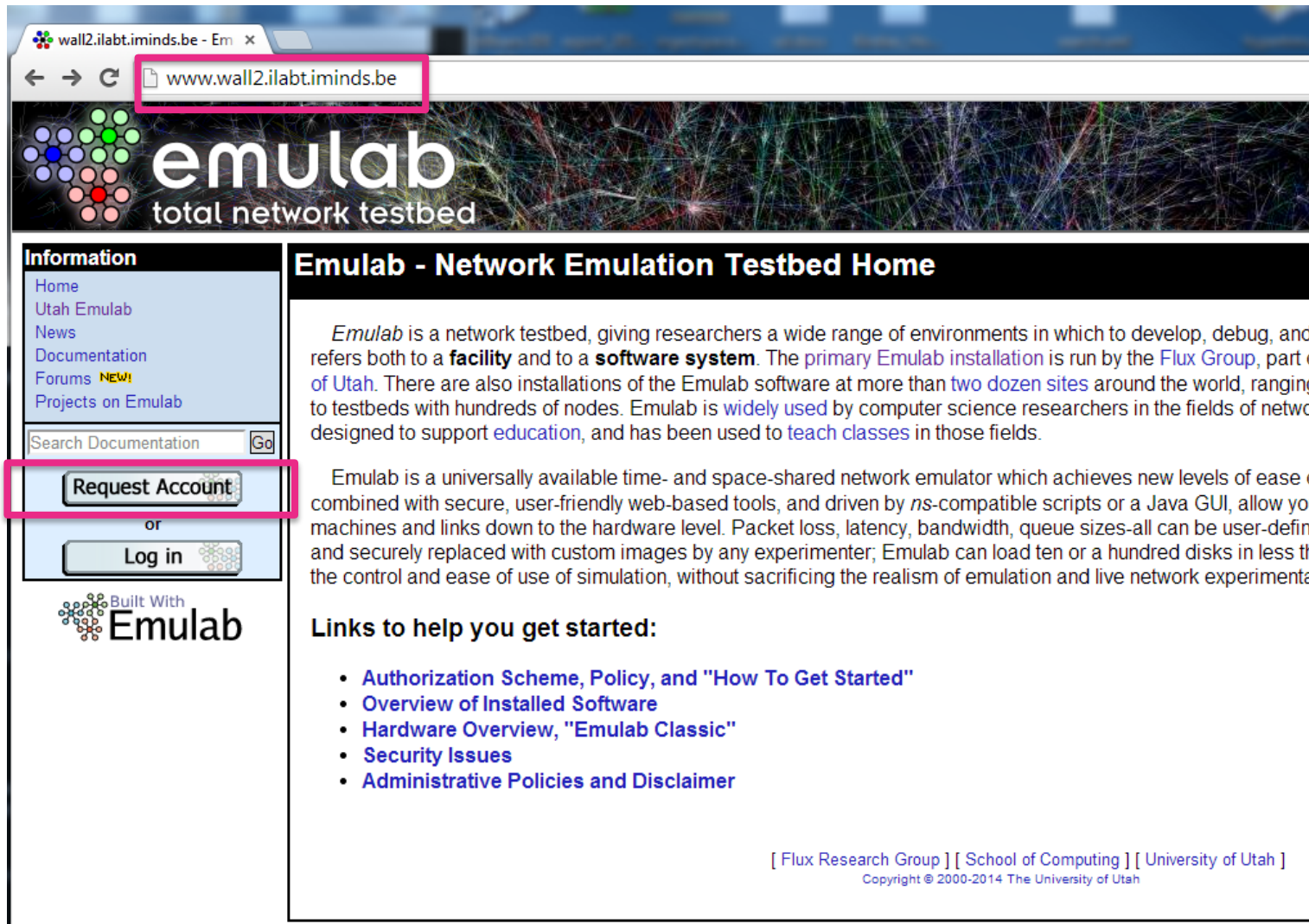
Create SSH key



Create SSH key



Create User



The screenshot shows a web browser window with the URL www.wall2.ilabt.iminds.be in the address bar. The page header features the Emulab logo and the text "total network testbed". The main content area is titled "Emulab - Network Emulation Testbed Home" and contains a paragraph describing Emulab as a network testbed. A sidebar on the left contains a navigation menu with links for Home, Utah Emulab, News, Documentation, Forums (marked as NEW!), and Projects on Emulab. Below the menu is a search box for documentation and a "Request Account" button, which is highlighted with a pink box. Below the "Request Account" button is an "or" separator and a "Log in" button. At the bottom of the sidebar, there is a logo for "Built With Emulab". The main content area also includes a section titled "Links to help you get started:" with a list of links: "Authorization Scheme, Policy, and 'How To Get Started'", "Overview of Installed Software", "Hardware Overview, 'Emulab Classic'", "Security Issues", and "Administrative Policies and Disclaimer". At the bottom right of the page, there is a footer with the text "[Flux Research Group] [School of Computing] [University of Utah]" and "Copyright © 2000-2014 The University of Utah".

Information

- Home
- Utah Emulab
- News
- Documentation
- Forums **NEW!**
- Projects on Emulab

Search Documentation

Request Account

or

Built With **Emulab**

Emulab - Network Emulation Testbed Home

Emulab is a network testbed, giving researchers a wide range of environments in which to develop, debug, and refers both to a **facility** and to a **software system**. The **primary Emulab installation** is run by the **Flux Group**, part of **Utah**. There are also installations of the Emulab software at more than **two dozen sites** around the world, ranging to testbeds with hundreds of nodes. Emulab is **widely used** by computer science researchers in the fields of network design to support **education**, and has been used to **teach classes** in those fields.

Emulab is a universally available time- and space-shared network emulator which achieves new levels of ease of use combined with secure, user-friendly web-based tools, and driven by *ns*-compatible scripts or a Java GUI, allow you to run experiments on real machines and links down to the hardware level. Packet loss, latency, bandwidth, queue sizes-all can be user-defined and securely replaced with custom images by any experimenter; Emulab can load ten or a hundred disks in less than a second. The control and ease of use of simulation, without sacrificing the realism of emulation and live network experiments.

Links to help you get started:

- [Authorization Scheme, Policy, and "How To Get Started"](#)
- [Overview of Installed Software](#)
- [Hardware Overview, "Emulab Classic"](#)
- [Security Issues](#)
- [Administrative Policies and Disclaimer](#)

[Flux Research Group] [School of Computing] [University of Utah]
Copyright © 2000-2014 The University of Utah

Create User

The screenshot shows a web browser window with the URL <https://www.wall2.ilabt.iminds.be/reqaccount.php3>. The page features a header with the Emulab logo and the text "total network testbed". A left sidebar contains navigation links: Home, Utah Emulab, News, Documentation, Forums **NEW!**, and Projects on Emulab. Below these links is a search bar for documentation and two buttons: "Request Account" and "Log in". The main content area is titled "Request a New Emulab Account" and contains the following text: "If you already have an Emulab account, please log on first!". Below this is a button labeled "Join an Existing Project." which is highlighted with a pink border. Underneath the button is the word "or" and the text "Start a New Project.". A note follows: "If you are a student (undergrad or graduate), please do not try to start a project! Your advisor must do it.". At the bottom of the main content area, there are links for "[Flux Research Group]", "[School of Computing]", and "[University of Utah]", along with the copyright notice "Copyright © 2000-2014 The University of Utah".

Create User

Fields marked with * are required.

*Username (alphanumeric):	<input type="text" value="nbouten"/>		
*Full Name (first and last):	<input type="text" value="Niels Bouten"/>		
*Job Title/Position:	<input type="text" value="PhD Student"/>		
*Institutional Affiliation:	Name	<input type="text" value="Ghent University - iMinds"/>	
	Abbreviation:	<input type="text" value="UGent"/>	(e.g. MIT)
Home Page URL:	<input type="text" value="http://intec.ugent.be"/>		
*Email Address[1]:	<input type="text" value="nbouten@intec.ugent.be"/>		
*Postal Address:			
Line 1	<input type="text" value="Gaston Crommenlaan 8"/>		
Line 2	<input type="text"/>		
City	<input type="text" value="Ledeborg"/>	State/Province	<input type="text"/>
ZIP/Postal Code	<input type="text" value="9050"/>	Country	<input type="text" value="Belgium"/>
*Phone #:	<input type="text" value="+32498330533"/>		
Upload your SSH Pub Key[2]: (4K max)	<input type="button" value="Choose File"/> no file selected		
*Password[1]:	<input type="password" value="....."/>		
*Retype Password:	<input type="password" value="....."/>		
Geni Account <small>what's this?</small>			
Geni SSL Pass Phrase[3]:	<input type="password" value="....."/>		
Retype Geni Pass Phrase:	<input type="password" value="....."/>		

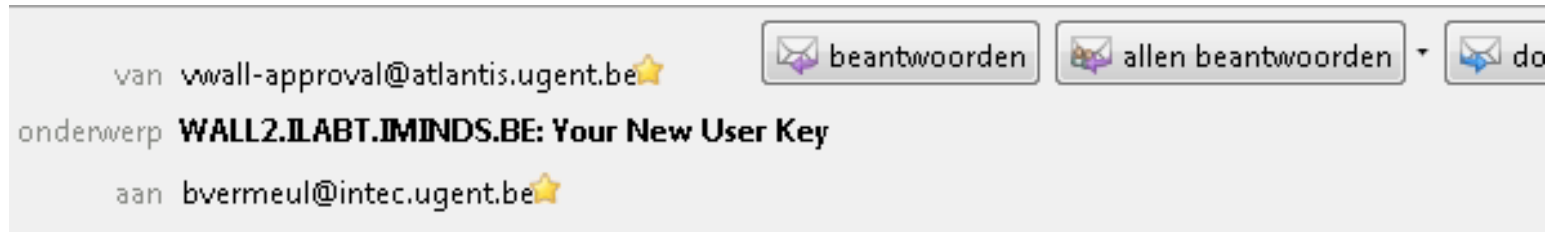
Password for
wall2 portal

Password for
Geni certificate

Create User

Project Info	
*Project Name:	<input type="text" value="AIMSTutorial"/>
Group Name: (Leave blank unless you <i>know</i> the group name)	<input type="text"/>
<input type="button" value="Submit"/>	
(See our Users page)	<input type="text" value="If no, please tell us why not."/>
*Will you add a link on your project page to www.wall2.ilabt.iminds.be ?	<input checked="" type="checkbox"/> Yes
*Funding Sources and Grant Numbers: (Type "none" if not funded)	<input type="text" value="fed4fire"/>
*Estimated #of Project Members[2]:	<input type="text" value="4"/>
*Estimated #of PCs[2]:	<input type="text" value="30"/>
*Please describe how and why you'd like to use the testbed.	
<div style="border: 1px solid #ccc; padding: 5px; min-height: 150px;"><p>open call experiment <u>for</u> Fed4FIRE</p></div>	
<input type="button" value="Submit"/>	

Create User



Dear Brecht Vermeulen (bvermeu5):

This is your account verification key: 81383e282f9bc5191a095f1d729958c2


Please use this link to verify your user account:

<https://www.wall2.ilabt.iminds.be/login.php3?vuid=bvermeu5&key=81383e2>

You will then be verified as a user. When you have been both verified and approved by Testbed Operations, you will be marked as an active user and granted full access to your account. You MUST verify your account before your project can be approved!

Thanks,
Testbed Operations

Create User



My Emulab | Logout | News | Contact Us

Information ▾ Experimentation ▾

42 Free PCs
0 PCs reloading
1 active users
10 active expts.

Confirm Verification

Done!

You have now been verified. However, your application has not yet been approved. You will receive email when that has been done.

[[Flux Research Group](#)] [[School of Computing](#)] [[University of Utah](#)]
Copyright © 2000-2014 The University of Utah

Questions? Join the [Help Forum](#)

Add SSH key to emulab

My Emulab

'aims01' Logged in.
Wed Jun 25 11:06am CEST

Projects Profile

- Options
- Edit Profile
- Edit SSH Keys**
- Generate SSL Cert
- Download your SSL Cert


Username:	aims01 (15712)
Full Name:	AIMS User 1
Email Address:	aims2014tutorial+1@gmail.com
Home Page URL:	http://www.iminds.be
Address 1:	Gaston Crommenlaan 8

Enter ssh public keys for user aims01 [1,2]
(We **strongly** encourage the use of Protocol 2 keys only! [6])

Upload Public Key[3,4]: (4K max)	<input type="button" value="Choose File"/> no file selected
Password[5]:	<input type="password"/>
<input type="button" value="Add New Keys"/>	

Last Web Login:	2014-06-25 11:06:19
Last Users Login:	N/A
Last Node Login:	N/A

Download Fed4FIRE certificate

 My Emulab | Logout | News | Contact Us 42 Free PCs
0 PCs reloading
2 active users
10 active expts.

Information ▾ Experimentation ▾

My Emulab


Projects

Profile

Options

- Edit Profile
- Edit SSH Keys
- Generate SSL Cert
- Download your SSL Cert**

Username:	bvermeu5 (10733)
Full Name:	Brecht Vermeulen
Email Address:	bvermeul@intec.u
Home Page URL:	http://www.iminds.b

 My Emulab | Logout | News | Contact Us 42 Free PCs
0 PCs reloading
2 active users
10 active expts.

Information ▾ Experimentation ▾

Download SSL Certificate for user: bvermeu5

Download your certificate and private key in PEM format, and then save it to a file in your .ssl directory.

You can also download it in *pkc12* format for loading into your web browser (if you do not know what this means, or v

We have also created a SSH key pair for you, derived from your new ssl certificate, using the same pass phrase. You
The private key is typically placed in your .ssh directory on your desktop machine. If you are running an agent such as
those programs.

jFed tool installation

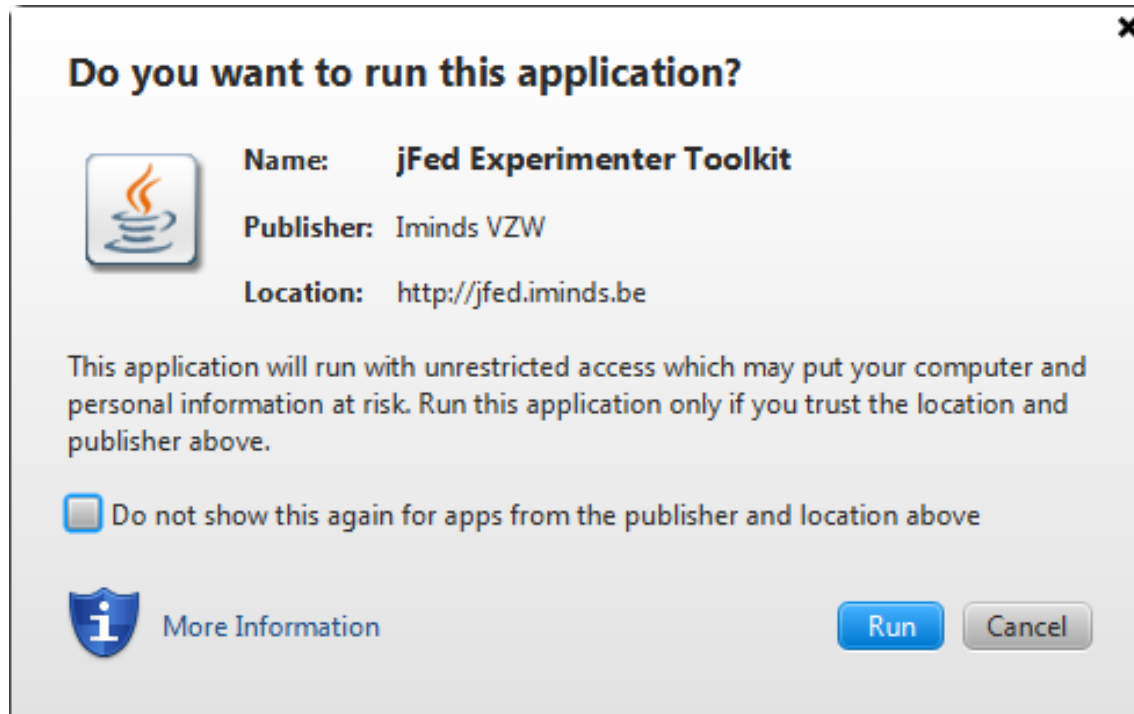
- You need a recent version of Java 7 (<http://www.java.com/verify>)
- Go to <http://jfed.iminds.be>

jFed : Java based framework to support SFA testbed federation client tools

It is released under the MIT license. You need a recent Java 7 install. The green button starts the most recent stable version.

Quickstart jFed experimenter tool

jFed tool installation



For OS X, you may need to change security settings
<http://fed4fire-testbeds.ilabt.iminds.be/jfed-documentation/mac.html>

jFed tool installation

User certificate: /Users/nielsbouten/.ssl/jfed.priv **Brows...**

Username: nbouten

Authority: iMinds Virtual Wall 2

Password:

Login

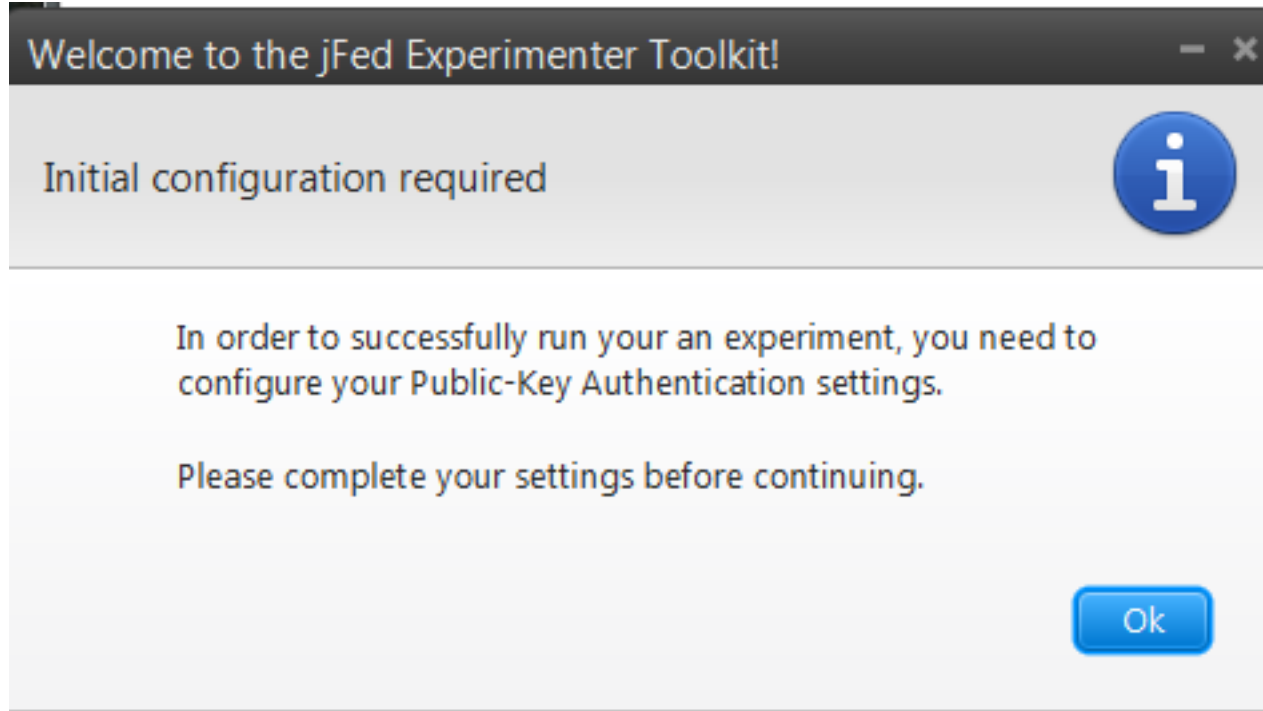
Enter the password associated with the certificate

Connectivity Tester **Advanced login** **Reset jFed**

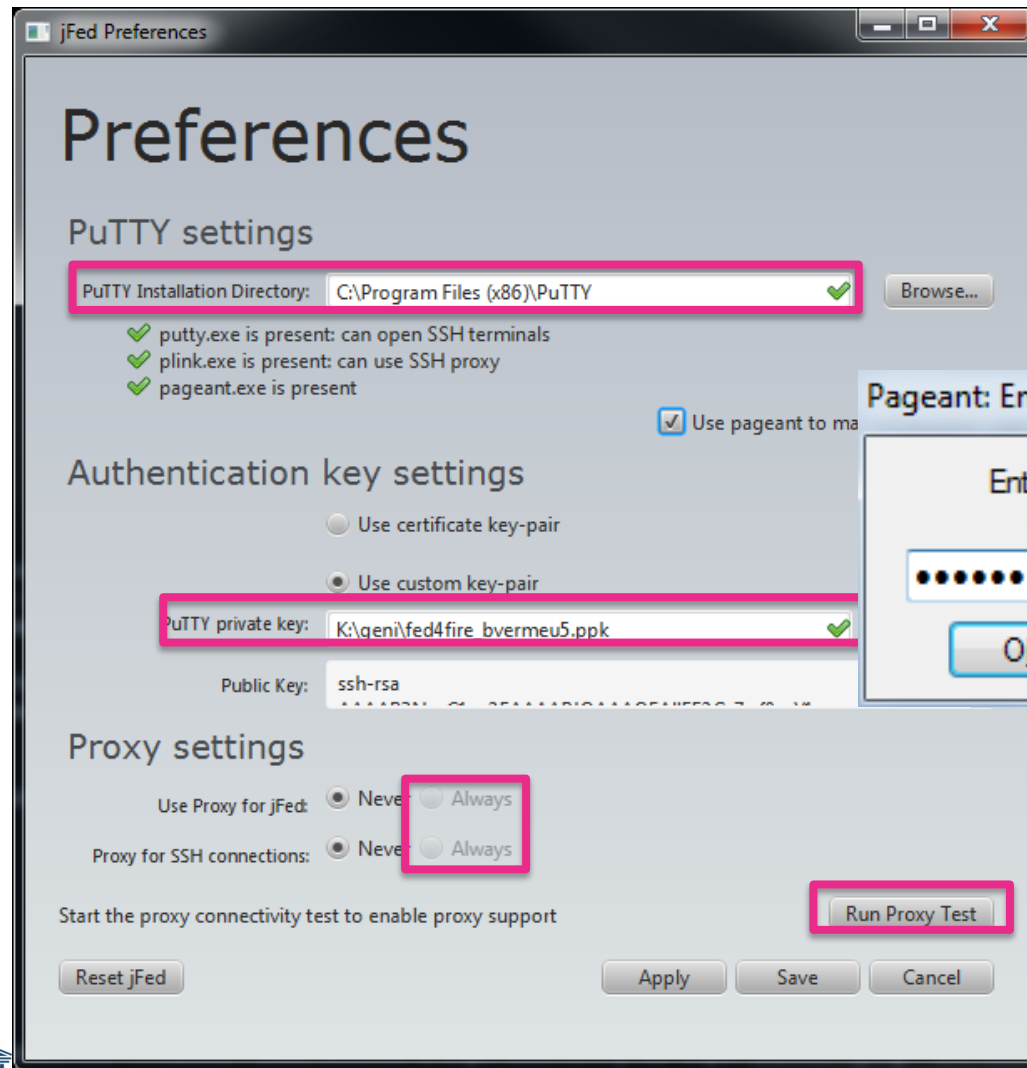
Browse for Geni
.pem file

Enter Geni
password

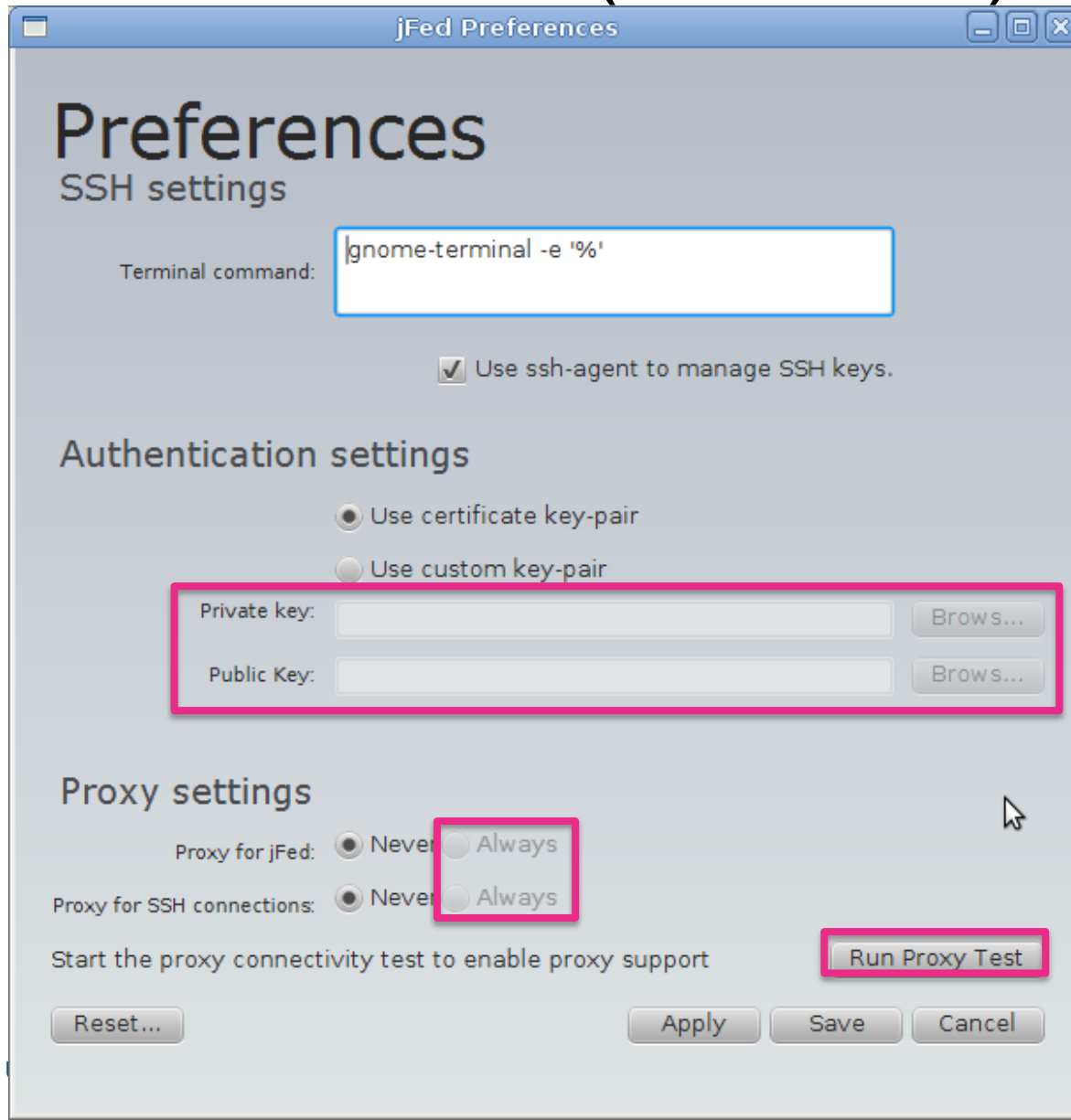
jFed tool installation



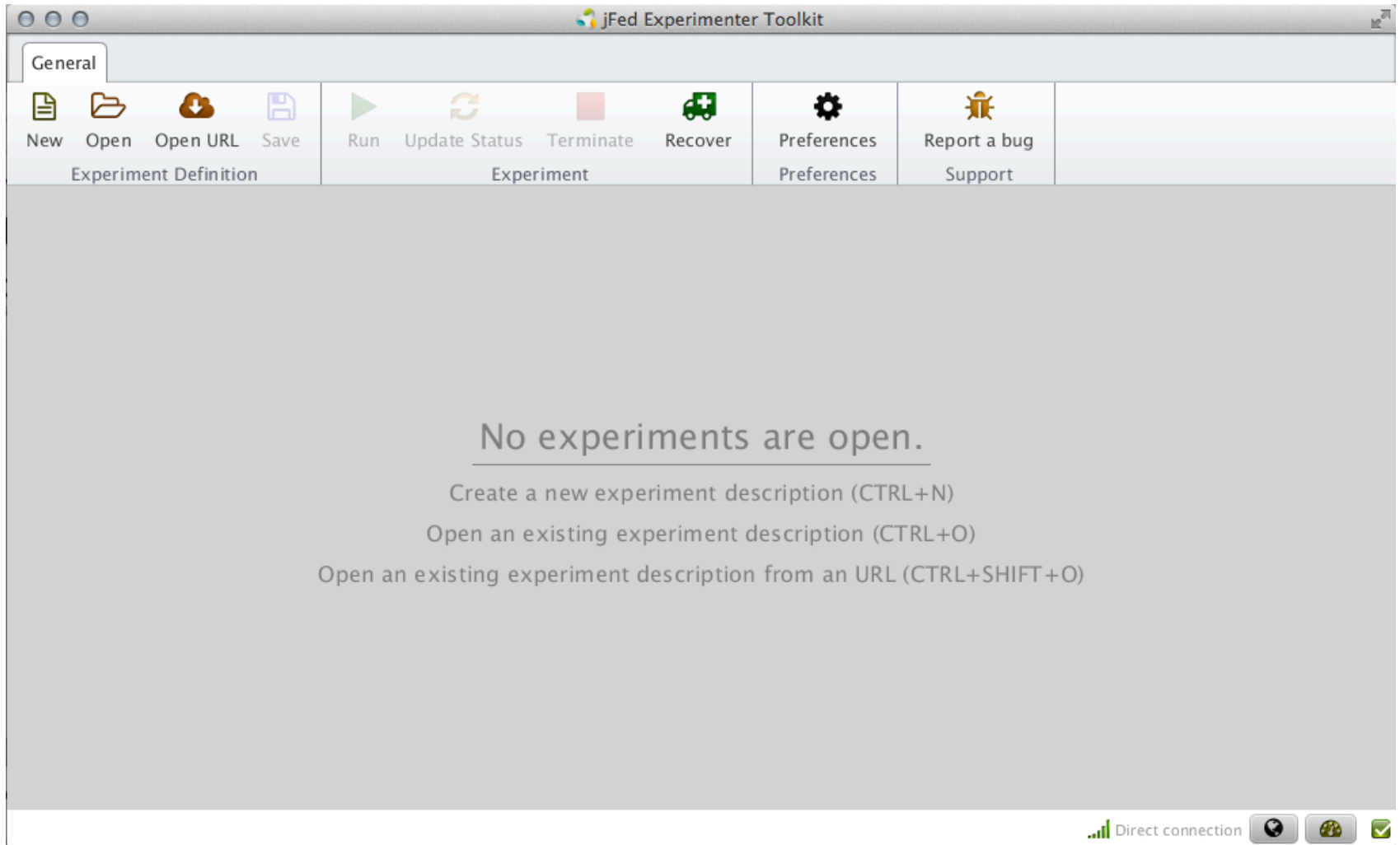
jFed tool installation (Windows)



jFed tool installation (Unix/Mac)



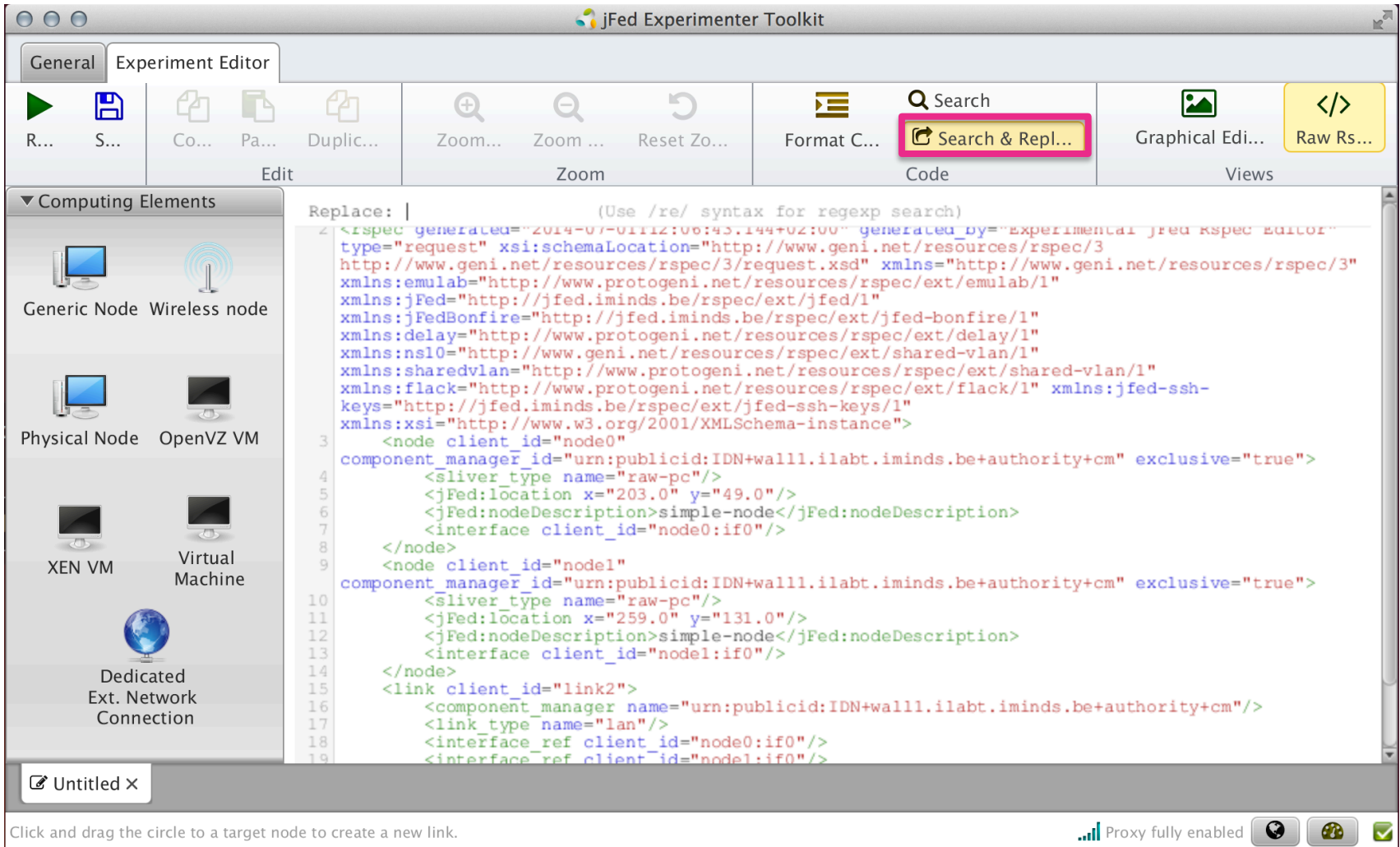
jFed tool



Create Experiment – Select testbed

The screenshot displays the jFed Experimenter Toolkit interface. The main window is titled "jFed Experimenter Toolkit" and has two tabs: "General" and "Experiment Editor". The "Experiment Editor" tab is active, showing a toolbar with icons for Run (R...), Save (S...), Copy (Co...), Paste (Pa...), Duplicate (Duplic...), Zoom In (Zoom...), Zoom Out (Zoom ...), and Reset Zoom (Reset Zo...). Below the toolbar is a "Computing Elements" panel with various node types: Generic Node, Wireless node, Physical Node, OpenVZ VM, XEN VM, Virtual Machine, and Dedicated Ext. Network Connection. The main workspace shows a network diagram with two nodes, "node0" and "node1", connected by a link labeled "link2". A dialog box titled "Properties of node0" is open over the diagram. The dialog has a "General" tab and contains the following fields: "Node name:" with the value "node0"; "Select testb..." with a dropdown menu showing "iMinds Virtual Wall 1" (highlighted with a pink box); and "Select node:" with radio buttons for "Any available node" (selected) and "Specific node:". The "Save" and "Cancel" buttons are at the bottom right of the dialog. At the bottom of the main window, there is a status bar with the text "Click and drag the circle to a target node to create a new link." and system icons including "Proxy fully enabled", a globe, a trash can, and a checkmark.

Create Experiment – Select testbed



The screenshot displays the jFed Experimenter Toolkit interface. The title bar reads "jFed Experimenter Toolkit". The interface is divided into several sections:

- General / Experiment Editor:** Contains icons for actions like Run (R...), Save (S...), Copy (Co...), Paste (Pa...), Duplicate (Duplic...), Zoom In (Zoom...), Zoom Out (Zoom...), Reset Zoom (Reset Zo...), Format Code (Format C...), Search (Search), Search & Replace (Search & Repl...), Graphical Editor (Graphical Edi...), and Raw Resources (Raw Rs...).
- Computing Elements:** A sidebar on the left with icons for:
 - Generic Node
 - Wireless node
 - Physical Node
 - OpenVZ VM
 - XEN VM
 - Virtual Machine
 - Dedicated Ext. Network Connection
- Main Editor:** A text area containing XML code for an experiment. The code includes metadata like `<rspec generated="2014-07-01T12:06:43.144+02:00" generated_by="Experimental jFed RSpec Editor" type="request" xsi:schemaLocation="http://www.geni.net/resources/rspec/3 request.xsd" xmlns="http://www.geni.net/resources/rspec/3" xmlns:emulab="http://www.protopeni.net/resources/rspec/ext/emulab/1" xmlns:jFed="http://jfed.iminds.be/rspec/ext/jfed/1" xmlns:jFedBonfire="http://jfed.iminds.be/rspec/ext/jfed-bonfire/1" xmlns:delay="http://www.protopeni.net/resources/rspec/ext/delay/1" xmlns:ns10="http://www.geni.net/resources/rspec/ext/shared-vlan/1" xmlns:sharedvlan="http://www.protopeni.net/resources/rspec/ext/shared-vlan/1" xmlns:flack="http://www.protopeni.net/resources/rspec/ext/flack/1" xmlns:jfed-ssh-keys="http://jfed.iminds.be/rspec/ext/jfed-ssh-keys/1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">` and defines nodes like `<node client_id="node0" component_manager_id="urn:publicid:IDN+wall1.ilabt.iminds.be+authority+cm" exclusive="true">` and `<node client_id="node1" component_manager_id="urn:publicid:IDN+wall1.ilabt.iminds.be+authority+cm" exclusive="true">`.
- Status Bar:** At the bottom, it says "Click and drag the circle to a target node to create a new link." and "Proxy fully enabled" with a signal strength indicator.

Throughput / Latency / Packet loss

- 2 ways
 - By using tc at both endpoints of the links
 - By using impairment nodes with OpenBSD, configured as a bridge

Link configuration

The screenshot shows the jFed Experimenter Toolkit interface. The main window has tabs for 'General', 'Experiment Editor', and 'Experiment Controller'. The 'Experiment Editor' tab is active, showing a toolbar with icons for New, Open, Open URL, Save, Run, Stop, Add, Settings, and Help. Below the toolbar is the 'Experiment Definition' section, which includes a 'Computing Elements' panel with icons for Generic Node, Wireless node, Physical Node, OpenVZ VM, XEN VM, and Virtual Mac... A 'Properties of link2' dialog box is open, showing the following configuration:

Link name: link2

Impairment: Simple link impairment

Capacity: 100000 Kbits/s

Latency: 20 ms

Packet Loss: 0.01 %

Advanced link impairment

From	To	Capacity (K...	Latency (ms)	Packet Loss...
node1:if0	node0:if0	100,000	20	0.01
node0:if0	node1:if0	100,000	20	0.01

Buttons: Save, Cancel

Bottom status bar: Click and drag the circle to a target node to create a new link. Direct connection

Adapt settings link

- tc script on each node located at

```
% /var/emulab/boot/rc.linkdelay
```

```
modprobe sch_netem
#!/bin/sh
modprobe ifb numifbs=10
sysctl -w net.core.rmem_max=8388608
sysctl -w net.core.wmem_max=8388608
sysctl -w net.core.netdev_max_backlog=2048
ifconfig eth1 txqueuelen
/sbin/tc qdisc del dev eth1 root
/sbin/tc qdisc del dev eth1 ingress
/sbin/tc qdisc add dev eth1 handle 130 root htb default 1
/sbin/tc class add dev eth1 classid 130:1 parent 130 htb rate 100000000 ceil 100000000
/sbin/tc qdisc add dev eth1 handle 120 parent 130:1 netem drop 0 delay 20000us
echo "Delay Configuration Complete"
```

- Modify settings

```
% sudo /sbin/tc class change ...
```

```
% sudo /sbin/tc qdisc change ...
```

Bridge configuration

The screenshot displays the jFed Experimenter Toolkit interface. The window title is "jFed Experimenter Toolkit". The interface is divided into several sections:

- General**: Contains icons for New, Open, Open URL, and Save.
- Experiment Editor**: Contains icons for Run, Update Status, Terminate, and Recover.
- Experiment Controller**: Contains icons for Preferences and Report a bug.

The main workspace shows a network configuration diagram on a grid background. The diagram consists of the following elements:

- node0**: A blue rectangular node at the top.
- link4**: A blue rectangular link connecting node0 to bridge0.
- bridge0**: A yellow rectangular bridge node in the center.
- link3**: A blue rectangular link connecting bridge0 to node1.
- node1**: A blue rectangular node at the bottom.

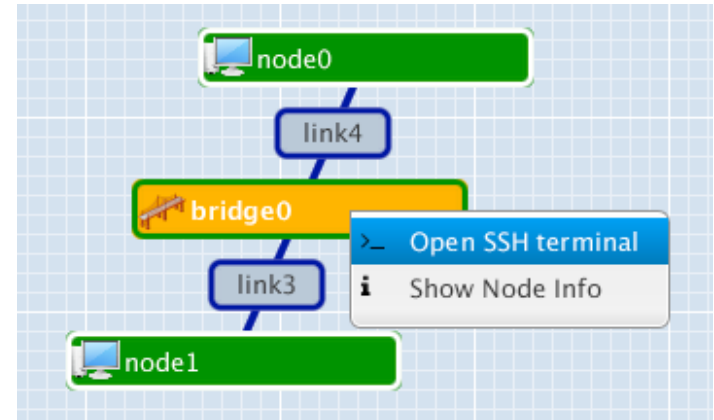
On the left side, there is a **Computing Elements** panel with the following options:

- Generic Node
- Wireless node
- Physical Node
- OpenVZ VM
- XEN VM
- Virtual Mac...

At the bottom of the interface, there is a status bar with the text "Click and drag the circle to a target node to create a new link." and a "Direct connection" indicator with a signal strength icon and a green checkmark.

Adapt settings bridge

- Login to bridge using jFed



- List settings

```
% sudo ipfw pipe show
```

- Adjust settings

```
% sudo ipfw pipe nr config bw 10Mbit/s  
plr 0.0 delay 40ms
```

Advanced use with RSpec

Install Specific Disk Image

```
<sliver_type name="raw-pc">
  <disk_image name="urn:publicid:IDN+wall2.ilabt.iminds.be+image+emulab-ops//DEB60_64-VLAN"/>
</sliver_type>
```

Install Software when provisioning, run script after provisioning

```
<sliver_type name="raw-pc"/>
<services>
  <execute command="sudo apt-get install dstat" shell="sh"/>
  <install install_path="/local" url="http://download.videolan.org/pub/videolan/vlc/0.5.1/vlc-0.5.1.tar.gz"/>
</services>
```

Specify link configurations

```
<link client_id="link2">
  <component_manager name="urn:publicid:IDN+wall1.ilabt.iminds.be+authority+cm"/>
  <link_type name="lan"/>
  <property source_id="node1:if0" dest_id="node0:if0" capacity="10000" latency="20" packet_loss="0.01"/>
  <property source_id="node0:if0" dest_id="node1:if0" capacity="20000" latency="20" packet_loss="0.01"/>
  <interface_ref client_id="node1:if0"/>
  <interface_ref client_id="node0:if0"/>
</link>
```

Shared Storage (per testbed)

- Shared folder of nodes is available via

`/groups/wall2-ilabt-iminds-be/projectname`

- This share is mounted on all nodes automatically
- This share is permanent, data stored here will remain available when experiment is done
- This share is shared with all people in the project

Advanced use Alpha jFed

Alpha jFed experimenter tool

- Support for command timeline
- Add barriers to command execution
- Execute instant commands

Create timed commands

The screenshot displays the jFed Experimenter Toolkit interface. The main window is titled "jFed Experimenter Toolkit" and has a menu bar with "General" and "Experiment Editor". The toolbar includes icons for Run, Save, Copy, Paste, Duplicate, Zoom In, Zoom Out, Reset Zoom, Format Code, Search, Add Command (highlighted with a red box), Add barrier, Graphical Editor, Raw Rspec, and Experiment Controller. The main workspace shows a timeline from 00:00 to 00:32:00 with a grid. A sidebar on the left shows a tree view with "node0", "Barrier segment 2", "node1", and "Barrier segment 2". A dialog box titled "Add command" is open in the center, containing the following fields and options:

- Command tag:
- Bash command:
- Starting time:
- Select nodes:
 - Your experiment
 - Barrier segment 2
 - node0
 - node1

At the bottom of the dialog are "Ok" and "Cancel" buttons. The status bar at the bottom of the main window shows "Click and drag the circle to a target node to create a new link." and "Direct connection" with various system icons.

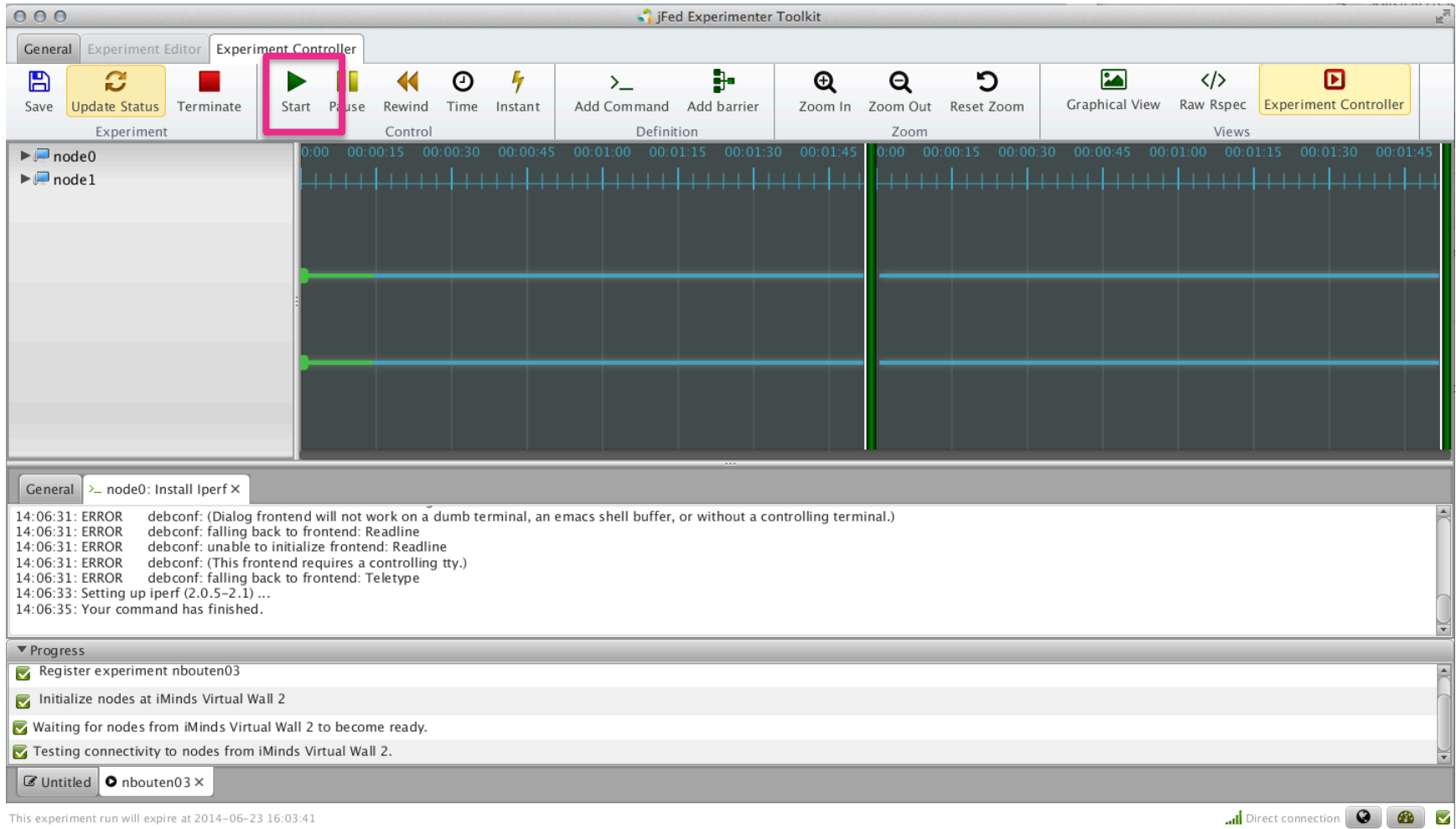
Add Barriers

The screenshot displays the jFed Experimenter Toolkit interface. The title bar reads "jFed Experimenter Toolkit". The main window is divided into several sections:

- General / Experiment Editor:** Contains various icons for Run, Save, Copy, Paste, Duplicate, Zoom In, Zoom Out, Reset Zoom, Format Code, Search, Search & Replace, Add Command, Add barrier (highlighted with a pink box), Graphical Editor, Raw RSpec, and Experiment Controller.
- Left Panel:** A tree view showing a hierarchy of nodes and segments. Under "node0", there are "Barrier segment 2" and "Barrier segment 3". Under "node1", there are also "Barrier segment 2" and "Barrier segment 3".
- Main Area:** A large dark blue workspace with a horizontal timeline at the top. The timeline is marked with time intervals from 00:00 to 00:16:00. Two horizontal blue lines are visible in the workspace, representing barriers. A vertical red line is also present, likely indicating the current time or a specific barrier position.
- Bottom Bar:** Shows "Untitled x" and a status bar with "Direct connection" and several small icons.

Click and drag the circle to a target node to create a new link.

Start execution and follow output



The screenshot displays the jFed Experimenter Toolkit interface. The top toolbar includes buttons for Save, Update Status, Terminate, Start (highlighted with a red box), Pause, Rewind, Time, Instant, Add Command, Add barrier, Zoom In, Zoom Out, Reset Zoom, Graphical View, Raw Rspec, and Experiment Controller. The main workspace is divided into three panes: Experiment (listing node0 and node1), Definition (a timeline with a vertical green line at 00:01:30), and Views (a zoomed-in view of the timeline). The bottom panel shows a terminal window with error messages and a progress list. The terminal output includes:

```
14:06:31: ERROR debconf: (Dialog frontend will not work on a dumb terminal, an emacs shell buffer, or without a controlling terminal.)
14:06:31: ERROR debconf: falling back to frontend: Readline
14:06:31: ERROR debconf: unable to initialize frontend: Readline
14:06:31: ERROR debconf: (This frontend requires a controlling tty.)
14:06:31: ERROR debconf: falling back to frontend: Teletype
14:06:33: Setting up iperf (2.0.5-2.1) ...
14:06:35: Your command has finished.
```

The progress list shows the following steps:

- Register experiment nbouten03
- Initialize nodes at iMinds Virtual Wall 2
- Waiting for nodes from iMinds Virtual Wall 2 to become ready.
- Testing connectivity to nodes from iMinds Virtual Wall 2.

The bottom status bar indicates the experiment run will expire at 2014-06-23 16:03:41 and shows a direct connection status.

Execute instant commands

The screenshot displays the jFed Experimenter Toolkit interface. The main window is titled "jFed Experimenter Toolkit" and has three tabs: "General", "Experiment Editor", and "Experiment Controller". The "Experiment Controller" tab is active, showing a toolbar with various control buttons. The "Instant" button, which features a lightning bolt icon, is highlighted with a pink rectangular box. Below the toolbar, there is a list of nodes: "node0" and "node1". A central dialog box titled "Instant command" is open, containing the following fields and options:

- Command tag:
- Bash command:
- Select nodes: A tree view showing "Your experiment" expanded with "node0" (checked) and "node1" (unchecked).

At the bottom of the dialog are "Ok" and "Cancel" buttons. The background interface shows a timeline at the top with markers from 00:00 to 00:01:45. Below the dialog, a terminal window displays the following log output:

```
14:06:31: ERROR debconf: (Dialog frontend w
14:06:31: ERROR debconf: falling back to fro
14:06:31: ERROR debconf: unable to initializ
14:06:31: ERROR debconf: (This frontend req
14:06:31: ERROR debconf: falling back to frontend: Teletype
14:06:33: Setting up iperf (2.0.5-2.1) ...
14:06:35: Your command has finished.
```

Below the terminal is a "Progress" section with four items, all marked with green checkmarks:

- Register experiment nbouten03
- Initialize nodes at iMinds Virtual Wall 2
- Waiting for nodes from iMinds Virtual Wall 2 to become ready.
- Testing connectivity to nodes from iMinds Virtual Wall 2.

The bottom status bar shows "This experiment run will expire at 2014-06-23 16:03:41" and "Direct connection" with several status icons.

Execute Instant commands

The screenshot displays the jFed Experimenter Toolkit interface. The main window is titled "jFed Experimenter Toolkit" and has three tabs: "General", "Experiment Editor", and "Experiment Controller". The "Experiment Controller" tab is active, showing a toolbar with buttons for "Save", "Update Status", "Terminate", "Start", "Pause", "Rewind", "Time", "Instant", "Add Command", "Add barrier", "Zoom In", "Zoom Out", "Reset Zoom", "Graphical View", "Raw Rspec", and "Experiment Controller".

The "Instant command" dialog box is open, titled "Execute a command instantly and off the record on the selected nodes". It contains the following fields and options:

- Command tag: Start iperf
- Bash command: iperf -c node0
- Select nodes: A tree view showing "Your experiment" expanded with "node0" (unchecked) and "node1" (checked).

The background interface shows a list of nodes (node0, node1) on the left, a timeline at the top right, and a terminal window at the bottom left displaying the following output:

```
14:07:13: Your command has started.
14:07:13: -----
14:07:13: Server listening on TCP port 5001
14:07:13: TCP window size: 85.3 KByte (default)
14:07:13: -----
```

The bottom of the interface shows a "Progress" section with four checked items:

- Register experiment nbouten03
- Initialize nodes at iMinds Virtual Wall 2
- Waiting for nodes from iMinds Virtual Wall 2 to become ready.
- Testing connectivity to nodes from iMinds Virtual Wall 2.

The status bar at the bottom indicates "This experiment run will expire at 2014-06-23 16:03:41" and "Direct connection".

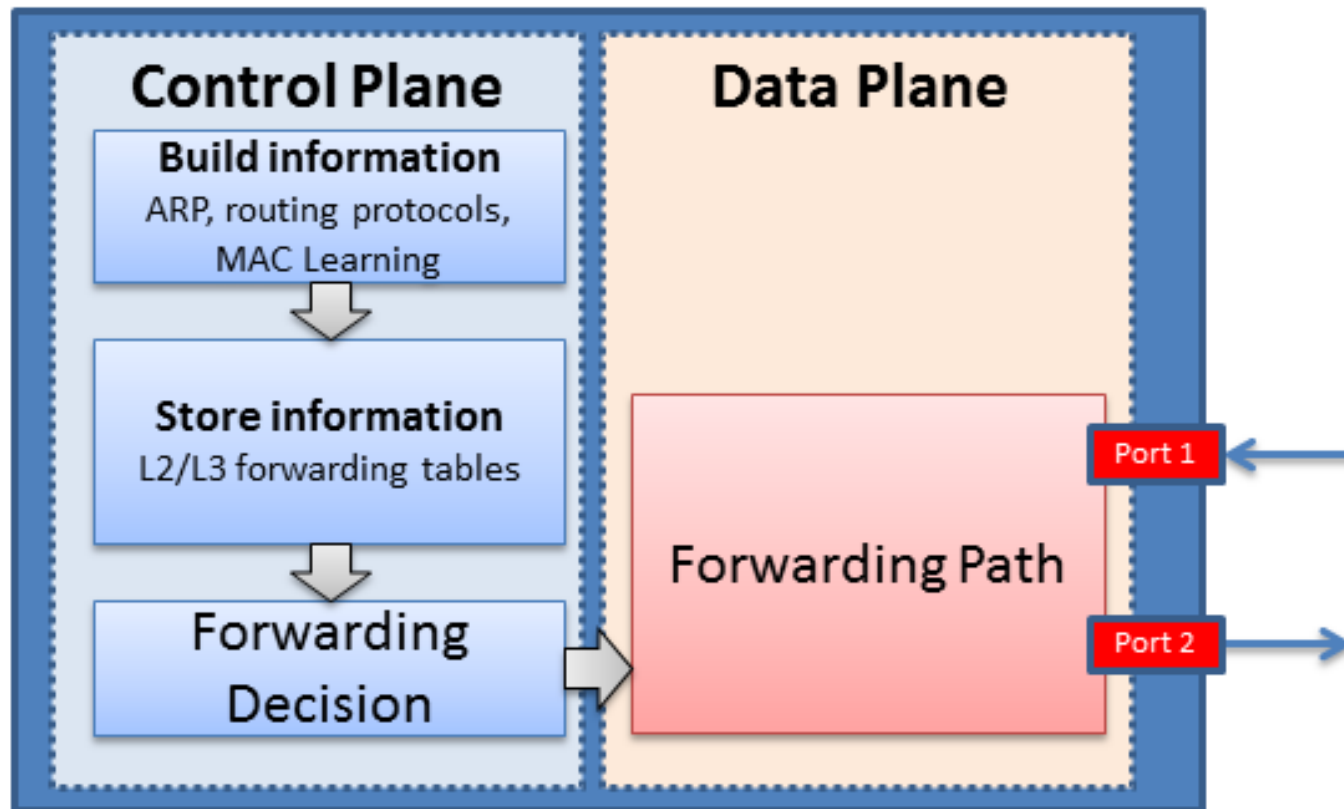
Follow output of instant commands

The screenshot displays the jFed Experimenter Toolkit interface. The top menu bar includes 'General', 'Experiment Editor', and 'Experiment Controller'. The 'Experiment Controller' tab is active, showing a toolbar with icons for Save, Update Status, Terminate, Start, Pause, Rewind, Time, Instant, Add Command, Add barrier, Zoom In, Zoom Out, Reset Zoom, Graphical View, Raw Rspec, and Experiment Controller. The main workspace is divided into three sections: 'Experiment' (left sidebar showing 'node0' and 'node1'), 'Control' (top timeline with a vertical green playhead at 00:01:45), and 'Definition' (middle area). The 'Views' section on the right shows a zoomed-in view of the timeline. Below the workspace, the 'General' tab is active, displaying a list of commands: 'node0: Install iperf X', 'node0: start iperf X', and 'node1: Start iperf X'. The command output for 'node0: start iperf X' is visible, showing a client connection to node0 on TCP port 5001, a TCP window size of 23.5 KByte, and a successful connection to 192.168.1.2 on port 5001. The output also shows the interval, transfer, and bandwidth of the connection: '14:07:53: [3] 0.0-10.0 sec 1.10 GBytes 944 Mbits/sec'. The 'Progress' section shows a list of tasks: 'Register experiment nbouten03', 'Initialize nodes at iMinds Virtual Wall 2', 'Waiting for nodes from iMinds Virtual Wall 2 to become ready.', and 'Testing connectivity to nodes from iMinds Virtual Wall 2.'. The bottom status bar indicates 'This experiment run will expire at 2014-06-23 16:03:41' and shows a 'Direct connection' status.

OpenFlow protocol basics

Traditional network switch

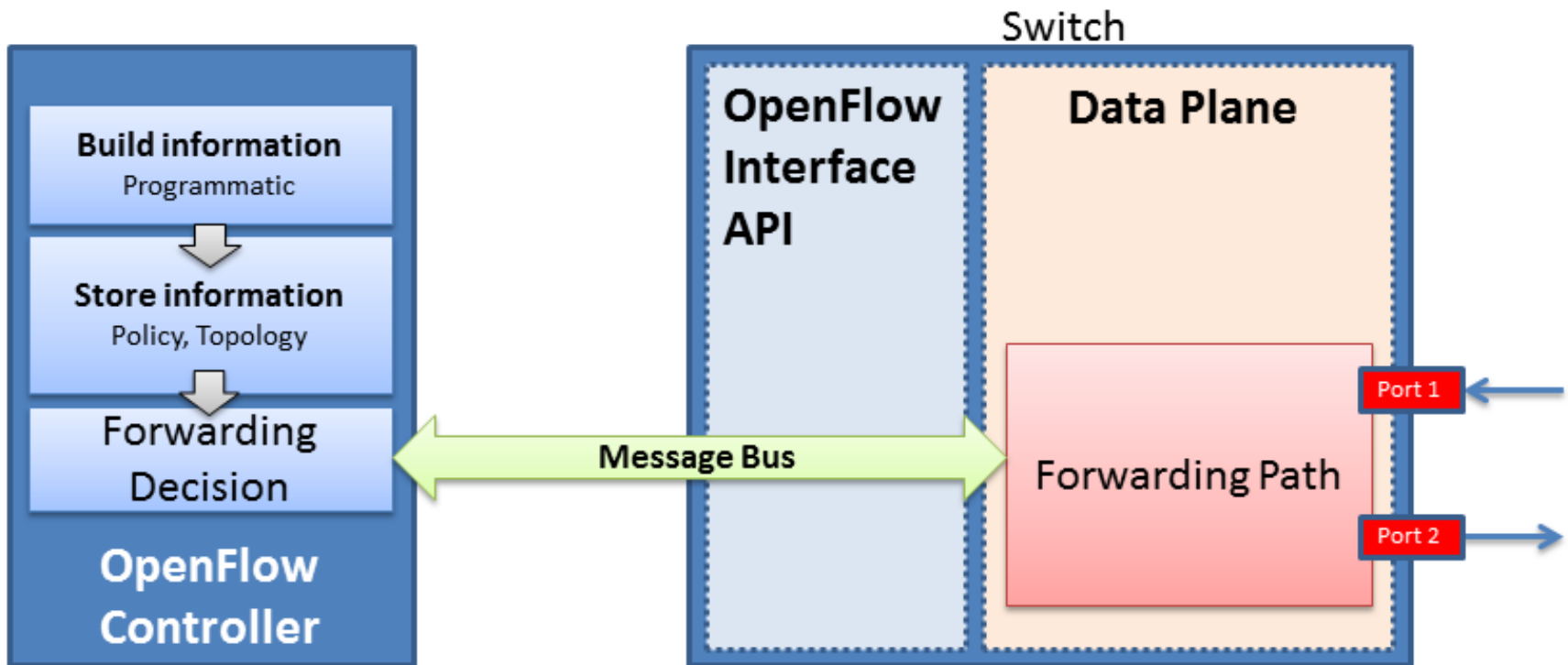
Switch



BRAD HEDLUND .com

OpenFlow-enabled switch

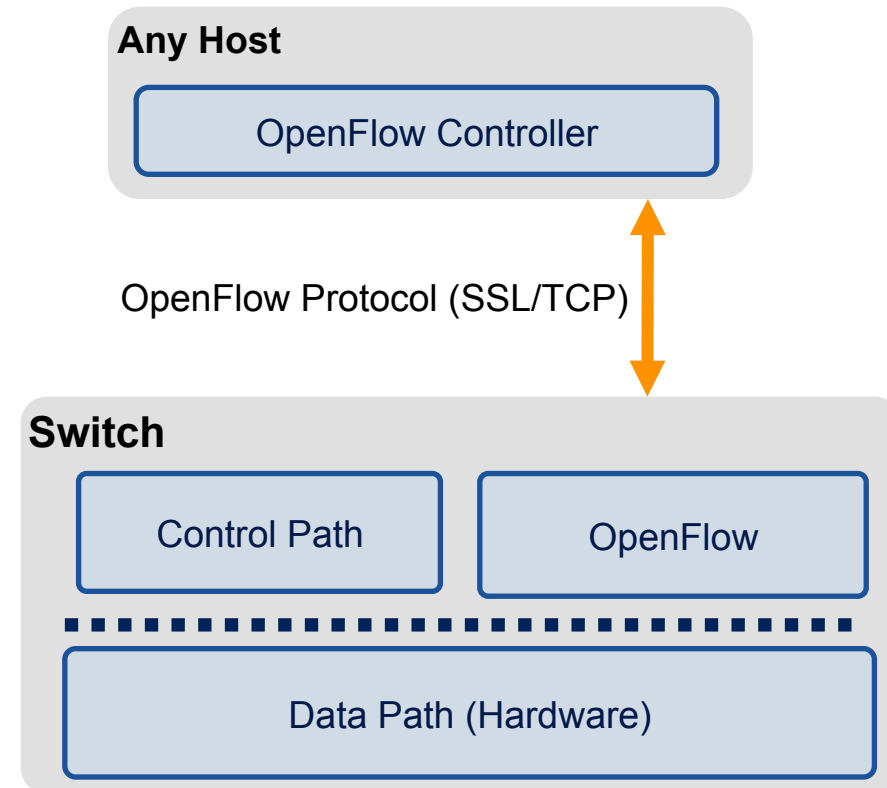
Externally controlled Switch



BRAD HEDLUND .com

The OpenFlow protocol and controller

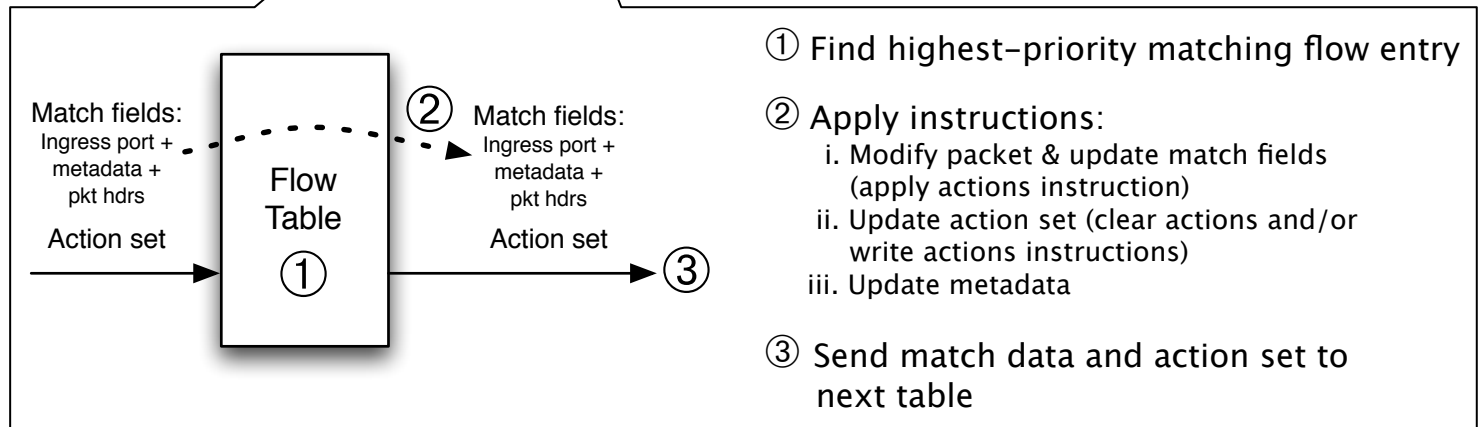
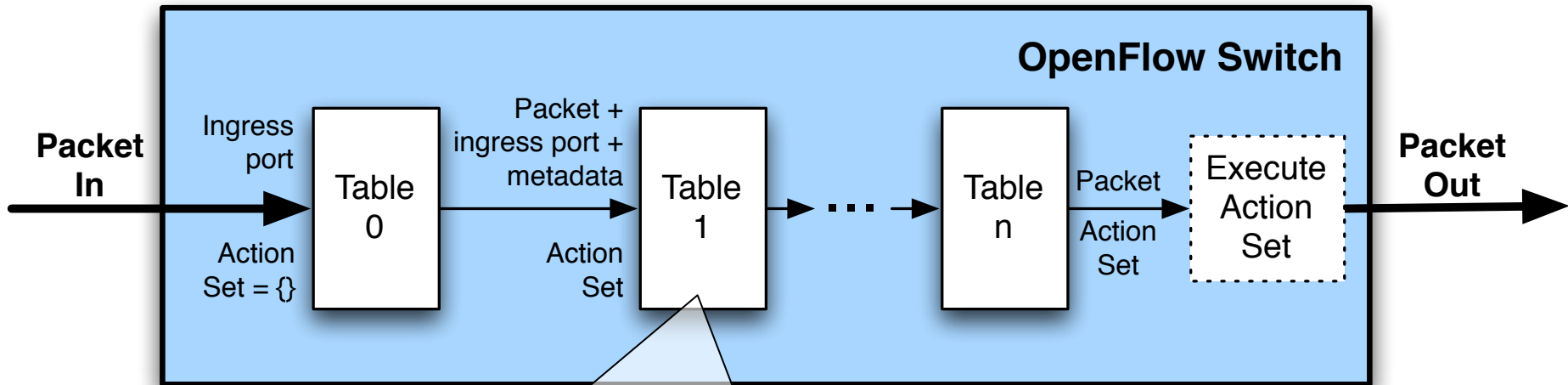
- Controller is responsible for populating flow table of the switch
- In case of a table miss the switch can, for example
 - Forward packet to controller
 - Drop the packet



OpenFlow ports

- Network interfaces for passing packets to and from OpenFlow switch
- Packets enter the switch through ingress ports
 - Ingress ports can be used to match packets
- Packets leave the switch through output ports
 - Packets can be sent to specific output port as action
- Several port types available
 - Physical: Correspond to specific hardware interface
 - Logical: e.g., tunnels, aggregation groups, loopback
 - Reserved: e.g., controller, flooding, normal switching

OpenFlow tables



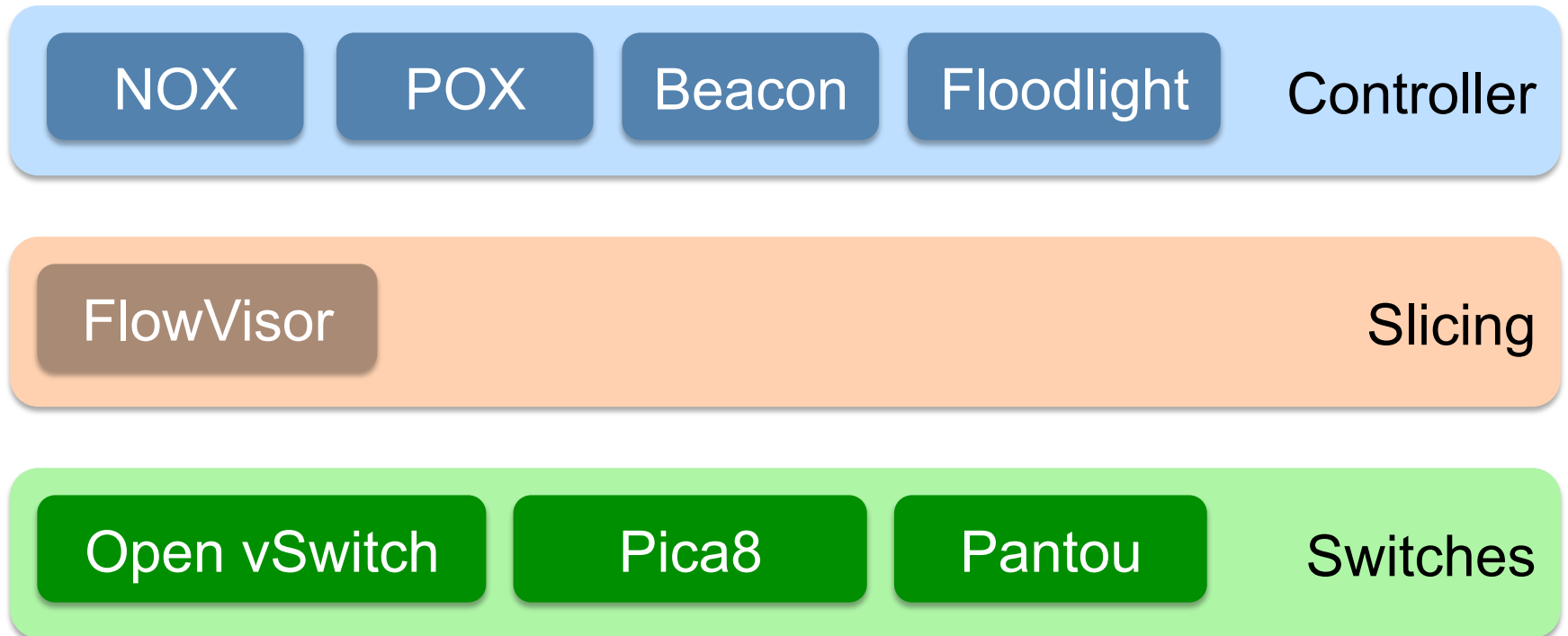
OpenFlow table entries

Match Fields	Priority	Counters	Instructions	Timeouts	Cookie	Flags
--------------	----------	----------	--------------	----------	--------	-------

1. Write Metadata
2. Goto Flow Table
3. Write action(s) to action set
 1. Output: Send packet to specified port
 2. Drop
 3. Set-Queue: Assign packet to specified queue
 4. Set-Field: Modify packet header field(s)
 5. Change-TTL

Ingress port	Packet header fields	Pipeline Metadata
--------------	----------------------	-------------------

OpenFlow software components

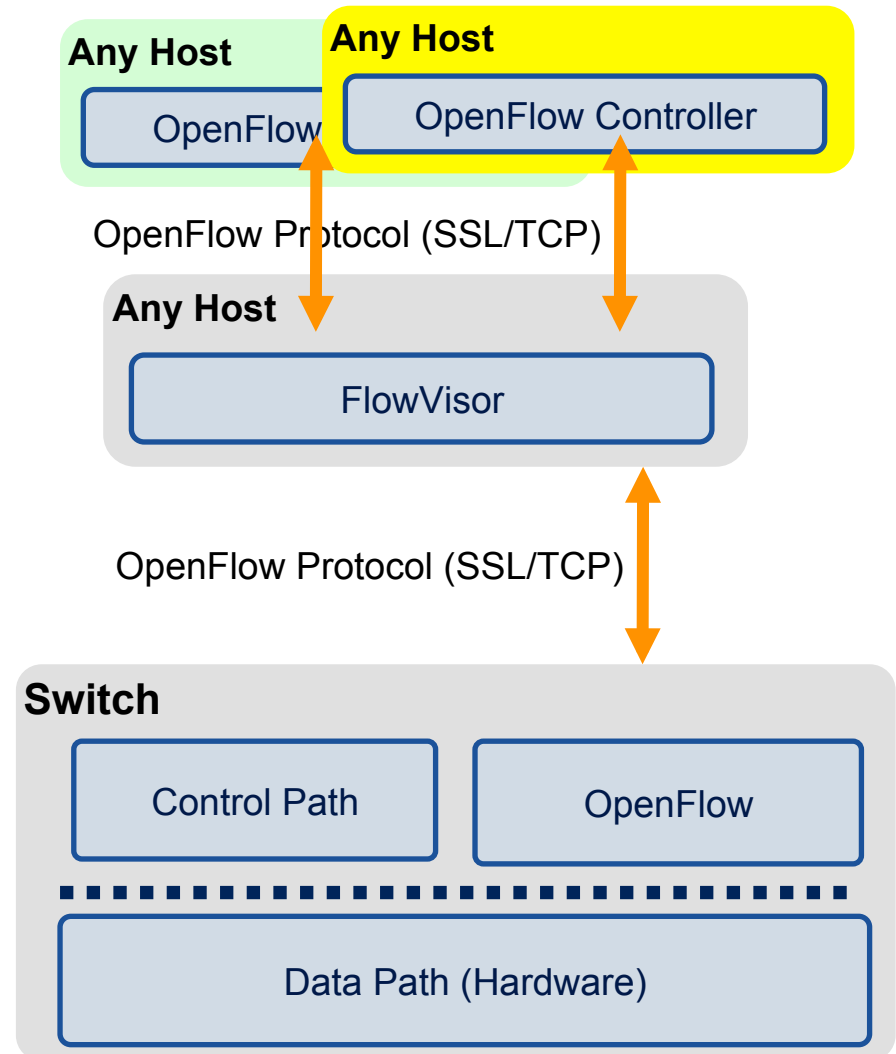


OpenFlow controller implementations

- NOX
 - Efficient C++ based controller for Linux
 - <http://www.noxrepo.org/>
- POX
 - Python-based version of NOX
 - Less efficient but useful for rapid prototyping
- Beacon
 - Modular Java-based OpenFlow controller
 - <https://openflow.stanford.edu/display/Beacon/Home>
- Floodlight
 - Java-based, similar to Beacon
 - <http://www.projectfloodlight.org/floodlight/>

Switch slicing with FlowVisor

- Normally: One controller per switch
- FlowVisor is a proxy that lets multiple controllers manage one switch
- Supports virtualization by splitting switch into slices, each managed by different controller



OpenFlow software switches

- Open vSwitch
 - C/Python software switch with OpenFlow support
 - Integrated into the GNU/Linux kernel
 - <http://openvswitch.org/>
- Pica8
 - C-based hardware-agnostic switch operating system with OpenFlow support
 - <http://www.pica8.com/open-switching/>
- Pantou
 - OpenFlow implementation for OpenWRT-based wireless routers

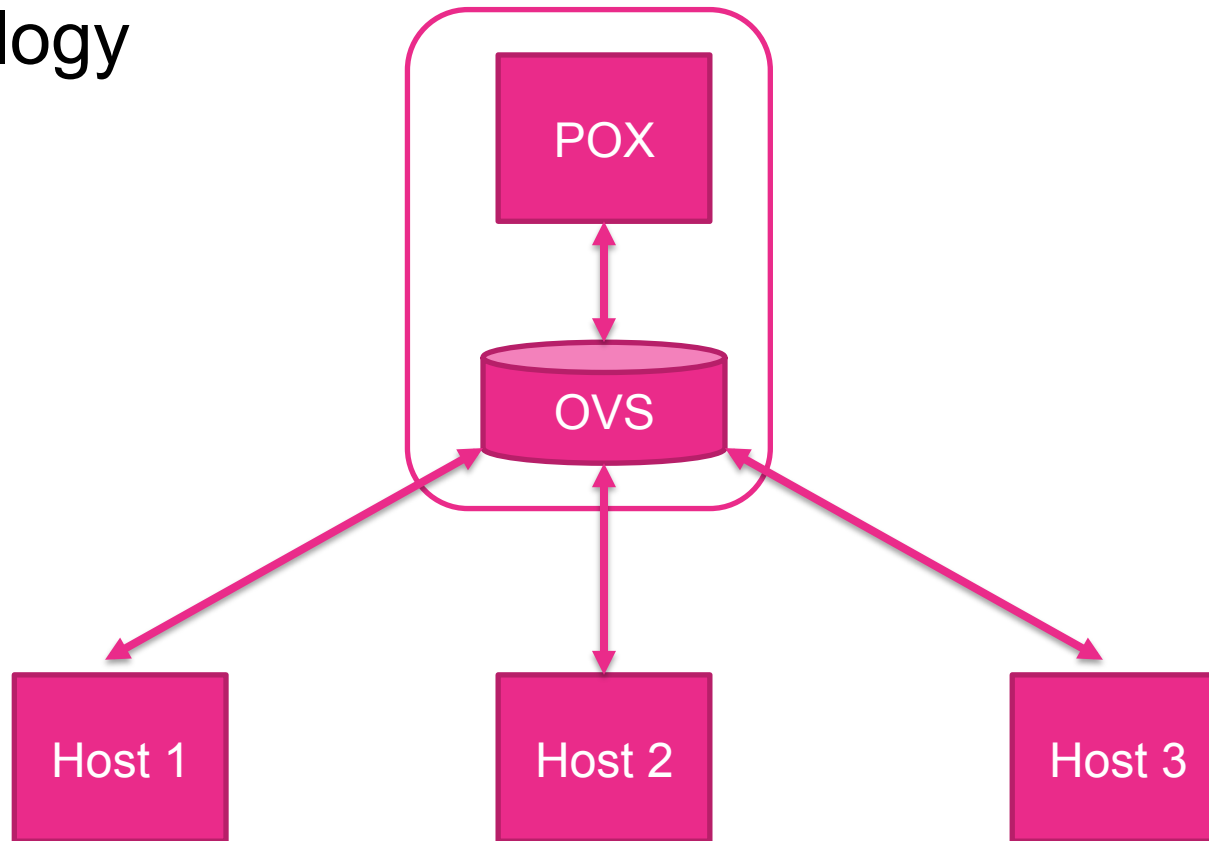
OpenFlow simulation with Mininet

- Mininet creates a virtual network on one or multiple computers
- The network consists of virtual machines running real code on top of GNU/Linux
- Supports OpenFlow through integration of Open vSwitch
- Many OpenFlow controllers can be used, including NOX/POX
- Developed controller code can be easily transferred to testbed or real deployment

Running OpenFlow experiments on the Virtual Wall

Loading RSpec

- Rspec at <http://jfed.iminds.be/ovs.rspect>
- Topology



Loading RSpec

The screenshot shows the jFed Experimenter Toolkit application window. The title bar reads "jFed Experimenter Toolkit". The main menu bar includes "General", "New", "Open", "Open URL", "Save", "Run", "Update Status", "Terminate", "Recover", "Preferences", and "Report a bug". The "Open URL" button is highlighted with a pink box. A dialog box titled "Open Experiment Definition" is open, showing a text input field with the URL "http://jfed.iminds.be/ovs.rspec". Below the dialog, a pink callout box contains the text "Find wall2 & Replace by wall1 in RSpec".

Configure switch

- Ethernet bridge acting as software switch was added during configuration

```
% sudo ovs-vsctl list-br
```

- Add interfaces to this bridge that will act as ports of the software switch

```
% sudo ifconfig eth1 0
```

```
% sudo ovs-vsctl add-port br0 eth1
```

```
% sudo ovs-vsctl list-ports br0
```

- Check interface-host mapping with ping

Point switch to controller

- The controller can be hosted anywhere, here it is on localhost

```
% sudo ovs-vsctl set-controller br0 tcp:  
127.0.0.1:6633
```

- Standalone vs secure mode

```
% sudo ovs-vsctl set-fail-mode br0 secure
```

```
% sudo ovs-vsctl set-fail-mode br0 standalone
```

- What is the difference?

Learning controller

- Try to ping between hosts
- Start learning controller

```
% cd /local/pox
```

```
% ./pox.py --verbose SimpleL2Learning
```

```
node2:~% ping node3
PING node3-link6 (10.0.1.3) 56(84) bytes of data.
From node2-link5 (10.0.1.2) icmp_seq=1 Destination Host Unreachable
From node2-link5 (10.0.1.2) icmp_seq=2 Destination Host Unreachable
From node2-link5 (10.0.1.2) icmp_seq=3 Destination Host Unreachable
From node2-link5 (10.0.1.2) icmp_seq=4 Destination Host Unreachable
From node2-link5 (10.0.1.2) icmp_seq=5 Destination Host Unreachable
From node2-link5 (10.0.1.2) icmp_seq=6 Destination Host Unreachable
64 bytes from node3-link6 (10.0.1.3): icmp_req=7 ttl=64 time=2008 ms
64 bytes from node3-link6 (10.0.1.3): icmp_req=9 ttl=64 time=0.428 ms
64 bytes from node3-link6 (10.0.1.3): icmp_req=8 ttl=64 time=1038 ms
64 bytes from node3-link6 (10.0.1.3): icmp_req=10 ttl=64 time=0.440 ms
64 bytes from node3-link6 (10.0.1.3): icmp_req=11 ttl=64 time=0.479 ms
```

Soft vs Hard Timeouts

- Soft Timeout
 - If no matching packets received, how long will flow remain in forwarding table
- Hard Timeout
 - Total time a flow will remain in forwarding table, independent of matching packets are received

```
msg.match = of.ofp_match.from_packet(self.packet, self.event.port)
msg.idle_timeout = 10
msg.hard_timeout = 30
```

Traffic duplication

- Check traffic on each interface of switch using tcpdump

```
% sudo tcpdump -i <data_interface_name>
```

- Duplicate all traffic and send to host3

```
% ./pox.py --verbose DuplicateTraffic  
--duplicate_port=[eth_host3]
```


Traffic duplication

```
class DuplicateTrafficSwitch(SimpleL2LearningSwitch):  
  
    def __init__(self, connection, duplicate_port):  
        SimpleL2LearningSwitch.__init__(self, connection, False)  
        self._connection = connection;  
        self._duplicate_port=duplicate_port  
        self._of_duplicate_port=getOpenFlowPort(connection, duplicate_port)  
  
    def _handle_PacketIn(self, event):  
        log.debug("Got a packet : " + str(event.parsed))  
        self.packet = event.parsed  
        self.event = event  
        self.macLearningHandle()  
        out_port = self.get_out_port()  
        self.forward_packet([out_port, self._of_duplicate_port])  
  
class DuplicateTraffic(object):  
    def __init__(self, duplicate_port):  
        core.openflow.addListener(self)  
        self._duplicate_port= duplicate_port  
  
    def _handle_ConnectionUp(self, event):  
        log.debug("Connection %s" % (event.connection,))  
        DuplicateTrafficSwitch(event.connection, self._duplicate_port)  
  
def launch(duplicate_port="eth4"):  
    log.debug("DuplicateTraffic" + duplicate_port);  
    core.registerNew(DuplicateTraffic, duplicate_port)
```

Port forward controller

- Redirect traffic to different port
- Run netcat on port 5000 and 6000 on host2

```
% nc -l 5000
```
- Check with learning controller if connection works

```
% nc 10.0.1.2 5000
```
- Update ext/port_forward.config
- Run portforwarding controller

```
% ./pox.py --verbose PortForwarding
```
- Check if port forwarding works

Server proxy controller

- Redirect traffic to other host
- Run netcat on port 5000 and 7000 on host2 and host3 respectively

```
% nc -l 5000
```
- Update ext/proxy.config
- Run proxy controller

```
% ./pox.py --verbose Proxy
```
- Check if proxy works

Further reading and contact information

- Fed4FIRE
 - Website: <http://www.fed4fire.eu>
 - Documentation: <http://doc.fed4fire.eu>
 - Portal: <http://portal.fed4fire.eu>
 - jFed: <http://jfed.iminds.be>
- Maxim Claeys: maxim.claeys@intec.ugent.be
- Jeroen Famaey: jeroen.famaey@intec.ugent.be
- Niels Bouten: niels.bouten@intec.ugent.be