

## 7. The Open Network Lab

- Overview and getting started
- Building a network topology
- Configuring routes and filters
- Monitoring traffic

*Jon Turner*

## The Open Network Lab

- Internet-accessible networking lab ([onl.wustl.edu](http://onl.wustl.edu))
  - » built around set of extensible gigabit routers
  - » intuitive Remote Lab Interface makes it easy to get started
  - » extensive facilities for performance monitoring
- Variety of resources
  - » 4 eight port routers, called Network Services Platform (NSP)
    - highly configurable
    - embedded processor at each port with *plugin* environment
  - » 14 five port Network Processor based Routers (NPR)
    - also highly configurable
    - each has five processor cores reserved for plugins
  - » 6 four port NetFPGA cards
    - hardware can be reconfigured to implement different devices
  - » over 100 rack-mount computers that serve as end systems
    - including multicore servers with 8 cores and 48 cores

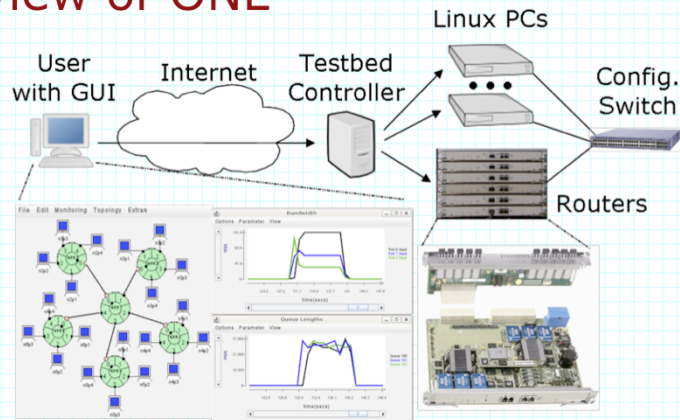
# Sample ONL Session

The screenshot displays a network simulation environment with several key components:

- Network Configuration:** A central diagram showing a network topology with multiple routers and hosts.
- Routing Table:** A window showing the routing table for NSP1:port7, listing destinations, metrics, and next hops.
- Bandwidth Usage:** A graph showing bandwidth usage over time for NSP1:port7.
- Queue Lengths:** A graph showing queue lengths over time for NSP1:port0.
- Packet Losses:** A graph showing packet losses over time for NSP1:port2.
- Queue Parameters:** A window showing queue parameters for NSP1:port0, including link bandwidth and queue sizes.
- Router Plugin Commands:** A window showing the execution of various router plugin commands.
- SSH Window:** A terminal window showing the output of an SSH session, including ping delays and connection statistics.

Callouts in green speech bubbles identify these components: "Bandwidth Usage", "Queue Lengths", "Packet Losses", "Queue Parameters", "Router Plugin Commands", "SSH window to host showing ping delays", and "Routing Table".

## Overview of ONL



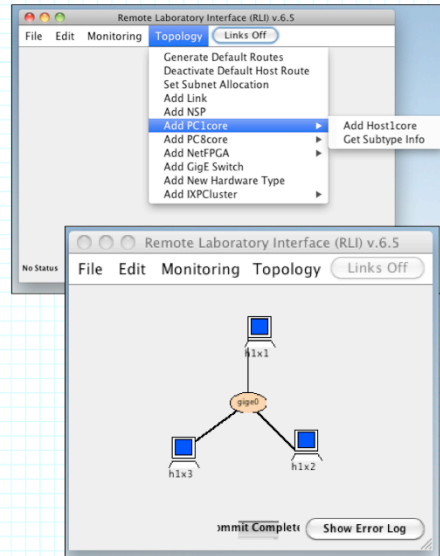
- Remote access through the Internet using a graphical user interface ( called the RLI)
- Provides access to variety of hardware resources
- Experimental networks built with configuration switches

## Getting Started

- Request an account at onl.wustl.edu
  - » read Getting Started page and look at tutorial pages
- Download the RLI (version 7.5 is the latest)
  - » to run RLI you must have Java Runtime Environment installed (version 1.6 or higher)
- Create experimental network and save to a file
  - » details on next slide
- Open an SSH connection to ONL with "tunnel" for RLI
  - » on command line: `ssh -L 7070:onlsrv:7070 user@onl.wustl.edu`
- Make a reservation using RLI and wait for confirmation
  - » note the start time in the confirmation message
- After start time, select File→Commit in RLI
  - » open ssh connections to your ONL hosts and run applications

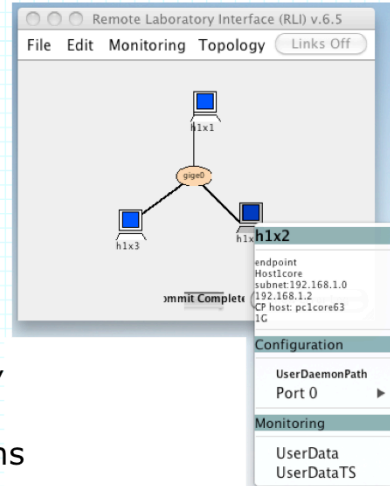
## Building a Basic Configuration

- Select
  - » Topology → Add PC1core
    - AddHost1core
  - » and specify 3 hosts
- Select
  - » Topology → Add GigE Switch
  - » and specify single 1 Gb switch
- Select
  - » Press "Links Off" button
  - » and wire hosts to switch
- Select
  - » File → Save As ...
  - » File → Make Reservation
  - » File → Commit (do not forget!!)



## Running Applications

- Open ssh connections to hosts
  - » first ssh to onl.wustl.edu
  - » then, to pc1coreNN ...
  - » note, each host has two names
- Run standard apps
  - » on h1x1, type "ping h1x3"
  - » on h1x2, type "iperf -s", then on h1x1, type "iperf -c h1x2"
  - » on h1x2, type "sudo /usr/sbin/tcpdump -i data0" then repeat ping from h1x1
- Run basic client/server programs
  - » on h1x2, run java UdpEchoServer
  - » on h1x1, run java UdpEchoClient h1x2 9876 "hello world"



# Working with Larger Configurations

The screenshot displays the Remote Laboratory Interface (RLI) v.6.5 software. The 'Topology' menu is open, showing options such as 'Generate Default', 'Deactivate Default', 'Set Subnet Allocation', 'Add Link', 'Add NSP', 'Add PC1core', 'Add PC8core', 'Add NetFPGA', 'Add GigE Switch', 'Add New Hardware', and 'Add IXPCluster'. The main window shows a network diagram with two routers, NPR.1 and NPR.2, connected to each other and to a central switch labeled 'spine'. Various hosts (h4x2, h5x2, h1x1, h2x3, h2x4, h2x5, h3x2, h7x1, h6x1) are connected to the routers. A 'Commit Completed' button and a 'Show Error Log' button are visible at the bottom right of the interface.

**To include pair of NPRs (routers), select Topology**  
 → Add IXPCluster  
 → Add NPRCluster

**Note subnet numbering**

**Note multiple paths between routers**



# Working With Routing Tables

The screenshot shows the Remote Laboratory Interface (RLI) v.6.5. On the left, a network topology diagram features a central green router with four ports (1-4) and several host icons (h4x2, h5x2, h1x1, h2x3, h2x5). A 'Generate Default Routes' menu is open over the router. On the right, the configuration window for 'NPR.1:port4' is displayed, showing a routing table with the following entries:

prefix/mask	nexthop	stats
192.168.4.0/24	0:0.0.0.0	60
192.168.8.0/24	1:192.168.8.2	65
192.168.2.0/24	2:0.0.0.0	70
192.168.1.0/24	3:0.0.0.0	75
192.168.5.0/24	4:0.0.0.0	80
192.168.7.0/24	1:192.168.8.2	95
192.168.6.0/24	1:192.168.8.2	100
192.168.3.0/24	1:192.168.8.2	105

Callout 1: 'Generate default routes to configure routing tables' points to the 'Generate Default Routes' menu.

Callout 2: 'Click on port to access specific table' points to the 'Port 4' configuration window.

Callout 3: 'Next hop specifies output port and next hop's IP address' points to the 'nexthop' column in the routing table.

Callout 4: 'Change routes using Operations -> Edit' points to the 'Operations' tab in the configuration window.

# Configuring Packet Filters

The screenshot shows the RLI interface with a network topology diagram at the top and a configuration window for 'NPR.1:port0' below. The configuration window displays a table for the 'FilterTable' with the following data:

priority	type	address/mask/port	protocol	tcpflags	exceptions	plugin...	output	qid	stats	on
50	aux:false sampling type:0 multicast:false	src:192.168.0.0/16 port * dest:192.168.0.0/16 port *	*	fin:* syn:* rst:* psh:* ack:* urg:*	nonip:0	0	dropped:false ports:use route plugins:0 pluginSelection:plugin(unicast)	1	1	<input checked="" type="checkbox"/>

Callouts in the image point to specific parts of the interface:

- Select Filter Table:** Points to the 'Port 0' configuration window in the topology diagram.
- address prefixes and ports:** Points to the 'address/mask/port' column in the table.
- protocol and flags (TCP):** Points to the 'protocol' and 'tcpflags' columns in the table.
- packet treatment:** Points to the 'output' column in the table.
- turn on/off:** Points to the 'on' checkbox in the table.

# Configuring Packet Filters

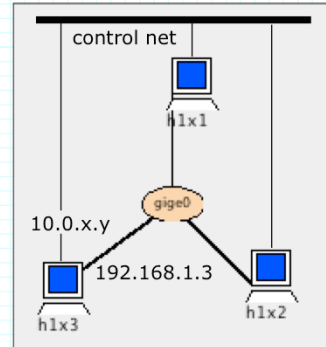
The screenshot shows a network configuration window titled "Edit filter" with a sidebar on the left containing sections for "Operations", "Configuration", and "Monitoring". The "Configuration" section is active, showing fields for "src: 192.168.0.0" and "dest: 192.168.0.0". A "queue number" callout points to the "qid: 1" field. An "address fields" callout points to the "destination\_address: 192.168.0.0" and "source\_address: 192.168.0.0" fields. A "protocol and ports" callout points to the "protocol: \*" and "destination\_port: \*" fields. An "output link" callout points to the "output\_plugins: 0" field. The "Operations" section shows "AddFilter", "Edit filter", and "DeleteFilter" options. The "Monitoring" section shows "StatsPreQPkt", "StatsPostQPkt", and "StatsPreQByte". The "Edit filter" window also displays "destination\_mask: 16" and "source\_mask: 16". At the bottom right of the window, there are "Enter" and "Cancel" buttons.

# Monitoring Traffic

The screenshot displays the Remote Laboratory Interface (RLI) v.6.5 software. The main window shows a network topology with nodes labeled h4x2, h5x2, h1x1, h2x3, h4x2, h5x2, h1x1, h7x1, and h6x1. A configuration window for 'Port 1' is open, showing a 'Monitoring' section with a list of parameters: ReadQueueLength, RXPKT, RXBYTE, TXPKT, and TXBYTE. A green callout bubble points to the 'Add Monitoring Display' menu item, stating 'To add new real-time chart'. Another callout bubble points to the 'TXPKT' parameter, stating 'to select outgoing packet rate at a port'. A third callout bubble points to a real-time chart titled 'packets' showing traffic volume over time (102 to 121.5 seconds), with a legend for 'out 1.1', 'out 1.2', 'out 2.0', and 'out 2.4'. The chart shows a series of green vertical bars representing traffic volume. The number '12' is visible in the bottom right corner of the interface.

## Hosts with Multiple Interfaces

- ONL hosts have two interfaces
  - » data interface used for experimental network
    - names of form hAxB. and addresses of form 192.168.AxB addresses,
  - » control net interface used to talk to ONL servers
    - names of form pc1coreNN, addresses of form 10.0.x.y
    - file /users/onl/.topology defines shell variables of form \$hAxB that map to control net names



- To force server connections to use a specific interface, bind socket to desired interface's IP address
  - » forces communication to server to come through specified iface
- If socket left "unbound", system uses *wildcard interface*

## Using the Linux Command Line

- ONL hosts all run Linux
  - » to use them effectively, need to be familiar with the command-line interface provided by the "shell" (specifically, bash)
- Common commands
  - » ls to list files in a directory (aka folder)
  - » cd to change the current working directory
  - » more to examine contents of a file
  - » ssh to open an ssh connection to another computer
  - » man to get documentation for a given command
    - type "man ls" to learn how to use the ls command
  - » See <http://www.ee.surrey.ac.uk/Teaching/Unix/index.html> and <http://www.gnu.org/software/bash/manual/bash.html>
- You need to invest some time up-front to learn Linux
  - » but, you'll be much more productive once you do

## Using Wireshark in ONL

- Wireshark is implemented using the X-windows system
  - » ssh supports tunneling of X-windows commands, allowing you to run *Wireshark* in ONL with GUI running on laptop
- On Urbauer lab computers
  - » first type "startxwin" in a command prompt window
  - » this starts new window; in this window, type

```
ssh -X userName@onl.wustl.edu
source /users/onl/.topology
ssh -X onlHostname (e.g. $h4x2)
sudo wireshark
```
  - » this starts *Wireshark* on remote machine, but your local computer acts as the display for remote instance of *Wireshark*
- Can use this with other X-windows applications also