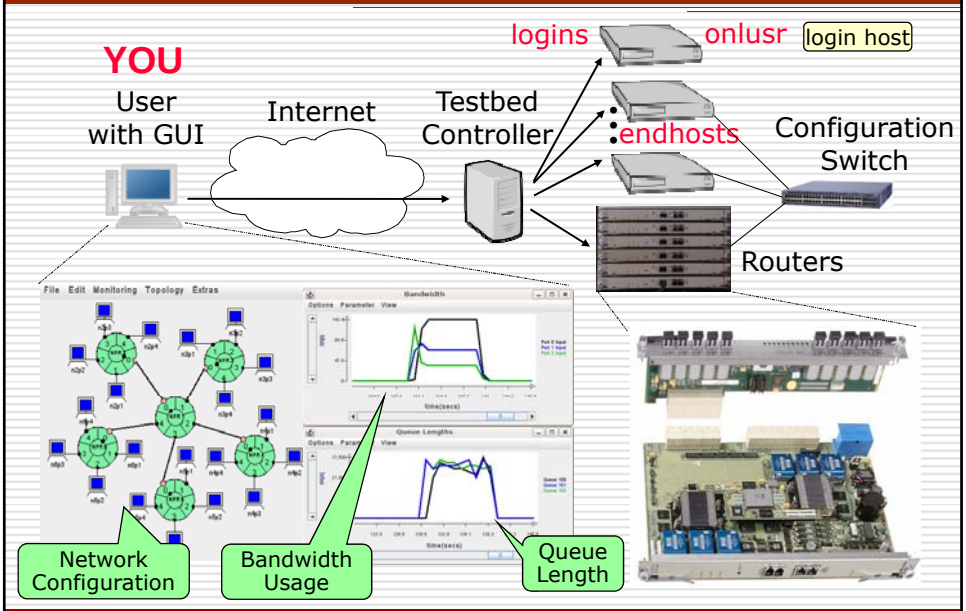
CSE 473S (Fall 2009)
The Open Network Lab (Part 1)

Ken Wong
Applied Research Laboratory
Computer Science and Engineering Department
http://www.arl.wustl.edu/~kenw
kenw@arl.wustl.edu
http://www.onl.wustl.edu (ONL)

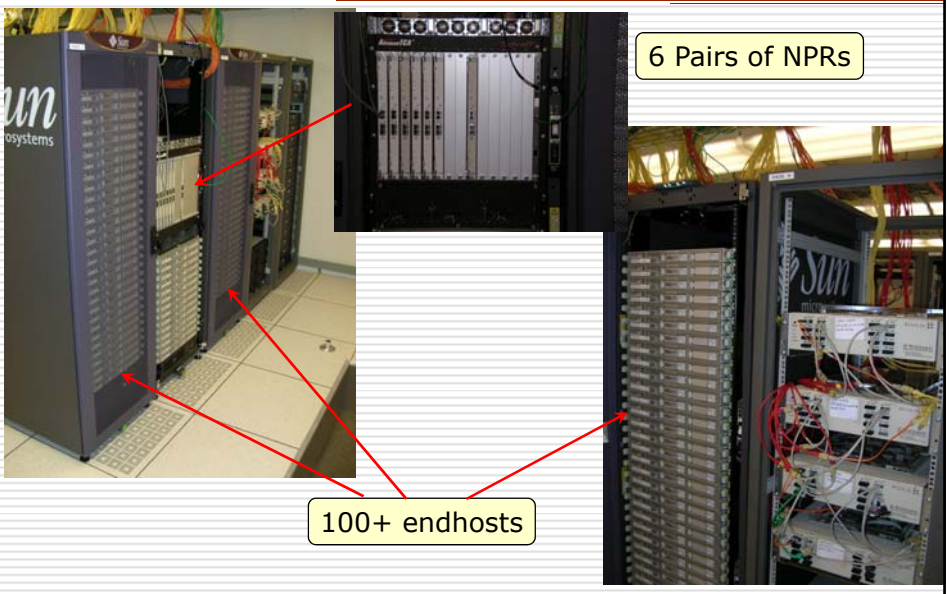
National Science Foundation ANI-023826,
CNS-0551651, REL-0632580

Washington University in St.Louis

Testbed Organization



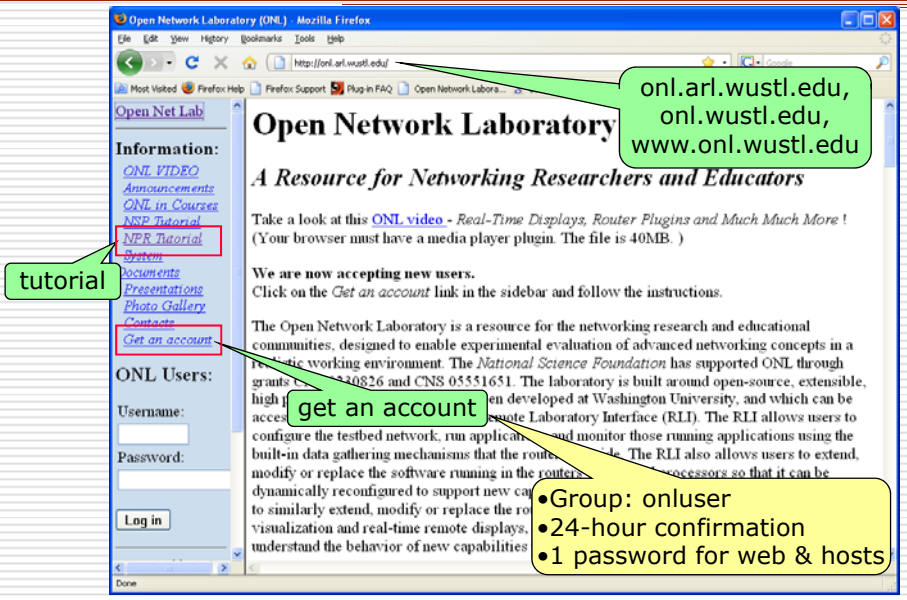
Network Processor Routers (NPRs)



Why Build ONL?

- Hands-on experience with real hardware
  - » Make education more concrete ➔ reinforce concepts
  - » Achieve through virtualization
    - Need to protect user from actions of other users
    - Need to mimic propagation delay and link capacity
- Support experiment/observation approach
  - » Extensive monitoring facility
  - » Observe effects of config. changes on network traffic
    - ➔ insights and greater understanding through observations
  - » Easy-to-use Remote Laboratory Interface (RLI)
- Access to advanced router features
  - » Gbps links, filters, packet scheduling
  - » Plugins: Program insertion along packet data path

Get An Account



Properties of ONL Accounts

- Same password for Web login and host login
- Host account restrictions
  - » Can only SSH to *onl.arl.wustl.edu* from remote host
    - Actually end up on *onlusr*, the ONL user host
  - » *Firewall* blocks all connections from within ONL
    - Can't push from ONL host to remote host
    - Can't pull to ONL host from remote host
    - But can push to ONL host from remote host (e.g., scp)
    - But can pull from ONL host from remote host
    - No email
  - » Can only access *onlusr* host and hosts assigned to your experiment
- Password-free SSH between ONL hosts
  - » See "NPR Tutorial➔Summary Information➔Password-Free SSH"

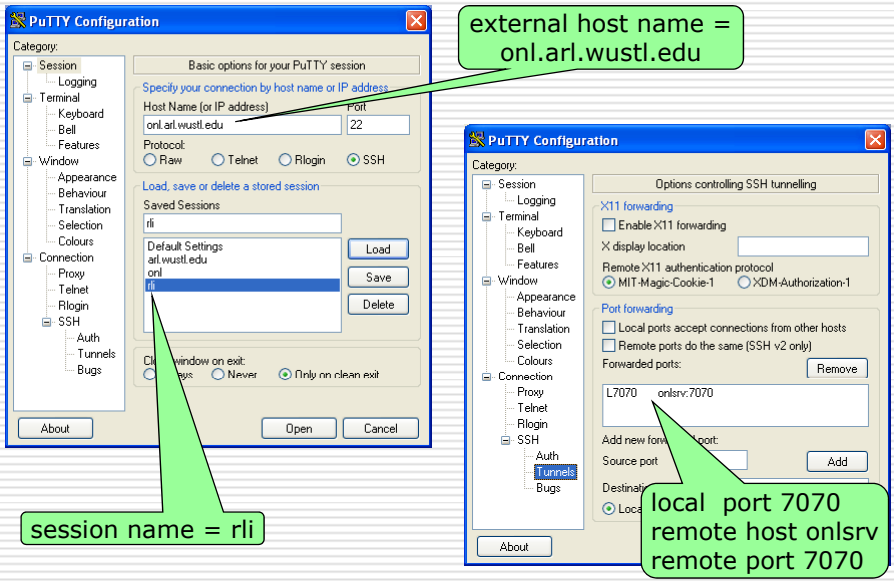
Getting Started

- See sidebar at www.onl.wustl.edu
- Follow instructions in "Getting Started" link
- Read NPR Tutorial
  - » Packet Processing
  - » The Remote Laboratory Interface
  - » Filters, Queues and Bandwidth
- Other useful tutorial sections
  - » Examples ➔ The Remote Laboratory Interface
  - » Examples ➔ Filters, Queues and Bandwidth

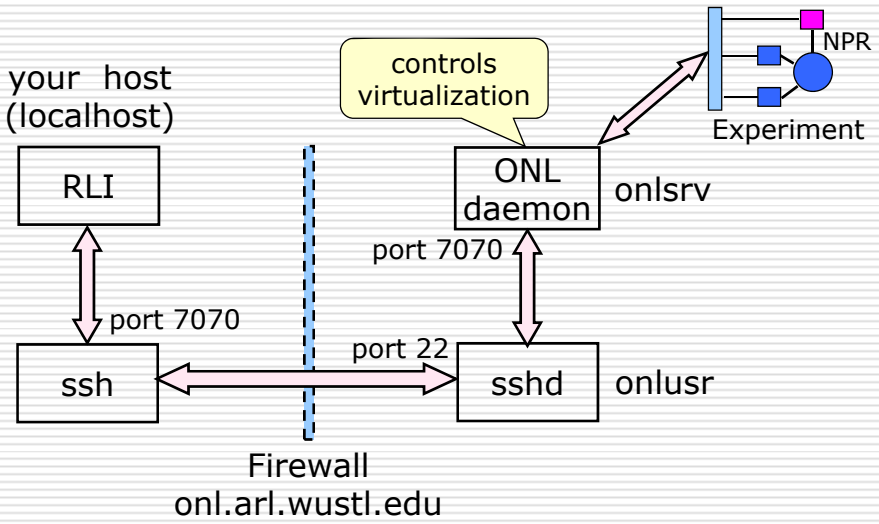
SSH Tunnel Configuration

- *Build before each experimental session*
  - » Allows your RLI to communicate with ONL daemon
  - » Needed to make reservation or run experiment
- Cookbook for 3 approaches
  - » *Unix or Unix-like* (e.g., cygwin)
    - **ssh -L 7070:onlsrv:7070 onl.arl.wustl.edu**
    - I use this (defined as an alias) from Linux
  - » *MS Windows PuTTY* (GUI-based)
    - I use this from my Microsoft XP laptop
      - And also the SSH agent *pageant* (optional)
    - See Tutorial for URL (free software)
  - » *Microsoft Windows SSH client tool from SSH Corp*
- Follow instructions on ONL web pages
  - » *Don't agonize over it ➔ See TA after 2-3 tries*
  - » If problems, send email to your grader/consultant

PuTTY SSH Tunnel Configuration



Under the SSH Tunnel Hood

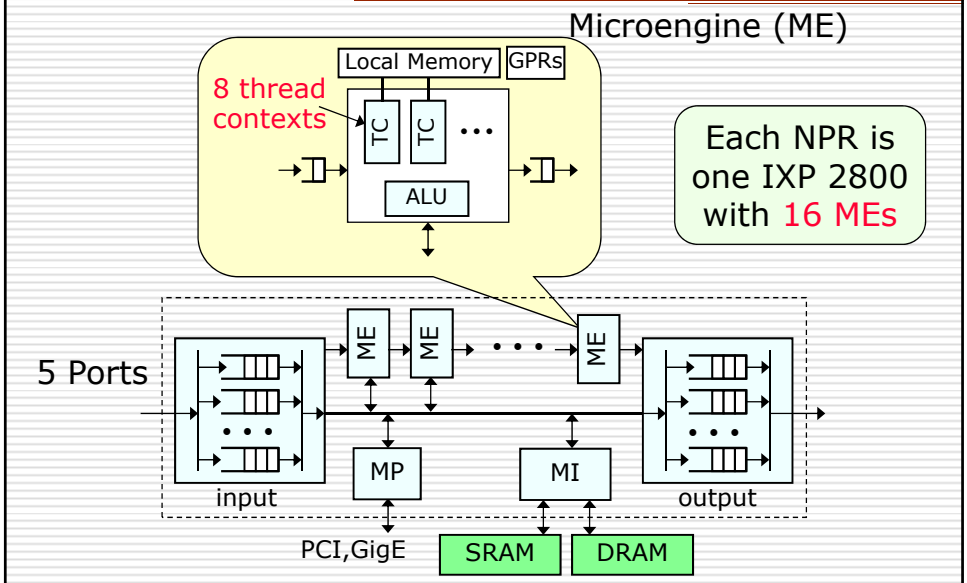


Configuring a Virtual IP Network

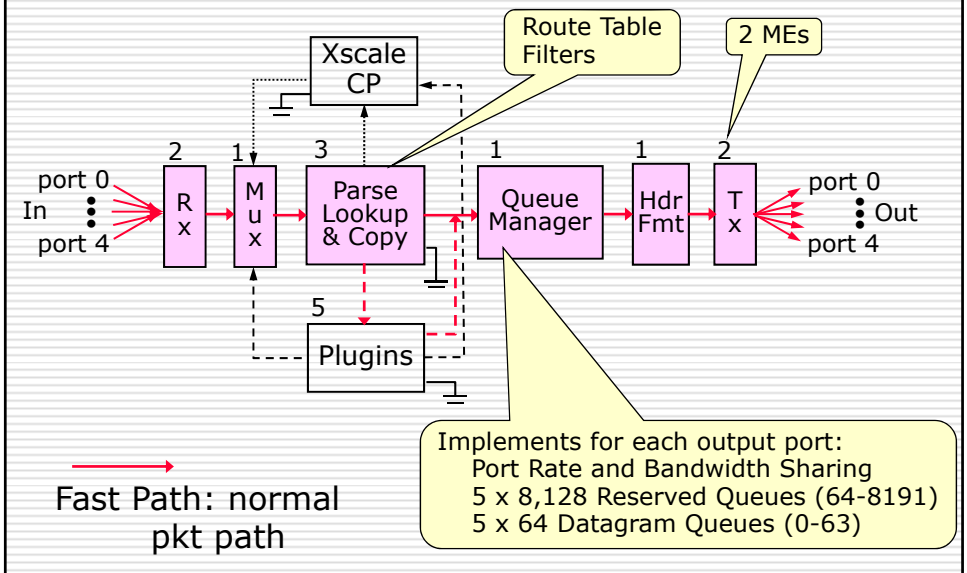
- Topology
  - » NPRs, hosts, links
  - » Used for *resource reservation*
- Route Tables
  - » per port
  - » *Initially empty at all ports*
  - » *default RT*: 1 port or all ports
  - » IP pkt forwarding
    - get pkt 1 hop closer to destination
  - » uses Longest Prefix Match
- Filter Tables
  - » per port
  - » forward to port/queue
  - » forward to plugin, port/queue
- Queue Tables
  - » per port
  - » port rate
  - » datagram queues
    - QIDs 0-63
  - » reserved queues
    - QIDs 64-8,191
  - » queue parameters
    - *threshhold*: queue size
    - *quantum*: scheduling weight
- Plugin Tables
  - » per NPR
  - » standard plugins
    - e.g., delay
  - » user-defined plugins

The NPR Data Path

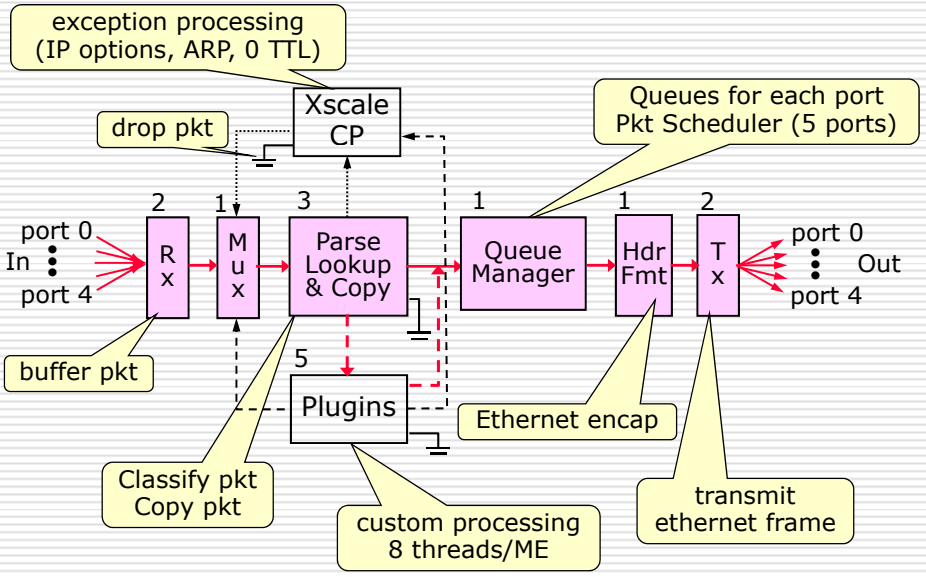
The Network Processor Router (NPR)



NPR Packet Processing (1)



NPR Packet Processing (2)



NPR Packet Processing (3)

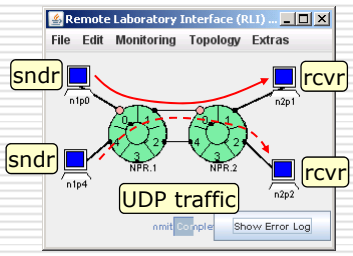
- Rx: Receive Pkt
  - » Put pkt in DRAM; Send meta-packet
- Mux: Multiplex traffic (inputs, CP, plugins)
- PLC: Parse, Lookup and Copy
  - » Implements Route Table and Filter Table lookup
  - » Uses TCAM (Ternary Content-Addressable Memory)
- QM: Queue Manager
  - » Pkt scheduler for each of 5 output ports
    - Implements *port rate concept* using *token bucket*
    - Implements *bandwidth sharing concept* using *Weighted Deficit Round Robin* (WDRR) algorithm
- Hdr Format
  - » Create ethernet frame
- Tx: Transmit Ethernet Frame

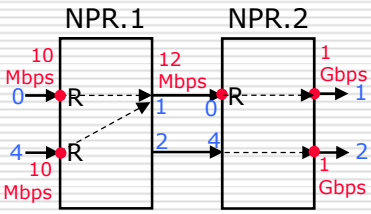
Experiment 1

Topics

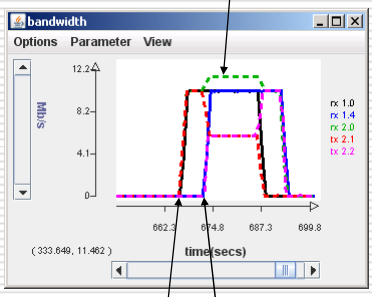
- 2 NPRs, 4 hosts, 2 links between NPRs
- Experiment files
  - 473-1-f09.exp
  - 473-1-filters-f09.exp (filters at ports 1.0 and 1.4)
- Unix commands
  - /sbin/ifconfig network interface properties
  - /bin/netstat network interface statistics
  - /bin/ping ICMP echo/reply
  - /usr/local/bin/iperf UDP/TCP traffic generator
- Route tables and port rate (capacity)

Experiment 1 (Final Version)





NPR 2, port 1 receives Traffic at ALMOST 12 Mbps



n1p0 starts

n1p4 starts

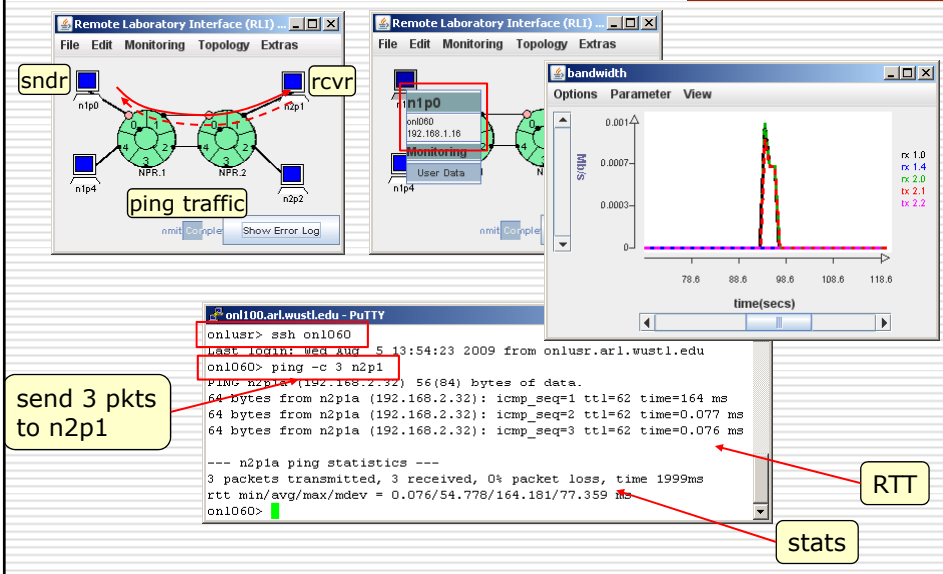
Steps in Running an Experiment

- Define *maximal* network resources using RLI
  - » Required: Add NPR routers, Hosts, Links
- Save configuration
  - » File ➔ Save as
- *Build SSH tunnel to onl.wustl.edu*
- Make a reservation
  - » File ➔ Make reservation
  - » Can be an *advanced reservation*
- When reservation time arrives:
  - » *Build SSH tunnel to onl.wustl.edu*
  - » Run RLI and open configuration file
  - » Ask for network to be built: File ➔ Commit
  - » Define more network parameters (Commit) and monitoring
  - » Wait for commit to finish
- Log into onl.wustl.edu using ssh
  - » Prompt will show you have logged into *onlusr*
  - » You can only access your ONL nodes from onlusr host
  - » Use Linux commands to run traffic generators

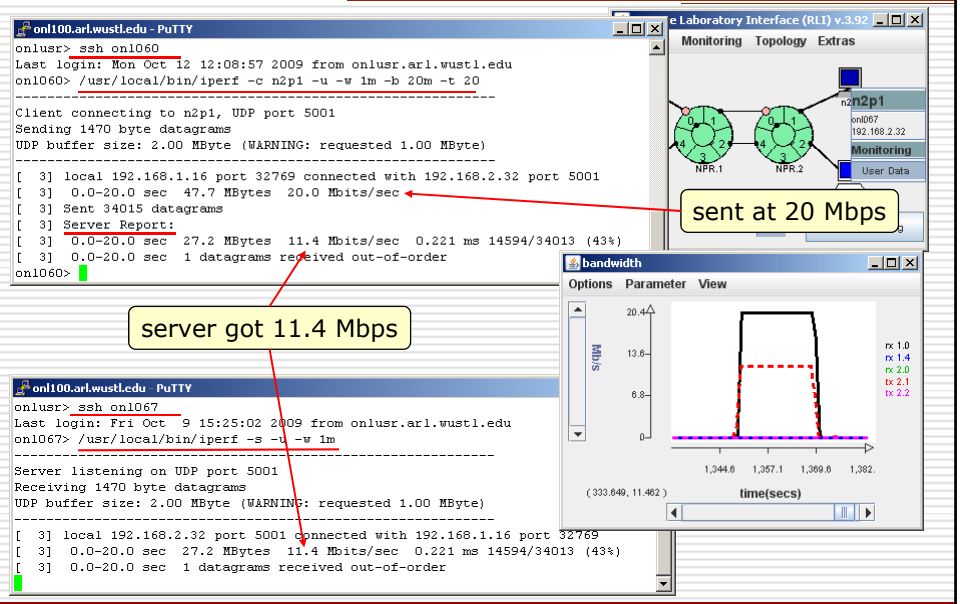
RLI Menus

- Menu File ➔ …
  - » *Open*, *Commit* (binds virtual network to actual network components)
- Port ➔ Route Table ➔ Edit
  - » Add/Delete Route, Generate Local Default Routes
- Port ➔ Queue Table
  - » Port Rate, Edit ➔ Add/Delete Queue, Threshold, Quantum
- Port ➔ Filter Table ➔ Edit
  - » Add/Edit/Delete Filter
- *Menu Monitoring ➔ Add Monitoring Display*

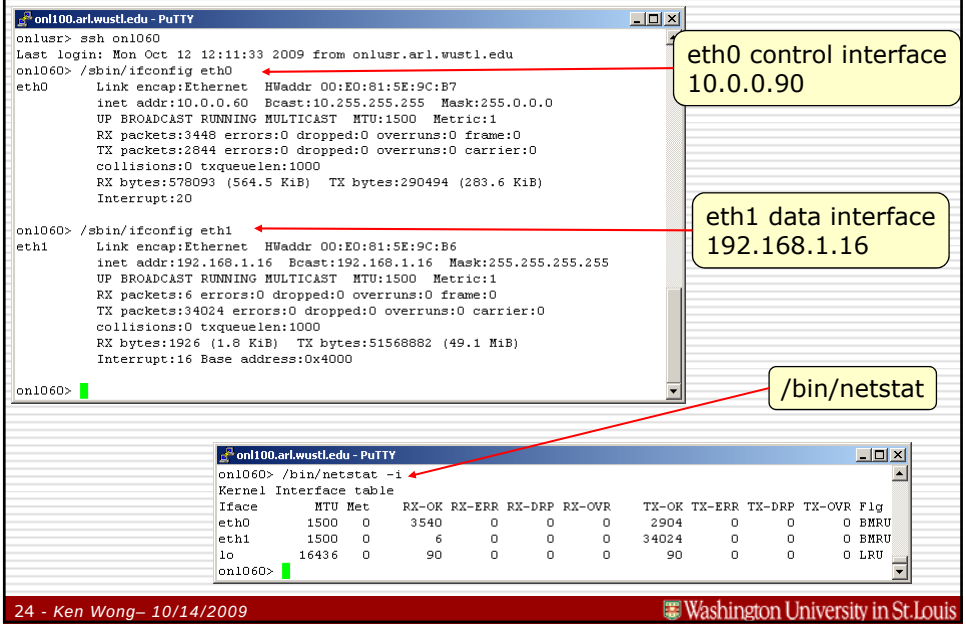
Send ping pkts from n1p0 to n2p1



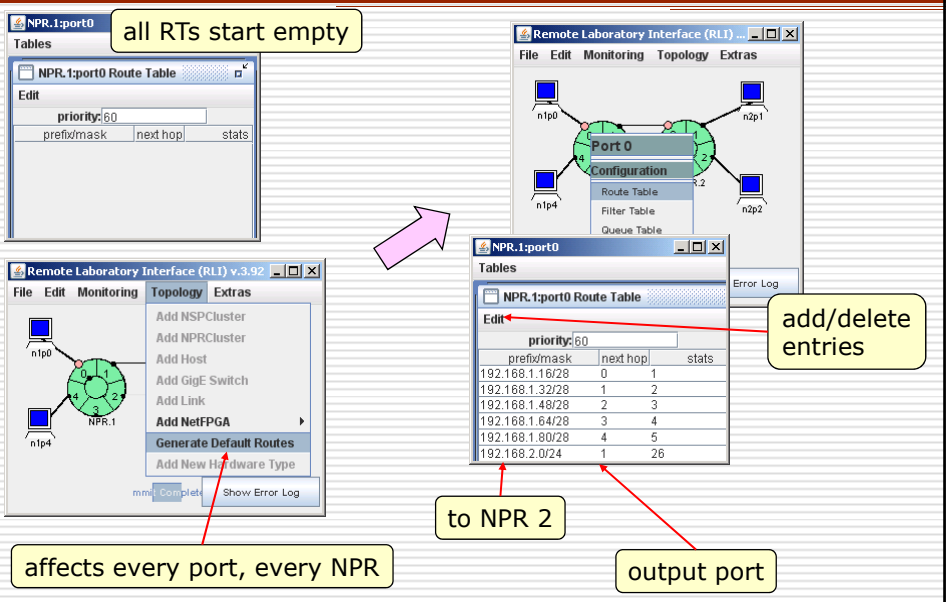
Send UDP pkts from n1p0 to n2p1



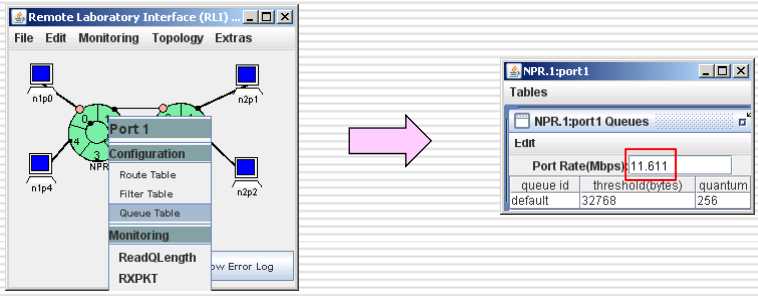
ifconfig and netstat commands



Route Tables

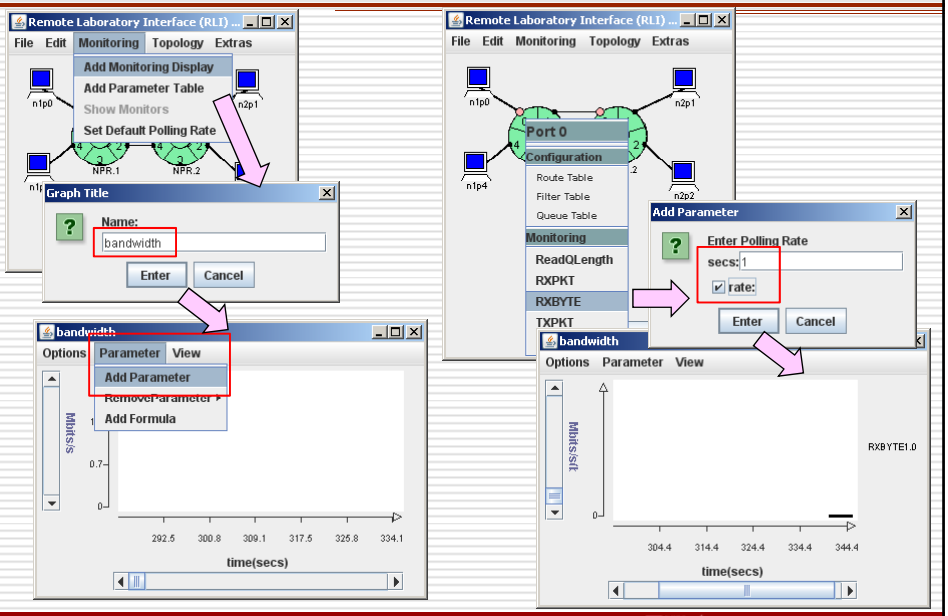


Queue Tables and Port Rate

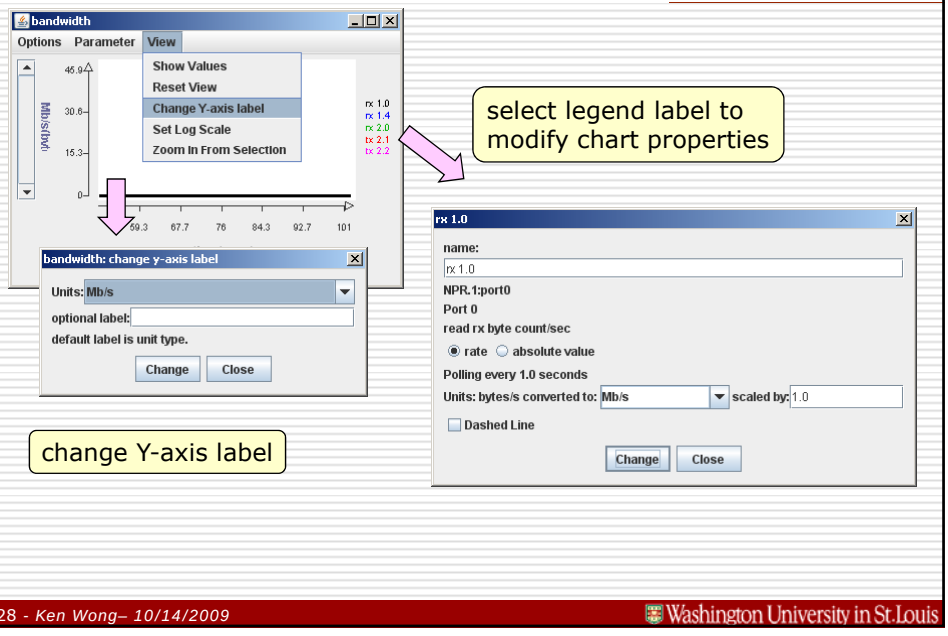


- Default port rate = 1 Gbps
- Requested 12 Mbps
  - » actual max port rate after commit = 11.611 Mbps
    - 683 Kbps granularity
    - 11.611 Mbps = 17 x 0.683 Mbps

Bandwidth Monitoring (1)

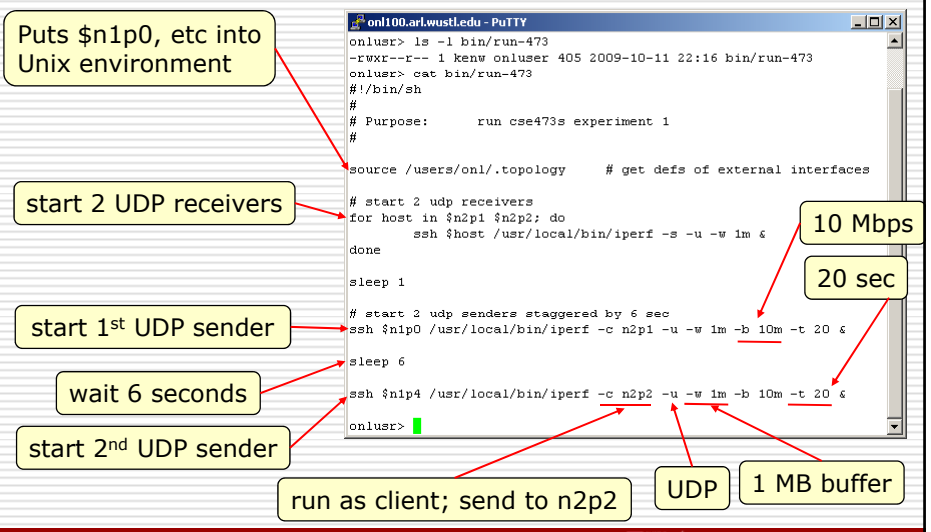


Bandwidth Monitoring (2)



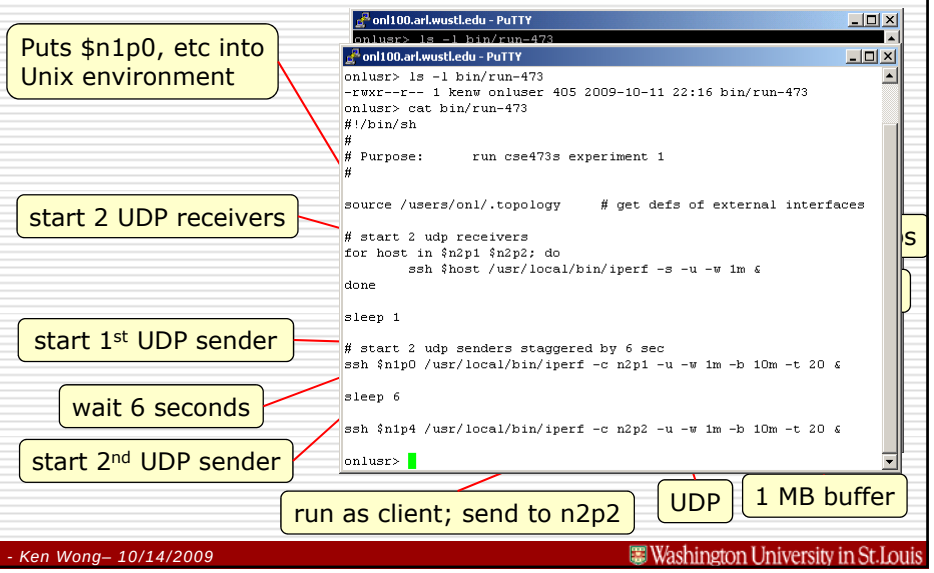
Experiment 1 Shell Script

See ~kenw/bin/run-473

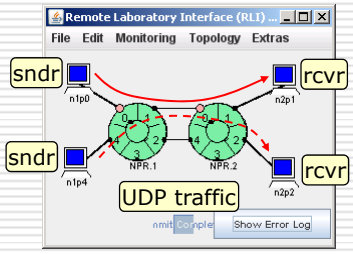


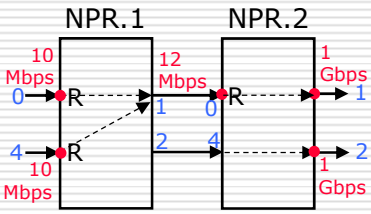
Experiment 1 Shell Script

See ~kenw/bin/run-473

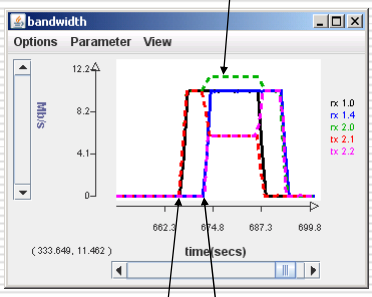


Experiment 1 (Final Version)





NPR 2, port 1 receives Traffic at ALMOST 12 Mbps

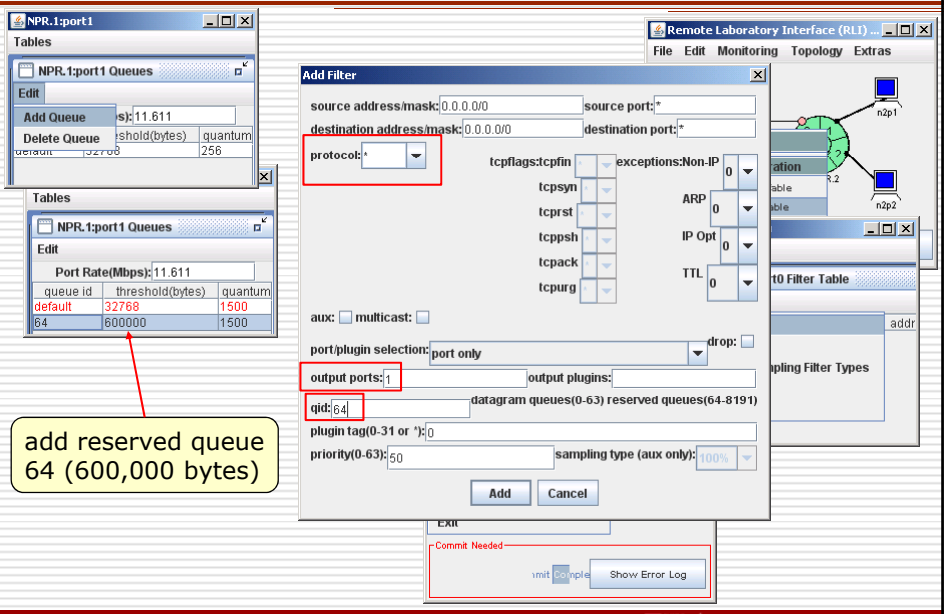


n1p0 starts

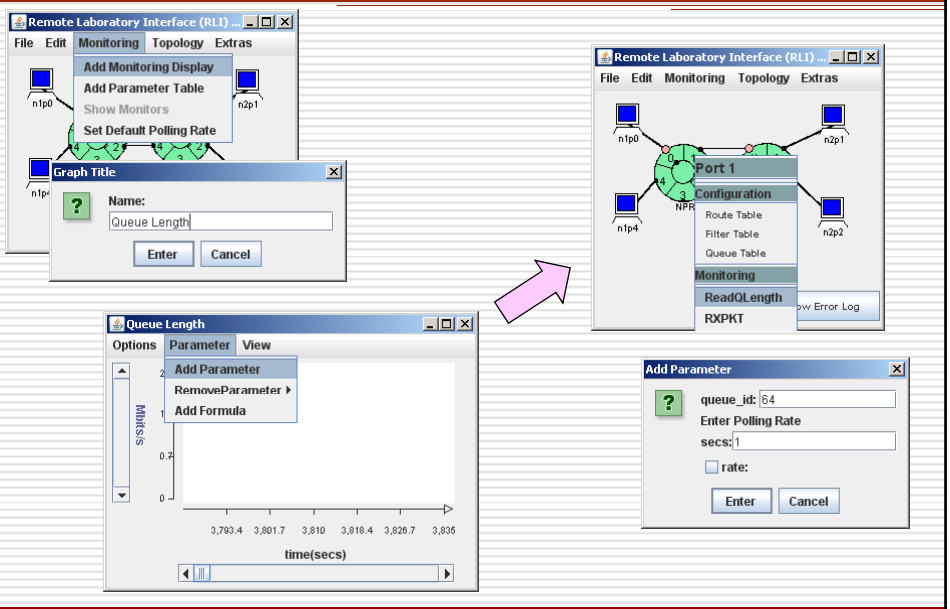
n1p4 starts

Experiment 2

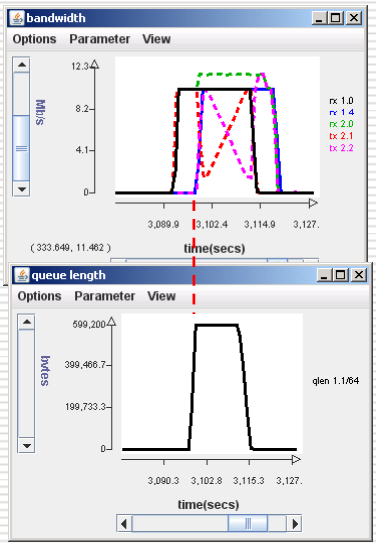
Using Reserved Queue 64



Monitoring Queue Length

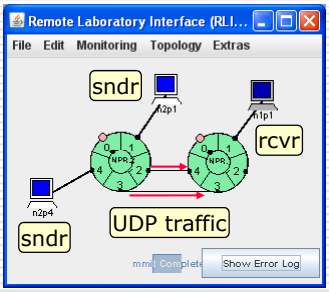


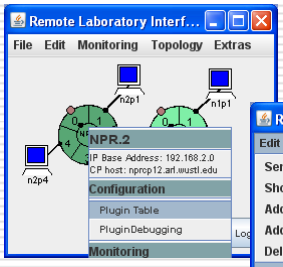
Experiment 2 Displays



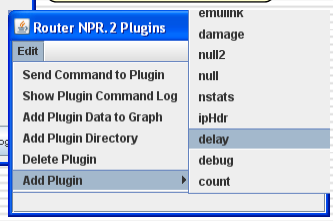
Delaying Packets

Delaying Packets From n2p1

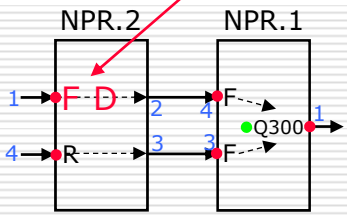




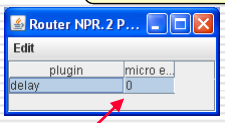
standard plugins
~onl/npr/plugins/



•Add a delay plugin D
•Add a filter F

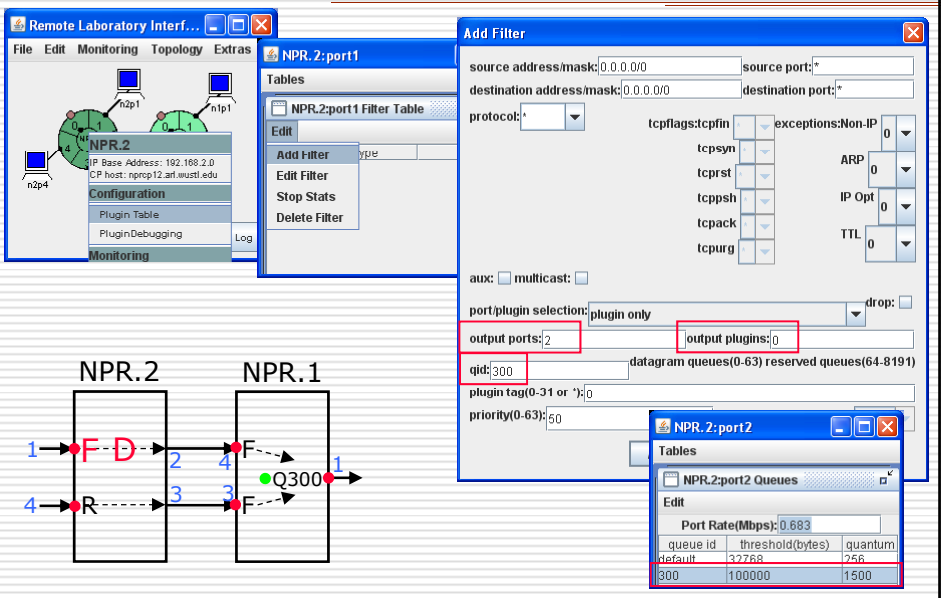


5 possible plugin MEs
numbered 0-4

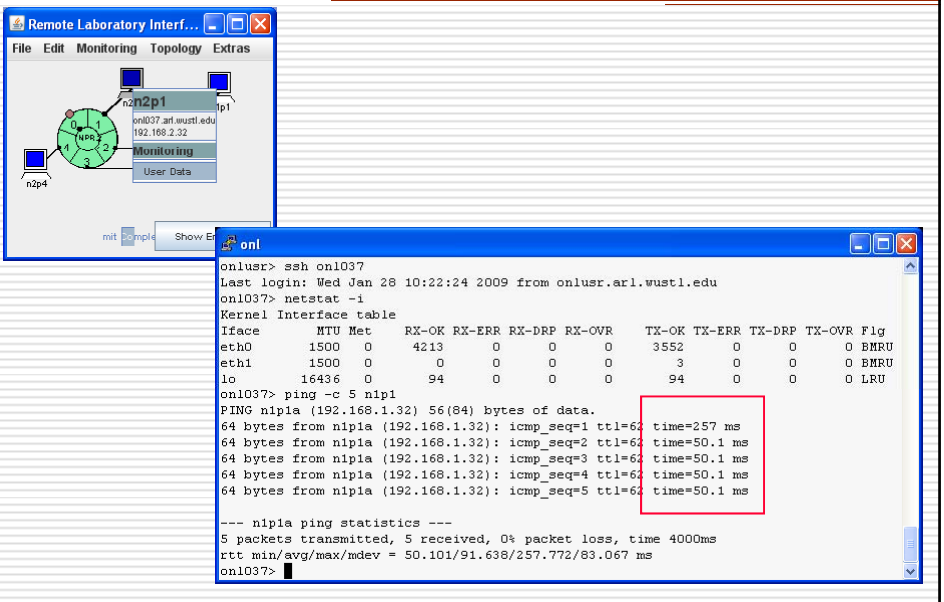


microengine 0

Add The Filter



50 msec Delay (Default)



Send Control Message to Plugin