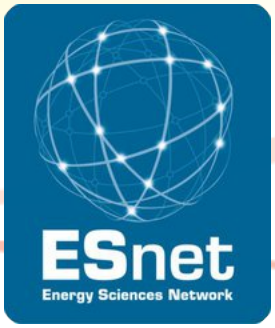


July 22nd 2013, XSEDE Network Performance Tutorial

Jason Zurawski – Internet2/ESnet

Kathy Benninger - Pittsburgh Supercomputing Center

BWCTL, IPERF, & NUTTCP (Oh My)

BWCTL – What is it?

- BWCTL is:
  - A command line client application
  - A scheduling and policy daemon
  - Wraps the throughput testing tools **Iperf** and **Nuttcp**.

- These tests are able to measure:
  - Maximum TCP bandwidth (with various tuning options available)
  - The delay, jitter, and datagram loss of a network when doing a UDP test

BWCTL – Quick Demo

- Caveat … BWCTL is a serial testing tool, e.g. only one test per server at a time. Keep this in mind as we use it – all of your testing at once will most likely cause problems
- Basic syntax:
  - **`bwctl –f m –t 10 –i 1 –c HOST`**
  - **`bwctl –f m –t 10 –i 1 –s HOST`**
- Try at your own risk to other servers, suggestions:
  - bwctl.losa.net.internet2.edu
  - To not overwhelm the server, also try replacing 'losa' with:
    - atla
    - chic
    - hous
    - kans
    - newy
    - salt
    - seat
    - wash

Problem Statement

- Users want to verify available bandwidth/throughput:
  - Between their site and a remote resource
  - Between two remote resources
  - Validate/Verify an SLA

- Methodology:
  - Verify available bandwidth from each endpoint to points in the middle
  - Determine problem area(s)
  - Re-run tests over time – requires access to tool instead of doing a 'one off' test

Typical Solution

- Run "iperf" or similar tool on two endpoints and hosts on intermediate paths
  - Roadblocks:
    - Need software on all test systems
    - Need permissions on all systems involved (usually full shell accounts*)
    - Need to coordinate testing with others *
    - Need to run software on both sides with specified test parameters *
- Desirable features for an alternate method
  - 'Daemon' to run in the background
  - Protocol to exchange results/errors
  - Works with firewalls
  - Protect resources

- (* BWCTL was designed to help with these)

Implementation

- Applications
  - Daemon (bwctld)
  - Client (bwctl)
- Open Source License & Development
  - Modified BSD (http://www.internet2.edu/membership/ip.html)
  - Mailing lists for developer communication – come join us!
- Protocol Abstraction Library
  - Will support development of new clients
  - Add custom 'hooks' into the policy (e.g. add authentication via OpenID or similar)

J. Zurawski – zurawski@es.net & K. Benninger - benninger@psc.edu

TCP Measurements

- Measures TCP Achievable Bandwidth
  - Measurement includes the end system
  - Sometimes called "memory-to-memory" tests
  - Set expectations for well coded application
- Limits of what we can measure
  - TCP ***hides*** details
  - In hiding the details it can obscure what is causing errors
- Many things can limit TCP throughput
  - Loss
  - Congestion
  - Buffer Starvation
  - Out of order delivery

TCP Performance: Window Size

- Use TCP auto tuning if possible
  - Linux 2.6.7 and newer, Mac OS X 10.5, FreeBSD 7.x, and Windows Vista
  - Allow the OS to decide how large the window needs to be based on current resources and performance
- The –w option can be used to request a particular buffer size.
  - Use this if your OS doesn't have TCP auto tuning
  - This sets both send and receive buffer size.
  - The OS may need to be tweaked to allow buffers of sufficient size.
  - See http://fasterdata.es.net/fasterdata/host-tuning/ for more details
- Parallel transfers may help as well, the –P option can be used for this
- To get full TCP performance the TCP window needs to be large enough to accommodate the Bandwidth Delay Product

TCP Performance: Read/Write Buffer Size

- TCP breaks the stream into pieces transparently
- Longer writes often improve performance
  - Let TCP "do its thing"
  - Fewer system calls
- How?
  - -l <size> (lower case ell)
  - Example –l 128K

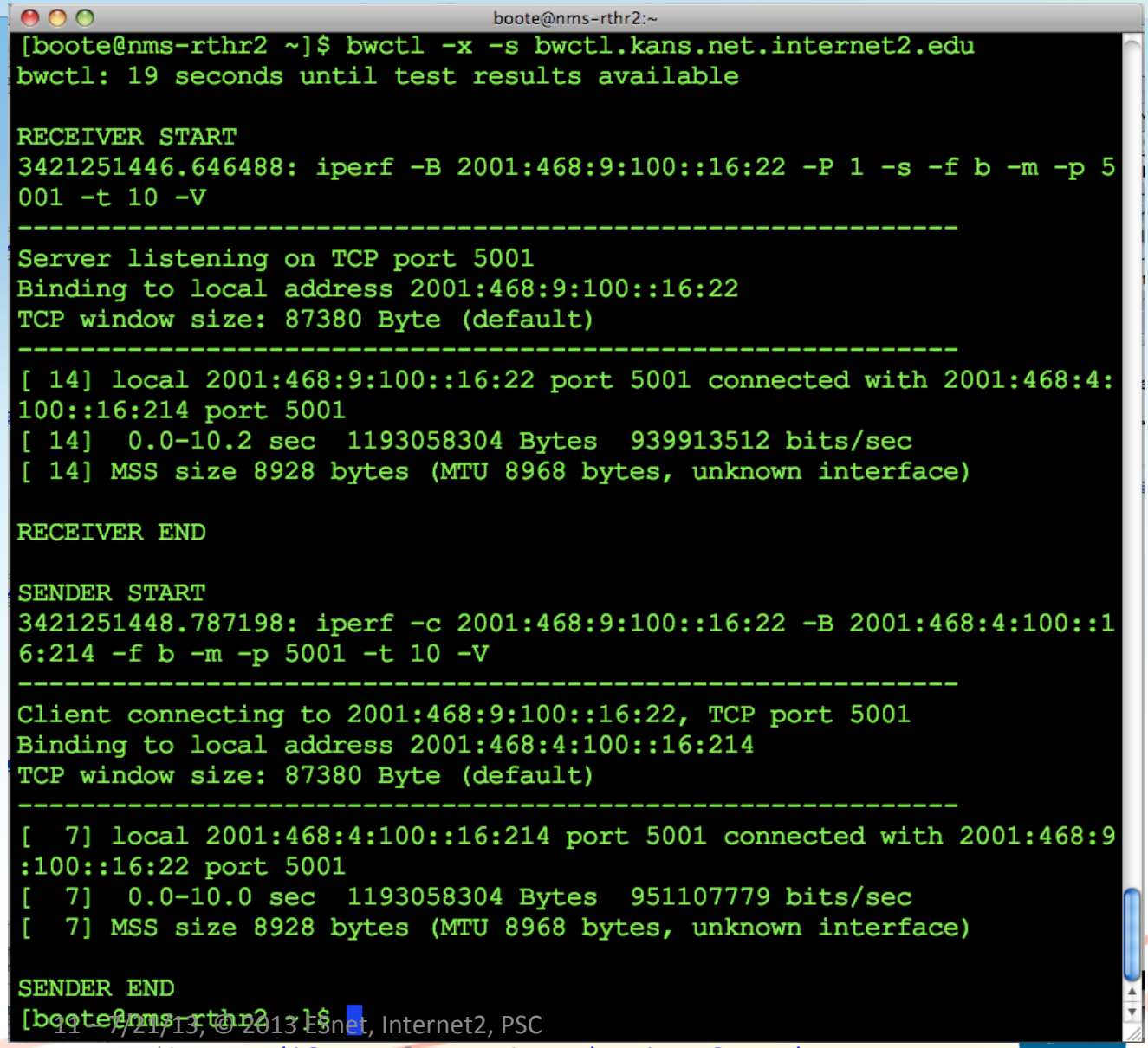
- UDP doesn't break up writes, don't exceed Path MTU

UDP Measurements

- UDP provides greater transparency
- We can directly measure some things TCP hides
  - ***Loss***
  - Jitter
  - Out of order delivery
- Use -b to specify target bandwidth
  - Default is 1M
  - Two sets of multipliers
    - k, m, g multipliers are 1000, 1000^2,1000^3
    - K, M, G multipliers are 1024, 1024^2,1024^3
  - Eg, -b 1m is 1,000,000 bits per second

J. Zurawski – zurawski@es.net & K. Benninger - benninger@psc.edu

Example



J. Zurawski – zurawski@es.net & K. Benninger - benninger@psc.edu

BWCTL GUIs

pS-Performance Node – Throughput Tests

https://desk172.internet2.edu/toolkit/gui/perfAdmin/serviceTest.cgi?url=http://localhost:8085/perfSONAR_PS/services/pSB&ev

Google

performance pS toolkit

User Tools
- Local Performance Services
- Global Performance Services
- Java OWAMP Client
- Reverse Traceroute
- Reverse Ping
- PingER Web GUI

Service Graphs
- Throughput
- One-Way Latency
- Ping Latency
- SNMP Utilization
- Cacti Graphs

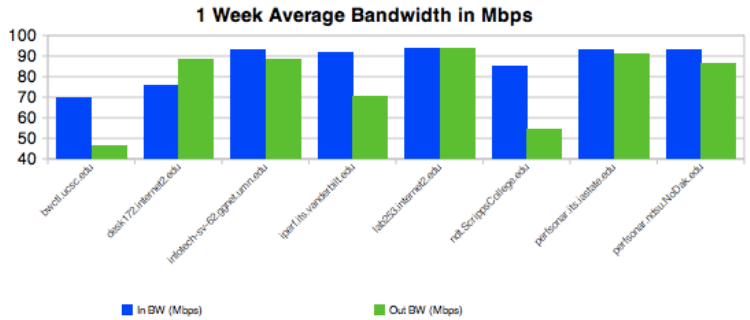
Toolkit Administration
- Administrative Information
- External BWCTL Limits
- External OWAMP Limits
- Enabled Services
- NTP
- Scheduled Tests
- Cacti SNMP Monitoring

Performance Toolkit
- Configuration Help
- Frequently Asked Questions
- About
- Credits

Throughput Tests

Active Data Sets

First Host	First Address	Second Host	Second Address	Protocol	Duration	Window Size	Bandwidth Limit	Bi-Directional	Line Graph	Scatter Graph
bwctl.ucsc.edu	128.114.0.205	desk172.internet2.edu	207.75.164.172	TCP	20			Yes	-- Select --	-- Select --
desk172.internet2.edu	207.75.164.172	infotech-sv-62.ggnet.umn.edu	146.57.255.17	TCP	20			Yes	-- Select --	-- Select --
desk172.internet2.edu	207.75.164.172	iperf.its.vanderbilt.edu	192.111.110.34	TCP	20			Yes	-- Select --	-- Select --
desk172.internet2.edu	207.75.164.172	lab253.internet2.edu	207.75.164.253	TCP	20			Yes	-- Select --	-- Select --
desk172.internet2.edu	207.75.164.172	ndt.ScrippsCollege.edu	134.173.151.207	TCP	20			Yes	-- Select --	-- Select --
desk172.internet2.edu	207.75.164.172	perfsonar.its.iastate.edu	129.186.6.241	TCP	20			Yes	-- Select --	-- Select --
desk172.internet2.edu	207.75.164.172	perfsonar.ndsu.NoDak.edu	134.129.90.1	TCP	20			Yes	-- Select --	-- Select --



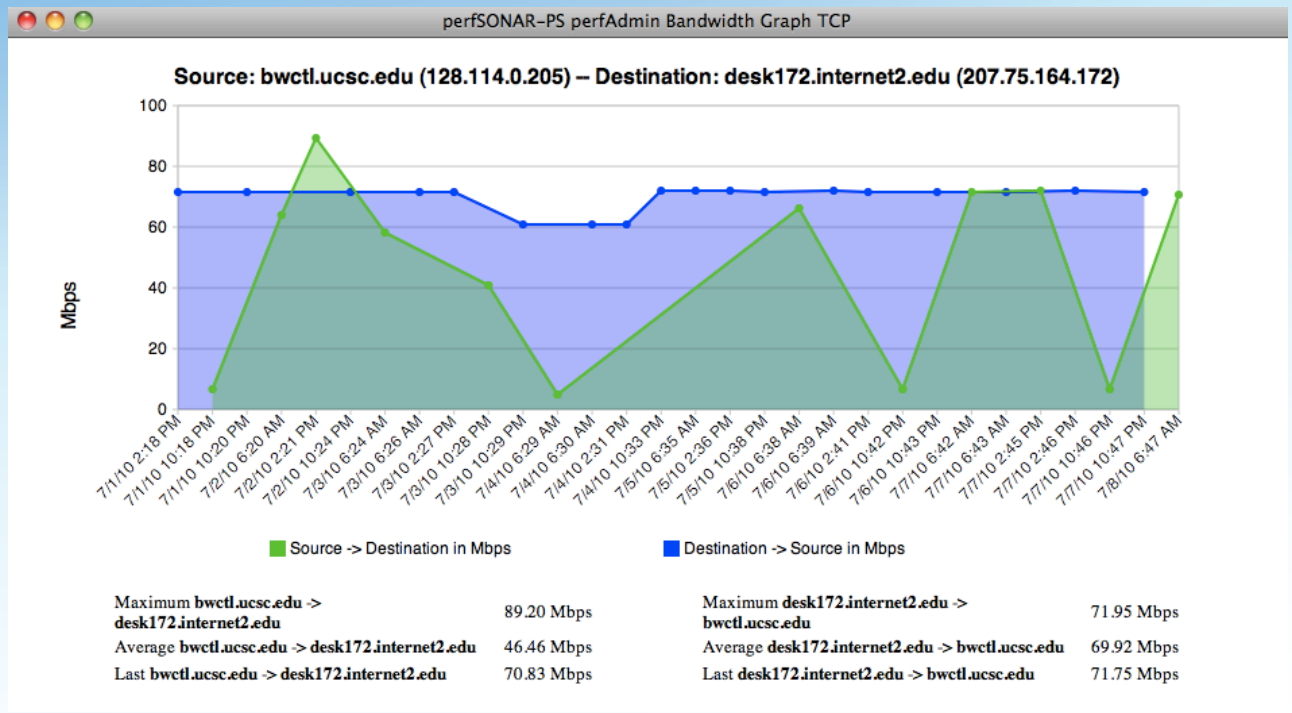
Non-Active Data Sets

First Host	First Address	Second Host	Second Address	Protocol	Duration	Window Size	Bandwidth Limit	Bi-Directional	Line Graph	Scatter Graph

perfSONAR powered

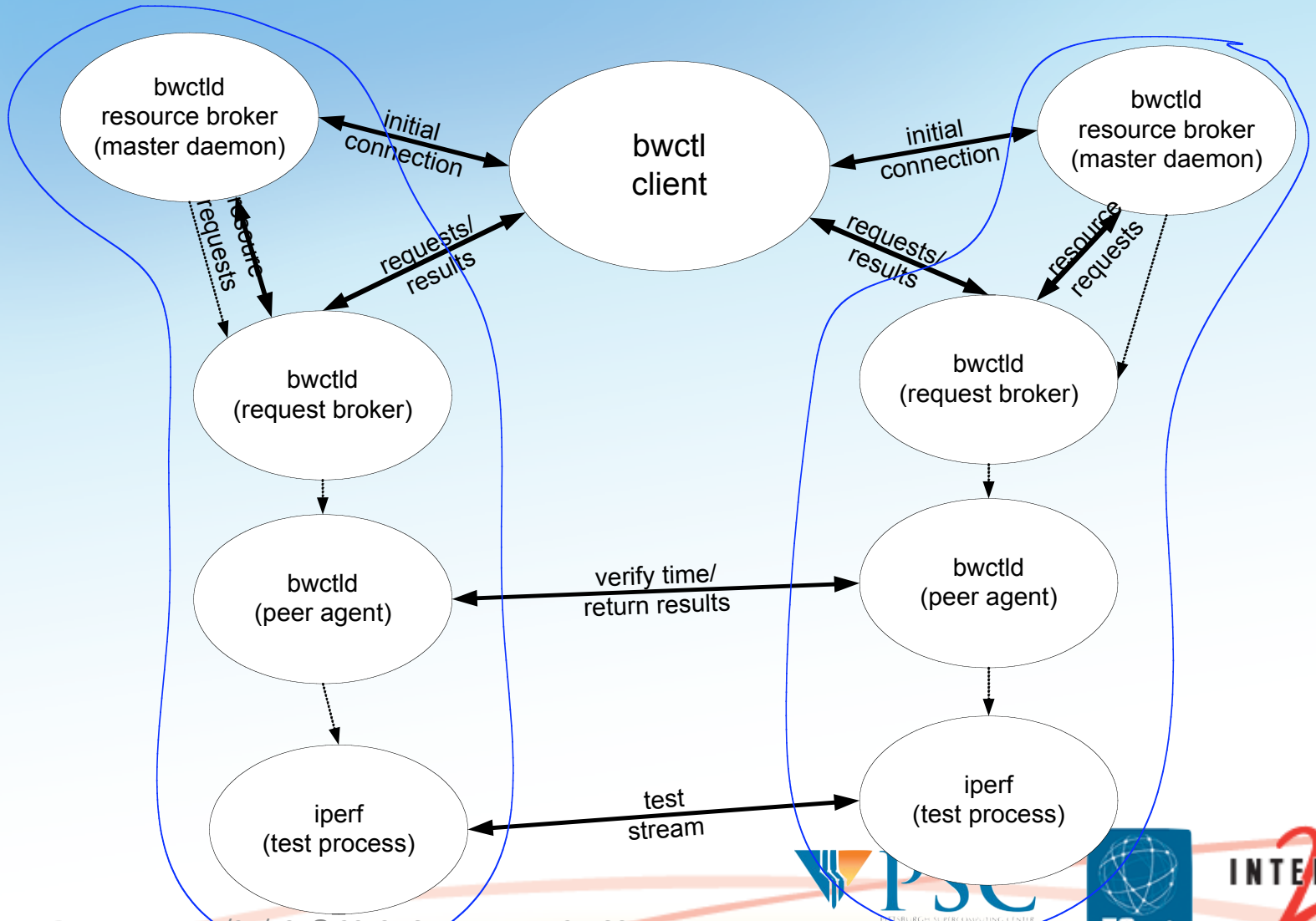
J. Zurawski – zurawski@es.net & K. Benninger - benninger@psc.edu

BWCTL GUIs



J. Zurawski – zurawski@es.net & K. Benninger - benninger@psc.edu

3rd Party Testing



J. Zurawski – zurawski@es.net & K. Benninger - benninger@psc.edu

General Requirements

- iperf version 2.0.x
  - Support for iperf 3 is being integrated
- NTP (ntpd) synchronized clock on the local system
  - Used for scheduling
  - More important that errors are accurate than the clock itself
- Firewalls:
  - Lots of ports for communication and testing – see the web for specifics
- End hosts must be tuned!
  - http://fasterdata.es.net/fasterdata/host-tuning
  - http://www.psc.edu/index.php/tcp-performance-tuning

Supported Systems

- Source Code
  - All modern Unix distributions (Free BSD/Linux)
  - OS X
- Packages
  - Support for CentOS 5.x and 6.x (x86 and 64 Bit)
  - Packages have been shown to operate on similar systems (Fedora, RHEL, SL)
  - Avoid ‘alien’ on non-RHEL lineage, stick with source

Security & Policy Considerations

- DoS source
  - Imagine a large number of compromised BWCTLD servers being used to direct traffic
- DoS target
  - Someone might attempt to affect statistics web pages to see how much impact they can have
- Resource consumption
  - Time slots
  - Network bandwidth
- Policy
  - Restrictive for UDP
    - Allow between peers
    - Limit bandwidth, and time of tests
  - More liberal for TCP tests
    - Open for all (or peers)
    - Limit length of tests

Availability

- Currently available
  - http://www.internet2.edu/performance/bwctl
  - http://software.internet2.edu
- Mail lists:
  - https://lists.internet2.edu/sympa/info/bwctl-users
    - bwctl-users@internet2.edu
  - https://lists.internet2.edu/sympa/info/bwctl-announce
    - bwctl-announce@internet2.edu

“Advanced” Use case/Debugging

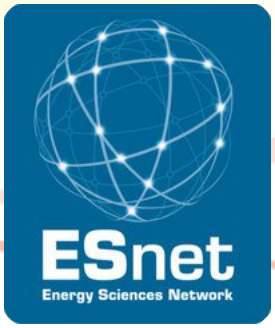
- Sometimes you really want to have more control over the client and server
  - E.g. BWCTL does a good job of automation and simple tasks in throughput calculations
- IPERF/NUTTCP can be used alone (although you need control over a client and server on each end)
- The following are some examples of how to use it.
  - Learning to use these by hand will help you debug issues in the future

Iperf Usage

- Switches:
  - -f m (Mbps output)
  - -t 30 (30 second test)
  - -i 1 (1 second interval)
  - -p 5131 (port)
  - -c (client)
  - -s (server)
  - -u (UDP mode)
  - -b 90m (rate limit for UDP)
  - -w 8M (set a window [client or server])
  - -P 4 (parallel streams)
- Things to try:
  - Effect of setting the window (on well tuned or not well tuned hosts)
  - Parallel streams
  - Links between other hosts

Nuttcp Usage

- Switches:
  - -S (server mode)
  - --nofork (don't background)
  - -P/-p (Ports for data/control channel)
  - -T 30 (30 second test)
  - -i 1 (1 second reporting interval)
  - -r (reverse direction)
  - -N 4 (Parallel streams)
  - -u (UDP node)
  - -R 90m (rate limit for UDP)
  - -w 8M (TCP window)
- Things to Try:
  - Effect of setting the window (on well tuned or not well tuned hosts)
  - Parallel streams
  - Links between other hosts



BWCTL, IPERF, & NUTTCP (Oh My)

July 22nd 2013, XSEDE Network Performance Tutorial

Jason Zurawski – Internet2/ESnet
Kathy Benninger - Pittsburgh Supercomputing Center

For more information, visit http://www.internet2.edu/workshops/npw

perfSONAR powered