Lanka Education and Research Network

Network Measurements and Troubleshooting

tools

Thilina Pathirana

based on INTERNET2 public information

"The Network is broken"

- How can your users effectively report problems?
  - And how can you learn to take them seriously...
- How can users and the local administrators effectively solve multi domain problems?
  - Eliminate the 'who you know' network to finding resources
  - Automate things when applicable
- Network as an instrument – should be as easy to use as possible
  - Smarter applications
  - Less 'friction'
- Components:
  - Tools to use
  - Methodology to follow
  - Questions to ask
  - How to ask for (and receive) help

Motivation

- Proactive vs Reactive Positions
  - Do you want to find problems before the users do?
  - Can monitoring tools help in other aspects of operations?
    - Capacity Planning
    - Scheduling Maintenance
    - Traffic Engineering

- “The Network is broken”, Is this justifiable?
  - In actuality, there is a lot of “network” between the applications
  - What about those applications?
  - What about the host itself?

A typical scenario

- User wants to access a file at the resource (e.g. ~600MB)
- Plans to use simple tools (e.g. “scp”, but could easily be something scientific like “GridFTP” or simple like a web browser)
- What are the expectations?
  - 1Gbps network (e.g. *bottleneck* speed on the LAN)
  - 600MB * 8 = 4,800 Mb file
  - User expects *line rate*, e.g. 4,800 Mb / 1000 Mbps = 4.8 Sec

Is this expectation too high?

- What are the realities?
  - Congestion and other network performance factors
  - Host performance
  - Protocol Performance
  - Application performance

A typical scenario

- 1MB/s (8Mb/s) ??? 10 Minutes to transfer???
- Seems unreasonable given the investment in technology
  - Backbone network – High speed LAN
  - Capable hosts
- Performance realities as network speed decreases:
  - 100MbpsSpeed – 48Seconds
  - 10 Mbps Speed – 8 Minutes
  - 1MbpsSpeed – 80Minutes
- How could this happen? More importantly, why are there not more complaints?
- Would you complain? If so, to whom?
- Brainstorming the above – where should we look to fix this?

A typical scenario

Expectation does not even come close to experience, time to debug. Where to start though?

– Application
Have other users reported problems? Is this the most up to date version?

– Protocol
Protocols typically can be tuned on an individual basis, consult your operating system.

– Host
Are the hardware components (network card, system internals) and software (drivers, operating system) functioning as they should be?

– LAN Networks
Consult with the local administrators on status and potential choke points

– Backbone Network
Consult the administrators at remote locations on status and potential choke points (Caveat – do you [should you] know who they are?)

A typical scenario

Following through on the previous, what normally happens ...

– Application
This step is normally skipped, the application designer will *blame the network*

– Protocol
These settings may not be explored. Shouldn't this be automatic (e.g. auto tuning)?

– Host
Checking and diagnostic steps normally stop after establishing connectivity.
E.g. "can I ping the other side"

– LAN Networks
Will assure "internal" performance, but LAN administrators will ignore most user complaints and shift blame to upstream sources. E.g. "our network is fine, there are no complaints"

– Backbone Network
Will assure "internal" performance, but Backbone responsibilities normally stop at the demarcation point, blame is shifted to other networks up and down stream

Why Worry About Network Performance?

- Most network design lends itself to the introduction of flaws:
  - Mixed equipment
  - Cost factors heavily into design – e.g. *Get what you pay for*
  - Design heavily favors ***protection*** and ***availability*** over performance
- Communication protocols are not advancing as fast as networks
  - *TCP/IP* is the king of the protocol stack

    Guarantees reliable transfers
    Adjusts to failures in the network

    Adjusts speed to be *fair* for all
- User Expectations

    Big Science is prevalent globally

    "The Network is Slow/Broken" – is this the response to almost any problem?
    Hardware? Software?

    Empower users to be more informed/more helpful

The Concerns

• Network Design
– Balancing the needs of all users (e.g. how does video differ from bulk data transfer)

– An ounce of prevention (e.g. configuration, monitoring)

– You care about your network, is it your job to care about the network of your peers?

• Packet Loss
– “Congestive”; the realities of a general purpose network
– “Non-Congestive”; fixable, if you can find it

Clean your fibers!
Throw away the crimped cable!
Increase your buffers!

The Network Design

- LAN vs WAN Design

– Multiple Gbit flows [to the outside] should be close to the WAN connection

– Eliminate the number of hops/devices/physical wires that may slow you down and add delay (buffering)

– Great performance on the LAN != Great performance on the WAN

  – Think about how TCP works – latency plays a big role in recovering from loss

- *You Get What you Pay For*

– Inexpensive equipment will let you down

– What could go wrong?

  Small buffers, potentially shared, creates questionable performance (e.g. internal switching fabric can't keep up demands)

  Lack of diagnostic tools (SNMP, etc.)

- Default configurations are (***always***) bad

  Hosts, Switches/Routers

Firewalls/ shapers

Designed to stop ‘traffic’
– Read this slowly a couple of times...
– Performing a read of headers and/or data. Matching signatures

Contain small buffers
– Concerned with protecting the network, not impacting your performance

Will be ***a lot*** slower than the original wire speed

– A “***10G Firewall***” may handle 1 flow close to 10G, doubtful that it can handle a couple.

If *firewall-like* functionality is a must – consider using router filters instead

– Or per host firewall configurations ...

Packet Loss

- Bandwidth Delay Product & Buffering

– The amount of “in flight” data allowed for a TCP connection
– BDP = bandwidth * round trip time

Example: 1Gb/s cross country, ~100ms
1,000,000,000 b/s * .1 s = 100,000,000 bits
100,000,000 / 8 = 12,500,000 bytes
12,500,000 bytes / (1024*1024) ~ 12MB

- “Buffer Bloat”

– Less of a concern in the R&E community; the added delay you get with too much buffering on a (low speed) connection

• TCP Dynamics (e.g. congestion control algorithms)
– Additive-increase/Multiplicative-decrease[AIMD]
E.g. You cut your speed in half (sometimes less) with each loss.
– Slowly increase to your prior speed and hope you don’t take more loss.
Think about a short path with a lot of loss
Think about a long path with little loss

Configuration

- Host Configuration

– Tune your hosts (especially compute/storage!)
– Changes to several parameters can yield 4 – 10 x improvement
– Takes minutes to implement/test
– Instructions: http://fasterdata.es.net/tuning.html

- Network Switch/Router Configuration

– ***Out of the box*** configuration may include small buffers
– Competing Goals: Video/Audio etc. needs small buffers to remain responsive. Science(big data) flows need large buffers to push more data into the network.
– Read your manuals and test LAN host to a WAN host to verify (not LAN to LAN).

Soft failures

• ***Soft Failures*** are any network problem that does not result in a loss of connectivity

– Slows down a connection

– Hard to diagnose and find

– May go unnoticed by LAN users in some cases, but remote users may be the ones complaining

• Caveat

– How much time/energy do you put into listening to complaints of remote users?

• Common:
– Dirty or Crimped Cables
– Failing Optics/Interfaces
– [Router] Process Switching, aka "*Punting*"
– Router Configuration (Buffers/Queues)

Troubleshoot

- **Diagnosis Methodology**

– Find a measurement server “near me”

  - Why is this important?
  - How hard is this to do?

– Encourage user to participate in diagnosis procedures

– Detect and report common faults in a manner that can be shared with admins/NOC

  - ‘Proof’ goes a long way

– Provide a mechanism for admins to review test results

– Provide feedback to user to ensure problems are resolved

Troubleshoot

- **Systematic Troubleshooting**

– Having tools deployed (along the entire path) to enable adequate troubleshooting

– Getting end-users involved in the testing

– Combining output from multiple tools to understand problem
    - Correlating diverse datasets–only way to understand complex problems.

– Ensuring that results are adequately documented for later review

- **On Demand vs Regular Testing**

– On-Demand testing can help solve existing problems once they occur
– Regular performance monitoring can quickly identify and locate problems before users complain
    - Alarms
    - Anomaly detection

– Testing and measuring performance increases the value of the network to all participants

PING

The most commonly used network tool is the ping utility.

This utility is used to provide a basic connectivity test between the requesting host and a destination host.

This is done by using the Internet Control Message Protocol (ICMP) which has the ability to send an echo packet to a destination host and a mechanism to listen for a response from this host.

Simply stated, if the requesting host receives a response from the destination host, this host is reachable. This utility is commonly used to provide a basic picture of where a specific networking problem may exist.

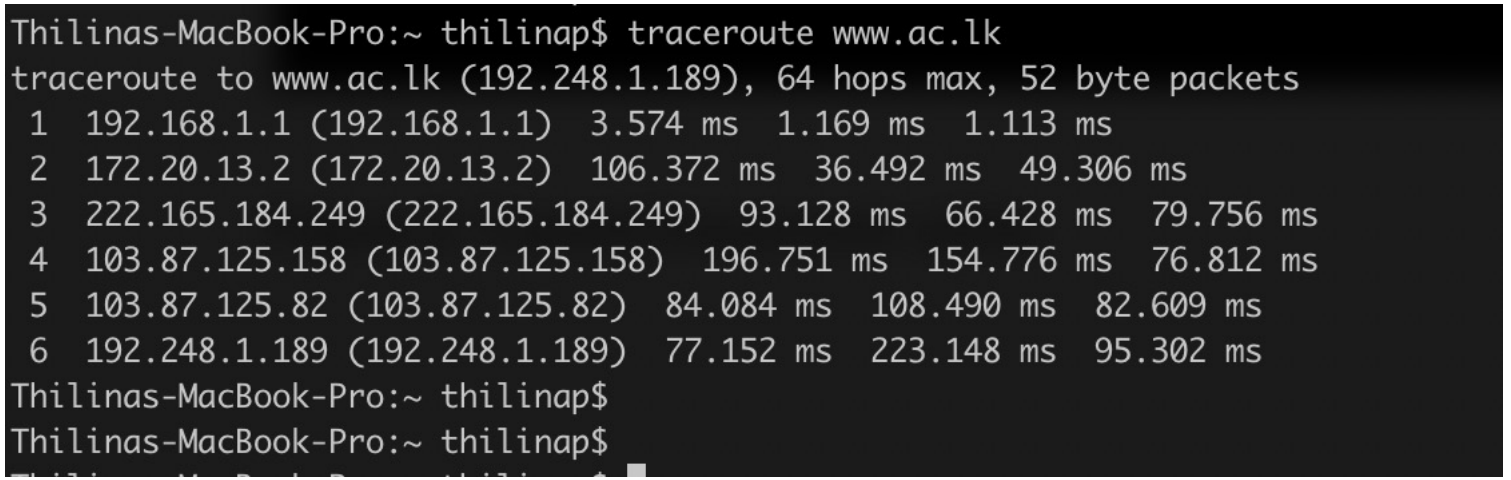
- ping 192.168.1.30
- ping www.thilinapathirana.xyz
- ping 2401:dd00:1::161
- PING4 / PING6

Tracert / traceroute

Typically, once the ping utility has been used to determine basic connectivity, the tracert/traceroute utility can used to determine more specific information about the path to the destination host including the route the packet takes and the response time of these intermediate hosts.

The tracert utility and traceroute utilities perform the same function but operate on different operating systems, Tracert for Windows machines and traceroute for Linux/*nix based machines.

Following figure is an example trace route to www.ac.lk from a SLT4G



What else?

nslookup

netstat

mtr

arp

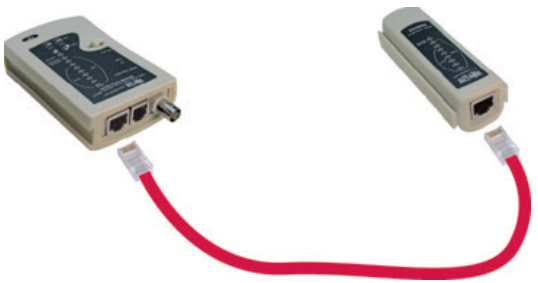
LibreNMS

https://smokeping.learn.ac.lk/cgi-bin/smokeping.cgi?target=Access

Perfsonar

www.speedtest.net

Hardware tools



What is perfSONAR?

Collection of open source software for performing and sharing end-to-end network measurements.

Tools include:

- owamp - A tool used for measuring packet loss and one-way delay.
- iperf- A tool used to measure network throughput and associated metrics.
- nuttcp - Another throughput tool with some useful options not found in other tools.
- traceroute - The classic packet trace tool used in identifying network paths
- tracepath - Another path trace tool that also measures path MTU
- paris-traceroute - A packet trace tool that attempts to identify paths in the presence of load balancers
- ping - The classic utility for determining reachability, round-trip time (RTT) and basic packet loss.
- bwctl – classic bandwidth measurement scheduling tool.

perfSONAR hardware

http://docs.perfsonar.net/install_hardware.html

Dedicated perfSONAR hardware is best

- Server class is a good choice
- Desktop/Laptop/Mini (Mac, Shuttle) can be problematic, but work in a diagnostic capacity
- Other applications running may perturb results (and measurement could hurt essential services)

Running Latency and Throughput on the Same Server

- If you can, devote 2 interfaces – the toolkit will support this.
- If you can't, note that Throughput tests can cause increased latency and loss (latency tests on a throughput host are still useful however)

perfSONAR hardware

Virtual Machines do not always work well as perfSONAR hosts

- Clock sync issues are a bit of a factor
- throughput is reduced significantly for 10G hosts
- VM technology and motherboard technology has come a long way
- NAGIOS/SNMP/1G Throughput are good choices for a VM, OWAMP/10G Throughput are not

perfSONAR hardware

For us in LK institutes,

as a start go with,

Processor:	Core i5 or higher
RAM:	4GB +
HDD:	500GB +
Network:	1G or depending on your Internet BW

Where to Place it?

Put it on borders

try to avoid firewalls / UTMs / IPS / rate limiters / traffic shapers

perfSONAR Installation

A CentOS netinstall is the quickest way to start

http://docs.perfsonar.net/install_centos_netinstall.html

Others explained here

http://docs.perfsonar.net/install_getting.html

There are two use cases for configuration:

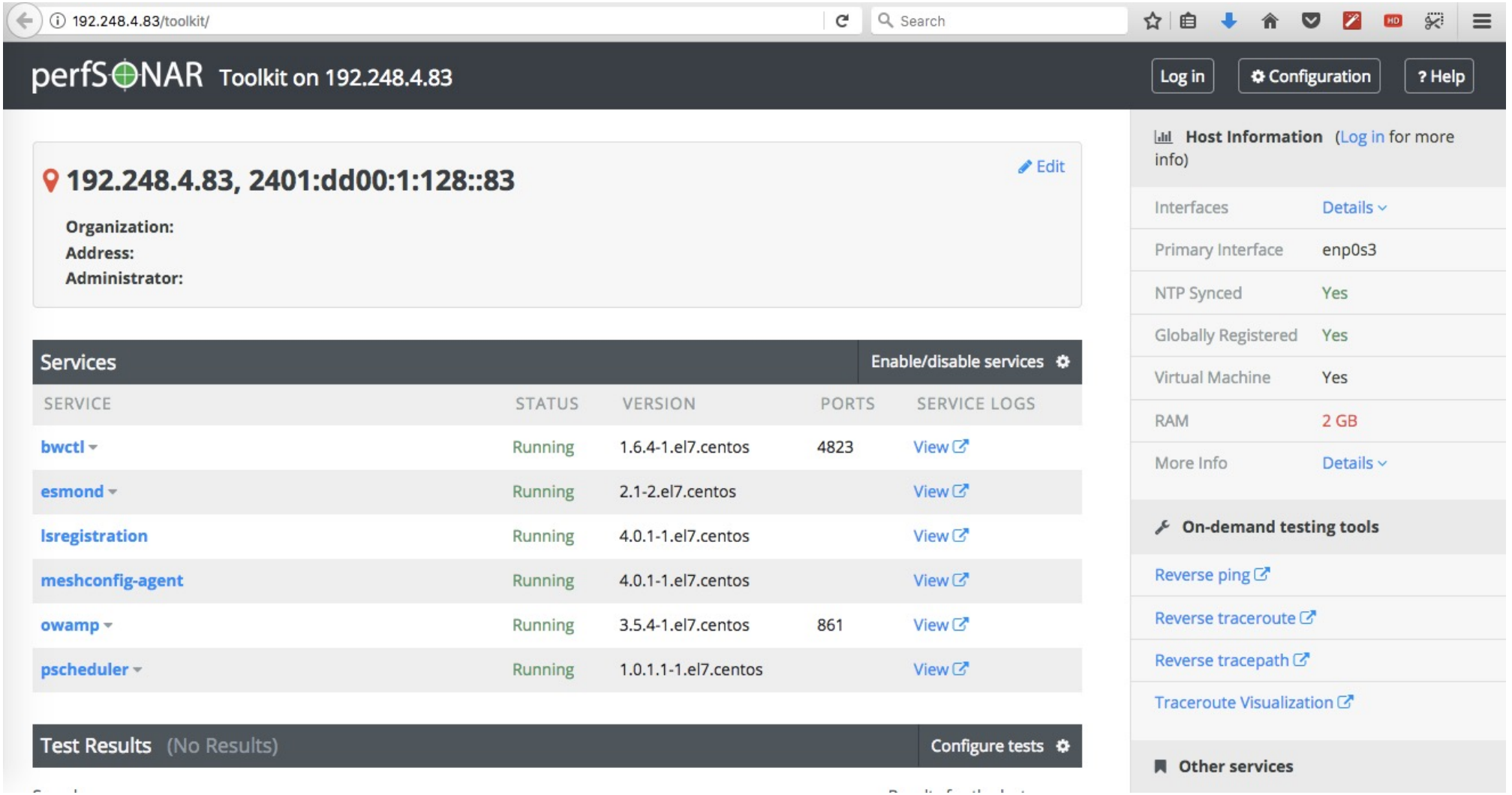
Installation is done via the use of an entire distribution via ISO, or use of RPM packages

Diagnostic: Configure host to allow tests from others, don't test on your own (e.g. beacon)

Permanent: Participate in a mesh, or configure tests as an island

perfSONAR Installation

Once your installation is finished log in to the web console.



NTP configurations

Scheduling, and one-way latency (OWAMP) depend on good knowledge of local time

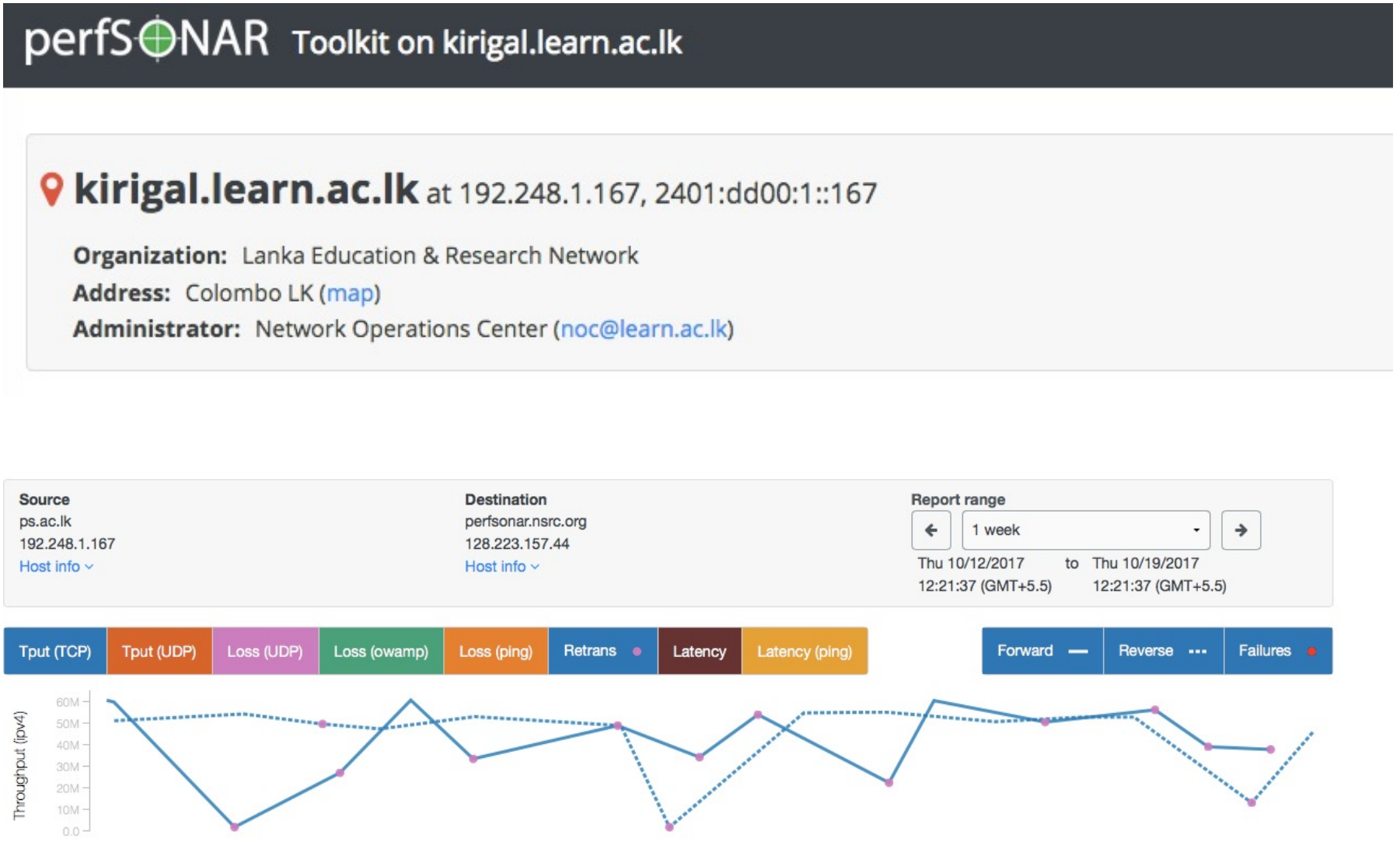
Note that it may take a day to fully stabilize the clock

Pick 4 – 5 Close servers for NTP

We have a fast way to do this, or you can manually select

Can also add your own servers if you don't like system defaults

Measurements



Regular testings

We can’t wait for users to report problems and then fix them (soft failures can go unreported for years!)

Things just break sometimes

Failing optics

Somebody messed around in a patch panel and kinked a fiber

Hardware goes bad

Problems that get fixed have a way of coming back

System defaults come back after hardware/software upgrades

New employees may not know why the previous employee set things up a certain way and back out fixes

Important to continually collect, archive, and alert on active throughput test results

Regular testing

There are a few ways putting this.

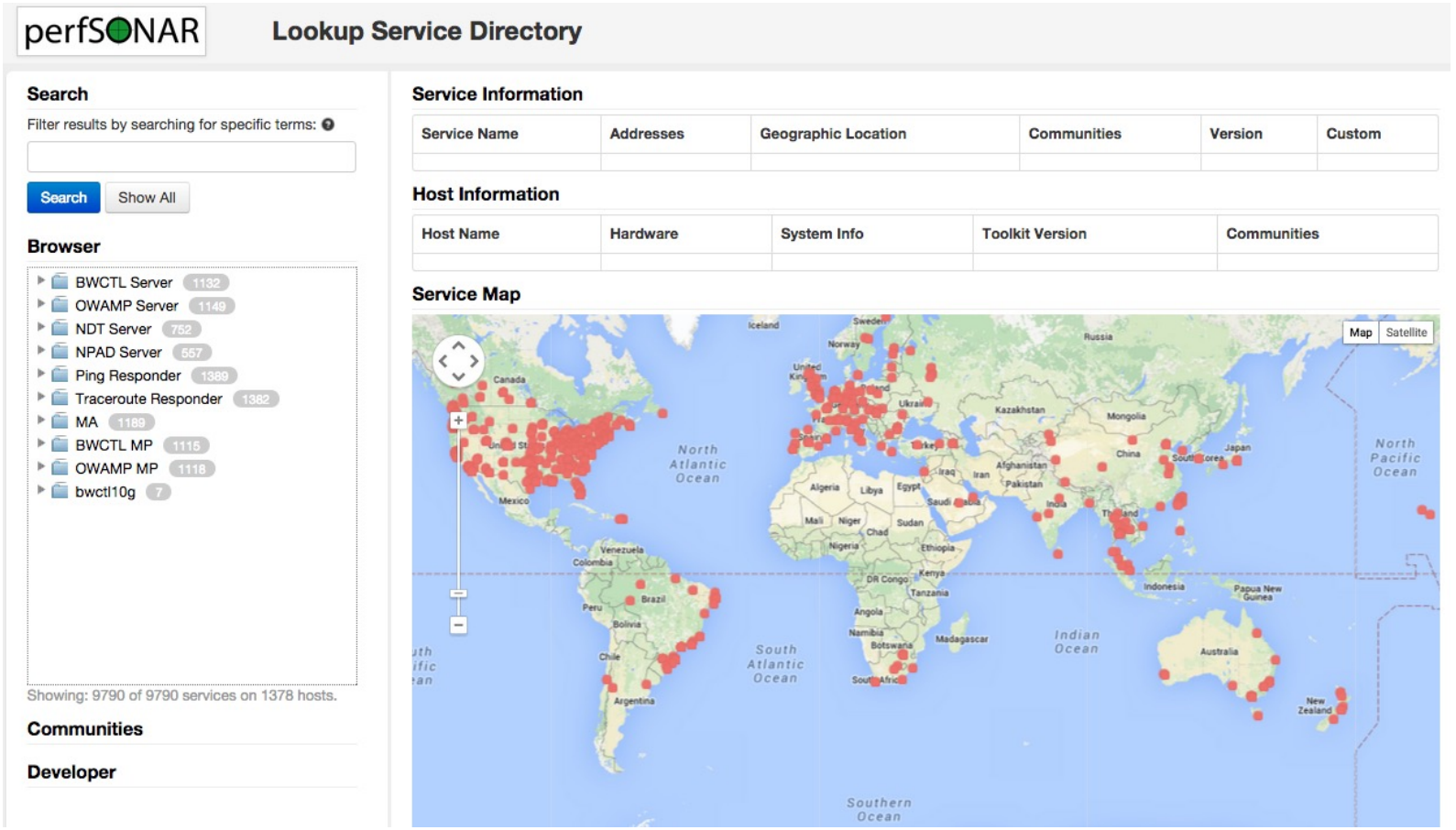
Beacon: Let others test to you (e.g. no regular configuration is needed)

Island: Pick some hosts to test to – you store the data locally. No coordination with others is needed. Can be done via web interface

Mesh: full advanced coordination between you and others (e.g. consume a testing configuration that includes tests to everyone, and incorporate into a visualization)

Do it with a plan

http://stats.es.net/ServicesDirectory/



Tool - IPERF

Start with a definition:

- ***network throughput*** is the rate of successful message delivery over a communication channel
- Easier terms: how much data can I shovel into the network for some given amount of time

What does this tell us?

- Opposite of utilization (e.g. its how much we can get at a given point in time, minus what is utilized)
- Utilization and throughput added together are capacity

Tools that measure throughput are a simulation of a real work use case (e.g. how well could bulk data movement perform)

Ways to game the system

- Parallel streams
- Manual window size adjustments
- 'memory to memory' testing – no spinning disk

Tool - IPERF

Couple of varieties of tester that BWCTL (the control/policy wrapper) knows how to talk with:

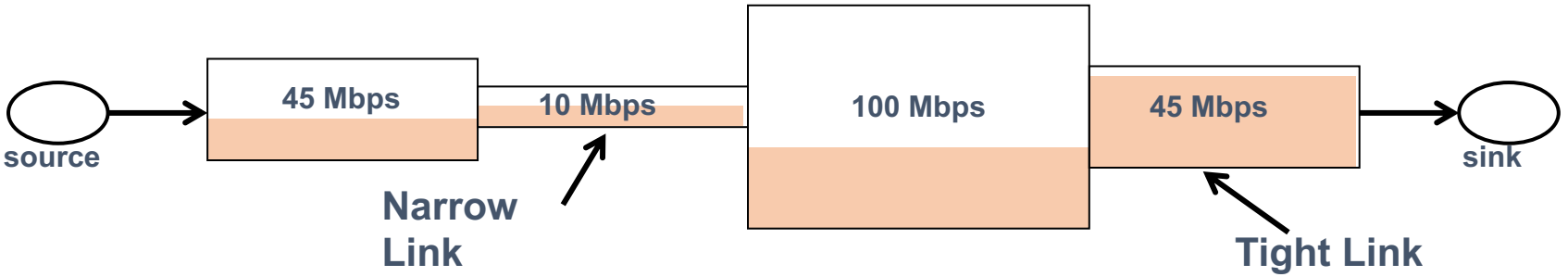
- Iperf2
  - Default for the command line (e.g. bwctl –c HOST will invoke this)
  - Some known behavioral problems (Older versions were CPU bound, hard to get UDP testing to be correct)
- Iperf3
  - Default for the perfSONAR regular testing framework, can invoke via command line switch (bwctl –T iperf3 –c HOST)
  - Note: Single threaded, so performance is gated on clock speed. Parallel stream testing is hard as a result (e.g. performance is bound to one core)

Tool – IPERF – what does it say

Lets start by describing throughput, which is vague.

- Capacity: link speed
  - Narrow Link: link with the lowest capacity along a path
  - Capacity of the end-to-end path = capacity of the narrow link
- Utilized bandwidth: current traffic load
- Available bandwidth: capacity – utilized bandwidth
  - Tight Link: link with the least available bandwidth in a path
- Achievable bandwidth: includes protocol and host issues (e.g. BDP!)

All of this is “memory to memory”, e.g. we are not involving a spinning disk (more later)



Tool – BWCTL

BWCTL is the wrapper around a couple of tools (we will show the throughput tools as for this time)

Policy specification can do things like prevent tests to subnets, or allow longer tests to others. See the man pages for more details

Some general notes:

- Use '-c' to specify a 'catcher' (receiver)
- Use '-s' to specify a 'sender'
- Will default to IPv6 if available (use -4 to force IPv4 as needed, or specify things in terms of an address if your host names are dual homed)
- The defaults are to be '-f m' (Megabits per second) and '-t 10' (10 second test)
- The '—omit X' flag can be used to parse off the TCP slow start data from the final results

IPERF & BWCTL examples

- Try these within you and neighbors
- **iperf –s** Start server
- **iperf –s –D** Run as daemon
- **iperf -c receive_host** Connects to a server for testing
- **iperf -c receive_host –d** bi-directional test at the same time
- **iperf -c receive_host –r** bi-directional after one another
- **iperf -c receive_host -w 2048** change window size
- **iperf -s -w 2048** Start server with window size 2048
- **iperf -c receive_host -p 9009** Change port number
- **iperf -s -p 9009** Start server on port 9000
- **iperf -c receive_host –t 60** Change test time in seconds
- **iperf –s –u** Start server for UDP testing
- **iperf –c receive_host –u** UDP testing
- **iperf –c receive_host –P 4** run multiple threads

IPERF & BWCTL examples

- Try these within you and neighbors
- **bwctl -c receive_host** Run a standard test
- **bwctl –x -c receive_host** Run a standard test, output both send/ receive
- **bwctl -c receive_host -t 30 -i 2** Run a 30 second test, and output every 2 secs
- **bwctl -c receive_host -t 30 -w 64M** Run a 30Sec test with TCP window 64MB.
- **bwctl -L 1000 -c receive_host** By default bwctl exits if it can not get a test slot within 5 minutes. Use the -L flag for heavily used servers where you might have to wait longer.
- **bwctl -c receive_host -i 1 -b 500M** Do a 500Mbps test.
- **bwctl -c receive_host –T iperf** Use tool iperf instead of default iperf3.
- **bwctl -c receive_host -s send_host -O 5 -t 20** Use iperf3 instead of iperf, and omit the first 5 seconds of a 20 second test (removes TCP slowstart)
- **bwctl -4 -c receive_host -s send_host** Force test to use IPv4 instead of IPv6

Thank You

Thilina Pathirana

Email: thilina@learn.ac.lk