perfSONAR: Better knowledge, better networks, better investments

Eric Boyd
Internet2

2014 Technology Exchange
10-29-2014
Problem Statement

- The global Research & Education network ecosystem is comprised of hundreds of international, national,
Problem Statement

• While these networks all interconnect, each network is owned and operated by separate organizations (called “domains”) with different policies, customers, funding models, hardware, bandwidth and configurations.
Problem Statement

- This complex, heterogeneous set of networks must operate seamlessly from “end to end” to support science and research collaborations that are distributed globally.
Problem Statement

• In practice, performance issues are prevalent and distributed.
• When a network is underperforming or errors occur, it is difficult to identify the source, as problems can happen anywhere, in any domain.
• Local-area network testing is not sufficient, as errors can occur between networks.
Problem Statement: Hard vs. Soft Failures

• “Hard failures” are the kind of problems every organization understands
  – Fiber cut
  – Power failure takes down routers
  – Hardware ceases to function

• Classic monitoring systems are good at alerting hard failures
  – i.e., NOC sees something turn red on their screen
  – Engineers paged by monitoring systems

• “Soft failures” are different and often go undetected
  – Basic connectivity (ping, traceroute, web pages, email) works
  – Performance is just poor

• How much should we care about soft failures?
Soft Failures Cause Packet Loss and Degraded TCP Performance

Throughput vs. Increasing Latency with .0046% Packet Loss

With loss, high performance beyond metro distances is essentially impossible

Measured (TCP Reno)  Measured (HTCP)  Theoretical (TCP Reno)  Measured (no loss)
The solution: perfSONAR

- A widely-deployed test and measurement infrastructure used by local, regional and national research networks as well as science facilities to actively track and troubleshoot network performance issues.

- An open-source, community-developed software suite which:
  - Provides a consistent set of proven tools to pinpoint and resolve network performance issues including soft failures across complex, multi-domain data paths.
  - Allows you to set network performance expectations with your stakeholders and make informed decisions on future network investments.
  - Creates a standard way to visualize, publish, and archive network metrics and data for future analysis, and aids other networks in debugging issues.
Benefits: Finding the needle in the haystack

• Above all, perfSONAR allows you to maintain a healthy, high-performing network because it helps identify the “soft failures” in the network path.
  – Classical monitoring systems have limitations
    • Performance problems are often only visible at the ends
    • Individual network components (e.g. routers) have no knowledge of end host state
  – perfSONAR tests the network in ways that classical monitoring systems do not
• More perfSONAR distributions equal better network visibility.
Benefit: Finding the needle in the haystack

• perfSONAR is designed to pinpoint and identify soft failures to accelerate resolution.

• Example: Find and replace failing optics
Benefit: Finding the needle in the haystack

- perfSONAR is designed to pinpoint and identify soft failures to accelerate resolution.

- Example: Host Tuning

![Image of perfSONAR BWCTL Graph]

MTU Changed to 9000
TCP Window settings changed
MTU = 1500 on 10G Host

10s BWCTL TCP Testing
Timezone: GMT-0400 (EDT)
Benefit: Finding the needle in the haystack

Inside the firewall
- One direction severely impacted by firewall
- Not useful for science data

Outside the firewall
- Good performance in both directions
Who is running perfSONAR?

- Currently close to 1200 deployments world-wide, and around 400 have already upgraded to version 3.4

http://stats.es.net/ServicesDirectory/
Benefit: Visibility – perfSONAR Dashboard

Status at-a-glance
• Packet loss
• Throughput
• Correctness

Current live instances at
http://pas.net.internet2.edu/
http://ps-dashboard.es.net/

Drill-down capabilities:
• Test history between hosts
• Ability to correlate with other events
• Very valuable for fault localization and isolation
Benefit: Investment Planning, Verification

• Test data and metrics verifies performance of current investments
  – Can you get full use out of the carrier circuit you’re paying for?
  – If performance suddenly changes, what caused it?
• Data informs future investments
  – Put resources in places with demonstrated need
  – Example: if your firewall is limiting performance, buying more bandwidth won’t solve the problem
• Make the case for architectural improvements
  – How might a Science DMZ or alternate architecture change performance?
  – How to evaluate potential purchases?
    • Put perfSONAR hosts on them and run tests
    • Prototype new model
Benefit: Improved Responsiveness

- Test data allows for expectation management
  - Communicate what is possible to stakeholders like users and senior management
- Efficient troubleshooting
  - If a customer is having problems, is it their machine or the network?
  - Focus scarce human cycles on problems quickly
- Localize troubleshooting
  - Networking is global
  - Often the problem isn’t local
  - perfSONAR helps determine area of responsibility
  - Again, directing human effort quickly and appropriately
Benefit: Active and Growing Community

• Active email lists and forums provide:
  – Instant access to advice and expertise from the community.
  – Ability to share metrics, experience and findings with others to help debug issues on a global scale.

• Joining the community automatically increases the reach and power of perfSONAR
In Closing...

• Long story made short,
  – you can’t fix what you can’t find
  – you can’t find what you can’t see
  – perfSONAR lets you see

• perfSONAR is a software tool that is:
  – flexible
  – provides active and passive network metrics
  – inexpensive, low risk to deploy
  – open-source
  – community-developed, community curated
Resources

- perfSONAR website
  - [http://www.perfsonar.net/](http://www.perfsonar.net/)
- perfSONAR mailing lists
  - [http://www.perfsonar.net/about/getting-help/](http://www.perfsonar.net/about/getting-help/)
- perfSONAR directory of users
  - [http://stats.es.net/ServicesDirectory/](http://stats.es.net/ServicesDirectory/)
- FasterData Knowledgebase
  - [http://fasterdata.es.net/](http://fasterdata.es.net/)
What is the Internet2 Performance Assurance Service (PAS)?

• Active monitoring framework for Internet2’s AL2S and AL3S and the services they support
  – 33 scheduled mesh test points
  – Continuous testing across all AL2S and AL3S nodes
  – 3 Ad-hoc test points

• Why so few ad-hoc test points?
  • There’s a better way to do it.

• Operationally focused, centered around the network’s ability to deliver services.
  – Identify real issues so we can fix them.
  – User reported = bad; pre-emptive detect/fix = good.
  – Make sure all the network is performing, all the time.
Internet2 Network
Advanced Layer2 Services
Topology Map
October 2014

Advanced Layer2 Service PoP
What does it look like?
What about ad-hoc again?

- Ad-hoc from location x, y, z on your network to a, b, c isn’t always the most effective way to detect network problems
  - Reactive, not proactive.
- Test your network *and monitor the results* all the time
  - We can focus on where the networks touch
- Partner meshes
  - Connectors and network partners can/should test across specific paths carefully chosen to isolate issues in connectivity between *networks*.
Why, who, how?

• You’re going to need some perfsonar test points
  – If you have some, awesome; if not, Internet2 can help
    • CTO’s office: rs@internet2.edu
• Once you have some test points:
  – Let’s do partner meshes:
    • noc@net.internet2.edu
  – We’re reaching out to connectors.
• Questions, comments?
  – daripley@grnoc.iu.edu, eboyd@internet2.edu
perfSONAR Toolkit v3.4: What’s new?

Brian Tierney – bltierney@es.net
Andy Lake– andy@es.net
ESnet
Lawrence Berkeley National Laboratory

2014 Technical Exchange
October 29, 2014
What is the perfSONAR Toolkit?

• Collection of open source tools that perform and publish network measurements

• Developed jointly by ESnet, Internet2, Indiana University, GEANT, and others
  – http://www.perfsonar.net

• Tools include
  – BWCTL for scheduling periodic throughout (iperf, iperf3), ping and traceroute tests
  – OWAMP for measuring one-way latency and packet loss
  – Esmond for storing measurements, summarizing results and making available via REST API
  – Lookup service for discovering other testers
  – On-demand test tools such as NDT
  – Configuration GUIs to assist with managing all of the above
  – Graphs to display measurement results
perfSONAR Toolkit 3.4

• New measurement archive
  – REST interface
  – 100x increase in query speed in many cases
  – Based on esmond (http://software.es.net/esmond)

• Support for dual-homed hosts
  – E.g: run owamp on 1G NIC, and iperf3 on 10G NIC

• Converged measurement scheduling
  – Same config file and daemon manage all regular measurements
  – Leverages features of BWCTL to run throughput, traceroute and ping tools

• New graphs
  – Show latency, loss and throughput data simultaneously
  – Show host NIC speed and MTU

• BWCTL and OWAMP Measurement Point
  – First step toward converged perfSONAR-PS and perfSONAR-MDM development

• Numerous other enhancements and fixes
  – Signed NDT applet
  – Web UI re-organization
  – Much more...
perfSONAR Deployment Trend

September 2014 - October 2014

Sep 24: pS Shellshock Announcement (1)
Sep 22: 3.4RC3
Sep 30: pS Shellshock Announcement (2)
Oct 7: 3.4 Release
Oct 1: pS Shellshock Announcement (Final)
Oct 15: POODLE Disclosure

3.3
3.4
Total
esmond (http://software.es.net/esmond)

• New perfSONAR measurement archive is esmond
  – Open Source time series database
    • Casandra + postgress backend
  – REST API
  – Configured to compute the following data summaries
    • owamp: 5 min, 1 hour, 1 day
    • iperf: 1 day
  – Over 100x faster than the old mySQL based MA
iperf3: http://software.es.net/iperf/

• iperf3 is a new implementation from scratch, with the goal of a smaller, simpler code base, and a library version of the functionality that can be used in other programs.

• Some new features in iperf3 include:
  – reports the number of TCP packets that were retransmitted and CWND
  – reports the average CPU utilization of the client and server (-V flag)
  – support for zero copy TCP (-Z flag)
  – JSON output format (-J flag)
  – “omit” flag: ignore the first N seconds in the results

• On RHEL-based hosts, just type ‘yum install iperf3’

Sample iperf3 output on lossy network

- Performance is < 1Mbps due to heavy packet loss

<table>
<thead>
<tr>
<th>ID</th>
<th>Interval</th>
<th>Transfer</th>
<th>Bandwidth</th>
<th>Retr</th>
<th>Cwnd</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>0.00-1.00 sec</td>
<td>139 MBytes</td>
<td>1.16 Gbits/sec</td>
<td>257</td>
<td>33.9 KBytes</td>
</tr>
<tr>
<td>15</td>
<td>1.00-2.00 sec</td>
<td>106 MBytes</td>
<td>891 Mbits/sec</td>
<td>138</td>
<td>26.9 KBytes</td>
</tr>
<tr>
<td>15</td>
<td>2.00-3.00 sec</td>
<td>105 MBytes</td>
<td>881 Mbits/sec</td>
<td>132</td>
<td>26.9 KBytes</td>
</tr>
<tr>
<td>15</td>
<td>3.00-4.00 sec</td>
<td>71.2 MBytes</td>
<td>598 Mbits/sec</td>
<td>161</td>
<td>15.6 KBytes</td>
</tr>
<tr>
<td>15</td>
<td>4.00-5.00 sec</td>
<td>110 MBytes</td>
<td>923 Mbits/sec</td>
<td>123</td>
<td>43.8 KBytes</td>
</tr>
<tr>
<td>15</td>
<td>5.00-6.00 sec</td>
<td>136 MBytes</td>
<td>1.14 Gbits/sec</td>
<td>122</td>
<td>58.0 KBytes</td>
</tr>
<tr>
<td>15</td>
<td>6.00-7.00 sec</td>
<td>88.8 MBytes</td>
<td>744 Mbits/sec</td>
<td>140</td>
<td>31.1 KBytes</td>
</tr>
<tr>
<td>15</td>
<td>7.00-8.00 sec</td>
<td>112 MBytes</td>
<td>944 Mbits/sec</td>
<td>143</td>
<td>45.2 KBytes</td>
</tr>
<tr>
<td>15</td>
<td>8.00-9.00 sec</td>
<td>119 MBytes</td>
<td>996 Mbits/sec</td>
<td>131</td>
<td>32.5 KBytes</td>
</tr>
<tr>
<td>15</td>
<td>9.00-10.00 sec</td>
<td>110 MBytes</td>
<td>923 Mbits/sec</td>
<td>182</td>
<td>46.7 KBytes</td>
</tr>
</tbody>
</table>
New bwctl features

• BWCTL now lets you run any of the following between any 2 perfSONAR nodes:
  – iperf3, nuttcp, ping, owping, traceroute, and tracepath

• Sample Commands:
  • bwctl -c psmsu02.aglt2.org -s elpa-pt1.es.net -T iperf3
  • bwping -s atla-pt1.es.net -c ga-pt1.es.net
  • bwping -E -c www.google.com
  • bwtraceroute -T tracepath -c lbl-pt1.es.net -l 8192 -s atla-pt1.es.net
  • bwping -T owamp -s atla-pt1.es.net -c ga-pt1.es.net -N 1000 -i .01
New website and user manual

- [http://www.perfsonar.net](http://www.perfsonar.net)
- [http://docs.perfsonar.net](http://docs.perfsonar.net)
New Plots!

Packet loss event

Route change to prior link - latency improves

Route change - latency is different

Corresponding low throughput

Throughput back to normal

Loss event?
New Plots!

Upgrade to v3.4 increased max TCP buffer size
How do I get the perfSONAR Toolkit?

• Available as a CentOS 6 ISO image containing OS plus all required perfSONAR packages
  – NetInstall most popular choice
  – New Full Install distribution in 3.4 release Note: LiveCD discontinued

• If you have installed Linux then you should be familiar with Toolkit process

• Individual packages available as RPMs for use on other CentOS-like systems (RHEL, Scientific)

• See http://www.perfsonar.net for download links and more details
For more information

• [www.perfsonar.net](http://www.perfsonar.net)
• [docs.perfsonar.net](http://docs.perfsonar.net)

• Note: FTW Workshop in January:
perfSONAR Toolkit 3.5: What’s Coming Next?

Luke Fowler
Indiana University
<luke@iu.edu>

2014 Technology Exchange
10-29-2014
Themes

• Improve Usability
• Increase Impact of perfSONAR
• Improve project Efficiency
• Support for Advanced Networking (e.g. SDN, Network Virtualization, Dynamic Networks, etc.)
Low Cost Nodes

• Lots of effort in the community to operate perfSONAR services on low cost ($100-$1,000) hardware

• High level of interest in community for low-cost deployment options (wiring closets, network engineer’s backpacks, “give-away” nodes, etc.)

• Want to provide a well supported and tested option for these users as part of pS-Toolkit

https://code.google.com/p/perfsonar-ps/wiki/lowCostPerfSONAR
User Interface Refresh

• The toolkit provides an easy to use web interface targeted at users at a variety of levels of technical expertise.

• We intend to refresh this UI in the 3.5 release to improve easy of use, visual appeal, and integration of state of the art UI components to extend UI lifespan.
Improving Efficiency and Consistency

• perfSONAR is a very active project, with a continuous development cycle.

• We want to spend less time building and testing and more time developing cool new features while maintaining/enhancing quality of each release!

• Enable this by focusing on:
  – Automated build and release management systems
  – Continuous integration of code, including automated unit and system testing.
Support Advanced Networking

• SDN, Network Virtualization, and Dynamic Networking technologies are being deployed across the global R&E substrate.

• perfSONAR needs to adapt to these new technologies

• Goal: make it just as easy to measure performance with perfSONAR in a virtual, dynamic, software defined network as it is on a “traditional” network.

• Assess landscape/develop roadmap for this during pS 3.5 development cycle.
perfSONAR Toolkit Security Assessment

• The perfSONAR toolkit is an integrated system, including open source software from third parties and software developed by the perfSONAR community.
• We’ve weathered several storms recently, most notably the “Shellshock” Bash bug.
• We have a proactive incident response strategy that works well.
• Changes to security posture in light of these vulnerabilities. Example: auto-updates enabled by default in v3.4.
• We think now is a good time to do a more formal assessment of the overall security posture of perfSONAR to identify areas of strength and areas for potential security improvements.
Also...

• Refactor BWCTL code to enhance maintainability and future development
• Modernize NDT functionality, including dependence on Web100 kernel and Java
• Improve support for network research by improving APIs and Documentation
• Auto-Config options to manage perfSONAR meshes
• Improve integration with NOC alarming systems
Sample iperf3 output on a clean network:

```
[tierney@pnwg-pt1 ~]$ bwctl -T iperf3 -w128M -c newy-pt1.es.net -s pnwg-pt1.es.net -0 5
bwctl: Using tool: iperf3
bwctl: 89 seconds until test results available

SENDER START
Connecting to host 198.124.238.54, port 5438
```

<table>
<thead>
<tr>
<th>ID</th>
<th>Interval</th>
<th>Transfer</th>
<th>Bandwidth</th>
<th>Retr</th>
<th>Cwnd</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>0.00-1.00</td>
<td>sec 166 MBytes</td>
<td>1.39 Gbits/sec</td>
<td>0</td>
<td>1.91 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>1.00-2.00</td>
<td>sec 52.5 MBytes</td>
<td>440 Mbits/sec</td>
<td>0</td>
<td>7.59 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>2.00-3.00</td>
<td>sec 140 MBytes</td>
<td>1.17 Gbits/sec</td>
<td>0</td>
<td>21.0 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>3.00-4.00</td>
<td>sec 691 MBytes</td>
<td>5.80 Gbits/sec</td>
<td>0</td>
<td>115 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>4.00-5.00</td>
<td>sec 1.12 GBytes</td>
<td>9.62 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>0.00-1.00</td>
<td>sec 1.13 GBytes</td>
<td>9.71 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>1.00-2.00</td>
<td>sec 1.11 GBytes</td>
<td>9.56 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>2.00-3.00</td>
<td>sec 1.12 GBytes</td>
<td>9.64 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>3.00-4.00</td>
<td>sec 1.13 GBytes</td>
<td>9.68 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>4.00-5.00</td>
<td>sec 1.13 GBytes</td>
<td>9.67 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>5.00-6.00</td>
<td>sec 1.12 GBytes</td>
<td>9.64 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>6.00-7.00</td>
<td>sec 1.13 GBytes</td>
<td>9.68 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>7.00-8.00</td>
<td>sec 1.12 GBytes</td>
<td>9.66 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>8.00-9.00</td>
<td>sec 1.12 GBytes</td>
<td>9.64 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>9.00-10.00</td>
<td>sec 1.13 GBytes</td>
<td>9.68 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>Interval</th>
<th>Transfer</th>
<th>Bandwidth</th>
<th>Retr</th>
<th>Cwnd</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>0.00-10.00</td>
<td>sec 11.2 GBytes</td>
<td>9.65 Gbits/sec</td>
<td>5</td>
<td>sender</td>
</tr>
<tr>
<td>17</td>
<td>0.00-10.00</td>
<td>sec 11.4 GBytes</td>
<td>9.78 Gbits/sec</td>
<td></td>
<td>receiver</td>
</tr>
</tbody>
</table>
Iperf3 “omit” option

```
[tierney@pnwg-pt1 ~]$ bwctl -T iperf3 -w128M -c newy-pt1.es.net -s pnwg-pt1.es.net
bwctl: Using tool: iperf3
bwctl: 36 seconds until test results available

SENDER START
Connecting to host 198.124.238.54, port 5436
[ 17] local 198.129.254.46 port 43901 connected to 198.124.238.54 port 5436

<table>
<thead>
<tr>
<th>ID</th>
<th>Interval</th>
<th>Transfer</th>
<th>Bandwidth</th>
<th>Retr</th>
<th>Cwnd</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>0.00-1.00</td>
<td>sec 166 MBytes</td>
<td>1.39 Gbits/sec</td>
<td>0</td>
<td>1.86 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>1.00-2.00</td>
<td>sec 53.8 MBytes</td>
<td>451 Mbits/sec</td>
<td>0</td>
<td>7.54 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>2.00-3.00</td>
<td>sec 138 MBytes</td>
<td>1.15 Gbits/sec</td>
<td>0</td>
<td>20.3 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>3.00-4.00</td>
<td>sec 674 MBytes</td>
<td>5.65 Gbits/sec</td>
<td>0</td>
<td>112 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>4.00-5.00</td>
<td>sec 1.12 GBytes</td>
<td>9.61 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>5.00-6.00</td>
<td>sec 1.13 GBytes</td>
<td>9.70 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>6.00-7.00</td>
<td>sec 1.11 GBytes</td>
<td>9.55 Gbits/sec</td>
<td>1</td>
<td>170 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>7.00-8.00</td>
<td>sec 1.13 GBytes</td>
<td>9.67 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>8.00-9.00</td>
<td>sec 1.12 GBytes</td>
<td>9.66 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
<tr>
<td>17</td>
<td>9.00-10.00</td>
<td>sec 1.12 GBytes</td>
<td>9.61 Gbits/sec</td>
<td>0</td>
<td>170 MBytes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>Interval</th>
<th>Transfer</th>
<th>Bandwidth</th>
<th>Retr</th>
<th>sender</th>
<th>receiver</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>0.00-10.00</td>
<td>sec 7.74 GBytes</td>
<td>6.65 Gbits/sec</td>
<td>1</td>
<td>sender</td>
<td>receiver</td>
</tr>
<tr>
<td>17</td>
<td>0.00-10.00</td>
<td>sec 7.74 GBytes</td>
<td>6.65 Gbits/sec</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```