# Firewall Port Recommendations for the pS Performance Toolkit

*Prepared by the NTAC Performance Working Group*
*November 2014*

*Edited by J. Zurawski (ESnet), A. Brown (Internet2), A. Lake (ESnet), K. Miller (The Pennsylvania State University), M. Swany (Indiana University), B. Tierney (ESnet), and M. Zekauskas (Internet2)*

## 1 Abstract

The ***perfSONAR Toolkit*** [pSPT] is a software package designed to facilitate *single shot* installation and configuration of network measurement tools and is designed to support deployments of all sizes. This infrastructure features a set of tools that gather and share metrics of interest about a target network path including achievable bandwidth, jitter, and latency to name a few. The task of gathering network measurements relies on methods that sample the medium, and requires stable and predictable end-to-end communication patterns to complete the operation.

Firewalls, and other in-line network traffic flow devices, will perturb the measurement process. Measurement tools are designed to be *fungible*, and often work on *ephemeral ports* (e.g. *communication channels*) to avoid the phenomena of traffic prioritization. A compromise in this space is the use of well-defined, yet still changeable, ports for communication purposes in each of the measurement tools. This permits the security infrastructure to know about the measurement work, and measurement to interface well with security tools. This also facilitates writing narrowly crafted traffic policy rules that permit measurements to occur without hindrance.

This document describes the recommended configuration of measurement ports for use with the ***perfSONAR Toolkit***. It is recommended that adopters of this product heed these recommendations in all cases, in order to reduce compatibility issues for those endpoints that may be using firewalls, to facilitate successful measurements between domains. The individual software packages on the ***perfSONAR Toolkit*** feature the same port recommendations as the new default behavior.

## 2 Performance Measurement

Research and Education (R&E) networks continue to increase capacity on a regular cycle. This growth is driven by an expanding user base and by ever-larger data set sizes. Network operators need monitoring tools that can report when the system fails to operate at its design capacity.

Traditional network monitoring is done within a domain. E.g. it is possible to passively observe the performance of network devices (e.g. *routers, switches, hosts*) through a wide range of tools meant to gauge uptime, service availability, and various metrics related to performance over time. While important in the overall ecosystem of operations – network use paradigms are shifting to emphasize *multi-domain* operation over strictly local traffic patterns.

Multi-domain monitoring involves a special set of concerns. It is necessary to evaluate the *end-to-end* performance across operational boundaries to simulate application performance. Active tools perform measurements of this nature; these tools test the network with applied traffic rather than recording traffic-related data generated by network routers and switches. Metrics such as *available bandwidth* help judge if a specific application will have a successful path to operate. If latency or data loss is a concern, there are tools available to address these concerns as well.

The myriad of choices available in the measurement and monitoring space can be daunting. Often the quantity of available tools can delay the deployment of a single solution; waiting for *the best* is the enemy of *the good*. To simplify the deployment of reliable and useful measurement tools, R&E partners have developed the ***perfSONAR Toolkit***. This framework of measurement tools, middleware, visualizations, and alarms, allows a compact deployment of monitoring functionality. Configuration of tools is reduced to a series of *wizard like* question and answer pairs, with the goal of learning about the local site and where tests to remote locations should be established.

Active measurement tools such as those found on the ***perfSONAR Toolkit*** typically have well defined communication patterns (e.g. *control protocols*) as well as data requirements to evaluate specific metrics. Since this is *real* network traffic, there is a risk that security devices will flag, or otherwise block, critical testing functionality. Intrusion detection tools may view sudden bursts of traffic from a measurement tool as a threat. A firewall must be aware of the traffic pattern and communication requirements to effectively prevent measurements from being blocked.

Modern network measurement tools are flexible to the needs of the environment. It is common for these tools to feature customized operational components. For example, a throughput tool such as **iperf** [iperf] is designed to use a default data port (i.e. *5001*) but can be configured via simple run time options to use alternate ports.

This flexibility has introduced an equally menacing problem in the operations space: with finite, yet large, amount of ports available for testing, the choice to use specific ranges becomes a site-by-site project. It may be the case that one site chooses a low number for their **iperf** server, while another chooses something very high. Without *a priori* knowledge of what a given site has chosen, automated measurement becomes a challenging exercise to determine conventions for all sites involved.

To combat the tyranny of choice, the authors are proposing a simple approach: recommend port ranges for tools in a community forum, change the default behaviors of measurement tools to use these ports, and encourage all sites (those that test without security infrastructure and those that do) to implement the recommendations. When adopted, these guidelines will facilitate measurement operation in a unified and predictable fashion.

# 3 pS Performance Toolkit

The R&E community has long had an interest in the development of tools to measure and monitor network performance. This interest stems from early realizations that nearly *unlimited bandwidth* and the presence management techniques such as QoS (Quality of Service) cannot guarantee seamless use by end users [BTG]. Many factors have been known to stand in the way of success:

- Infrastructure flaws
- Congestion
- End System
- Protocols
- Applications

R&E partners ESnet, Georgia Tech, Indiana University, Internet2, SLAC, and the University of Delaware joined forces in 2007 to combine many of the accepted solutions into a single component for modern networks utilizing the emerging ***perfSONAR*** [pS] standard as a starting point – the ***perfSONAR Toolkit***.

The individual components of the ***perfSONAR Toolkit*** are listed and explained below. This self-contained product consists of a tuned Linux operating system, performance tools, middleware to serve as glue between components, and an API that facilitates expansion. Deployment has been recognized at college campuses, research laboratories, regional networks, backbone networks, and exchange points, and continues to grow as the software matures.

The software featured in this system, whether measurement or middleware in nature, requires access to the network through specific communication ports. These ports should be uniformly selected between test end points to facilitate easy operation. In cases where there is no guidance to specific port bindings, a litany of choices can force compatibility issues between tools, which reduces the likelihood of adoption, and success in operation.

## 3.1 Core Services

There are several core *non-measurement* services that require exposure via the security infrastructure. These are well known systems used in other contexts, and are well supported by the open source development community.

- ***HTTP*** - The Apache web server is used to manage the ***perfSONAR Toolkit*** through a series of configuration interfaces, and to view the results of performance tests.
- ***NTP*** - The measurement tools require a stable time source. To facilitate this, an NTP daemon is enabled by default and configured to synchronize with well-known and stable clocks.
- ***DNS*** – The measurement tools must take advantage of the DNS system to resolve the mapping between addresses and host names.

- ***SSH*** – An additional management functionality in the form of remote shell access, is provided by an SSH daemon. This service is not enabled by default.

## 3.2 Measurement Tools

Measurement tools are the raw software components that measure networking infrastructure. Some of these tools are embedded and supported by operating system directly (e.g. *Ping*), others are products produced and supported by external parties.

- ***Primitive Tools*** - The Ping, Traceroute, and Tracepath tools rely on ICMP (and in some cases, TCP and UDP) packets sent between a source and a destination. A host may actively send, or be the target of reception, for these measurements.
- ***NPAD and NDT*** – Some measurements have a specific user target. NDT and NPAD are invoked via web pages, and then deliver their results to the user via a web browser. Both tools perform a variety of small tests on metrics including throughput and latency.
- ***OWAMP*** – Latency tools send streams of small packets and measure the results in terms of metrics such as loss, reordering, duplication, and packet delay variation. OWAMP performs these tests independently in both directions.
- ***BWCTL*** – Available bandwidth is a measure of how much a network resource is available at a given point in time. It combines a measure of 'throughput' against the ceiling of 'capacity'. BWCTL invokes several bandwidth measurement tools between a target and the local host.

## 3.3 Measurement Middleware

***perfSONAR*** services can be viewed as web services, similar to an actual web server such as Apache. They deliver structured data to a client that sends a decodable request.

- ***esmond*** – This is a database that stores variours forms of measurement data.
- ***Simple Lookup Service*** – This service catalogs and exposes measurement data on a local and global scale.
- ***OPPD*** – A regular testing service for one way delay.

## 3.4 Historic Measurement Middleware

Previous ***perfSONAR*** release featured a wider set of services that required opened ports for operation. Releases after version 3.4 do not contain these services, but the ports are provided for historical reasons, these are described in more detail in Section 5:

- ***Echo*** – Some of the tools perform a simple connect to ensure the other side is listening. Leaving this port open increases the chance of success for this test.
- ***SNMP Measurement Archive*** – This service interfaces with SNMP collection systems and exposes counter data via a web service interface.

- ***Traceroute Measurement Archive*** – This service actively makes traceroute measurements to dedicated sites as well as sharing the information through a web services interface.
- ***perfSONAR-BUOY Measurement Archive*** - This service actively makes OWAMP and BWCTL measurements to dedicated sites as well as sharing the information through a web services interface.
- ***PingER*** - This service actively makes Ping measurements to dedicated sites as well as sharing the information through a web services interface.
- ***Lookup Service*** – This service catalogs and exposes measurement data on a local and global scale.

# 4 Port to Tool Mapping

**Table 1 - Port to Tool Mapping**

| Service Type | Service Name | Transport Protocol | Local Ports | Direction |
|---|---|---|---|---|
| Core | Web Server (Apache, esmond) | TCP | 80<br>443 | Incoming |
| | Time Server (NTP) | UDP | 123 | Outgoing<br>Incoming* |
| | DNS | UDP | 53 | Outgoing |
| | Remote Access (SSH) | TCP | 22 | Incoming<br>Outgoing† |
| Measurement Tools | Primitive Tools (Ping, Traceroute, Tracepath) | ICMP | N/A‡ | Outgoing<br>Incoming§ |
| | | TCP,UDP** | 0 | |
| | Traceroute | UDP | 33434-33534 | Incoming |
| | NPAD Control | TCP | 8000 | Incoming |
| | NPAD Testing | | 8001-8020 | Incoming<br>Outgoing |
| | NDT Control | TCP | 7123 | Incoming |
| | NDT Testing | | 3001-3003 | Incoming<br>Outgoing |
| | OWAMP Control | TCP | 861 | Incoming<br>Outgoing |
| | OWAMP Testing | UDP | 8760-9960†† | |
| | BWCTL Control | TCP | 4823 | Incoming<br>Outgoing |
| | BWCTL Peer | TCP, UDP | 6001-6200‡‡ | |
| | BWCTL Testing | TCP, UDP | 5000-5900§§ | |
| Measurement Middleware | Lookup Service | TCP | 8090<br>8096 | Incoming<br>Outgoing |
| | OPPD | TCP | 8090 | Incoming<br>Outgoing |
| Directory Server | Lookup Service | TCP | 61617 | Incoming<br>Outgoing |

* If queries are allowed against time server; this is strictly optional
† If local users choose to SSH to other machines once logged in
‡ Note that ICMP has no notion of *ports* as TCP and UDP do
§ It is highly recommended that incoming packets be allowed; outgoing is more critical for local measurement
** These are less common options, but still available via the tools
†† See Section 6.1 for details on how to configure these ports.
‡‡ See Section 6.2 for details on how to configure these ports.
§§ See Section 6.2 for details on how to configure these ports.

# 5 Historical Port to Tool Mapping

**Table 2 – Historical Port to Tool Mapping**

| Service Type | Service Name | Transport Protocol | Local Ports | Direction |
|---|---|---|---|---|
| ***Historic (releases prior to perfSONAR 3.4)*** | Echo | TCP, UDP | 7 | Incoming<br>Outgoing |
| | SNMP MA | TCP | 8065<br>9990 | Incoming |
| | Traceroute MA | TCP | 8086 | Incoming<br>Outgoing |
| | Traceroute MA Measurement | ICMP<br>TCP<br>UDP | 8087 | |
| | PingER MA | TCP | 8075 | Incoming |
| | PingER Measurement | ICMP<br>TCP<br>UDP | | Outgoing |
| | perfSONAR-BUOY MA | TCP | 8085 | Incoming |
| | perfSONAR-BUOY Control | | 8569<br>8570 | Incoming<br>Outgoing |
| | Lookup Service | TCP | 8095<br>9995 | Incoming<br>Outgoing |

# 6 Configuration Options

The tools mentioned in Table 1 and Table 2 are configured by default to use the ports mentioned with two historic exceptions: older releases of OWAMP and BWCTL tools do not have specific recommendations for testing ports. The developers of these tools made the proper modifications to software and documentation to ease the transition to a recommended port range; older versions may still require a manual configuration. The next sections will outline the changes to implement the recommended ports.

## 6.1 OWAMPD

The OWAMPD tool has a single configuration file that will need to be edited:

1. Edit **`/etc/owampd.conf`** , **`/etc/owampd/owampd.conf`** or **`/etc/owamp/owampd.conf`** (N.B. this location will change depending on the age of your installation)
2. Set the **`testports`** variable to the agreed range of values (8760-9960)
3. Save the file
4. Restart the service with **`sudo /etc/init.d/owampd restart`**

## 6.2 BWCTLD

The BWCTLD tool has a single configuration file that will need to be edited:

1. Edit **`/etc/bwctld.conf`** , **`/etc/bwctld/bwctld.conf`** or **`/etc/bwctl/bwctld.conf`** (N.B. this location will change depending on the age of your installation)
2. Releases after and including 1.5.2 of BWCTL:
    a. Remove references to **`iperf_port`**, **`iperf2_port`**, **`nuttcp_port`**, **`thurlay_port`**, or **`owamp_port`**.
    b. Set the **`test_port`** variable to the agreed range of values (5001-5900)
3. Release prior to 1.5.2 of BWCTL:
    a. Set the **`iperf_port`** variable to the agreed range of values (5001-5200)
    b. Set the **`thrulay_port`** variable to the agreed range of values (5201-5400)
    c. Set the **`nuttcp_port`** variable to the agreed range of values (5401-5600)
    d. Set the **`iperf3_port`** variable to the agreed range of values (5601-5800)
    e. Set the **`owamp_port`** variable to the agreed range of values (5801-5900)
4. Set the **`peer_port`** variable to the agreed range of values (6001-6200)
5. Save the file
6. Restart the service with **`sudo /etc/init.d/bwctld restart`**

## 6.3 POWSTREAM & OWPING

The POWSTREAM and OWPING tools do not automatically search for port values in the **owampd.conf** file to use for outbound traffic. As such, operation of these tools requires using a runtime option to invoke the proper port choices. This will be rectified in a future release. For now the following steps should be taken:

- Invoke **owping** or **powstream** with **–P 8760-9960**

## 6.4 BWCTL

If there is not a local copy of BWCTLD running on a host, invoking BWCTL will spawn a new daemon. This daemon may not know of the proper ports to use *a priori*. There is a configuration file (bwctlrc) that can be created in a user's home directory that knows the proper port options to invoke. The following steps should be followed:

1. Edit **~/bwctlrc**
2. Releases after and including 1.5.2 of BWCTL:
    a. Remove references to **iperf_port, iperf2_port, nuttcp_port, thurlay_port,** or **owamp_port.**
    b. Set the **test_port** variable to the agreed range of values (5001-5900)
3. Releases prior to 1.5.2 of BWCTL:
    a. Set the **iperf_port** variable to the agreed range of values (5001-5200)
    b. Set the **thrulay_port** variable to the agreed range of values (5201-5400)
    c. Set the **nuttcp_port** variable to the agreed range of values (5401-5600)
    d. Set the **iperf3_port** variable to the agreed range of values (5601-5800)
    e. Set the **owamp_port** variable to the agreed range of values (5801-5900)
4. Set the **peer_port** variable to the agreed range of values (6001-6200)
5. Save the file
6. Invoke the **bwctl** application as normal.

# 7 Conclusion

Measurement and monitoring remain a crucial part of modern network infrastructure, as do methods to secure and protect critical components. Growth in capacity, number of users, and traffic end points, amplifies the critical nature of software frameworks that deliver on performance promises. Security and measurement can work within well-defined boundaries to ensure mutual success. One of the first steps in this direction is defining a well-known set of ports for use in measurement activities that can be white listed by security devices. This will allow success for measurement tools and facilitate end-to-end testing across domains for all users: those that use such security, and those that don't.

The authors believe that the recommendations set forth in this work, while brief and simplistic in nature, can clear up numerous problems as reported by the R&E networking community:

- Tool failure due to control channel failure
- Measurement abnormalities due to data loss
- Configuration hardships based on a lack of documentation or ways to specify tool behavior

This document serves as the basis for changes implemented within the ***perfSONAR Toolkit*** software to better describe and enforce measurement tool deployment.

# 8 Acknowledgements

The authors would like to thank the contributions of the NTAC Performance Working group [PerfWG] in the creation of this document. The insight and experience of these dedicated individuals continues to advance the community toward well functioning and predictable networks.

The authors would also like to thank the contributions of the ***perfSONAR*** development community. This dedicated team makes it possible to complain about trivial matters such as port assignments, versus the alternative of still struggling to identify the performance bottlenecks on networks via less sophisticated means.

# 9 References


[pSPT] The perfSONAR Toolkit
http://www.perfsonar.net/deploy/

[iperf] IPERF
http://sourceforge.net/projects/iperf/

[pS] perfSONAR
http://www.perfsonar.net

[PerfWG] NTAC Performance Working Group
https://spaces.internet2.edu/display/PerformanceWG

[BTG] Bridging the Gap
http://e2epi.internet2.edu/btg/