# One network metric datastore to track them all: the OSG network metric service

IOP Conf. Series: Journal of Physics: Conf. Series **898** (2017) 082044 doi:10.1088/1742-6596/898/8/082044

# One network metric datastore to track them all: the OSG network metric service

**Robert Quick[1], Marian Babik[2], Edgar M Fajardo[3], Kyle Gross[1], Soichi Hayashi[1], Marina Krenz[1], Thomas Lee[1], Shawn McKee[4], Christopher Pipes[1], Scott Teige[1]**

[1] Indiana University, Bloomington, IN, USA
[2] European Organization for Nuclear Research (CERN), Switzerland
[3] University of California San Diego, La Jolla, CA, USA
[4] Physics Department, University of Michigan, Ann Arbor, MI 48109-1040 USA

E-mail: rquick@iu.edu

**Abstract.** The Open Science Grid (OSG) relies upon the network as a critical part of the distributed infrastructures it enables. In 2012, OSG added a new focus area in networking with a goal of becoming the primary source of network information for its members and collaborators. This includes gathering, organizing, and providing network metrics to guarantee effective network usage and prompt detection and resolution of any network issues, including connection failures, congestion, and traffic routing.

In September of 2015, this service was deployed into the OSG production environment. We will report on the creation, implementation, testing, and deployment of the OSG Networking Service. Starting from organizing the deployment of perfSONAR toolkits within OSG and its partners, to the challenges of orchestrating regular testing between sites, to reliably gathering the resulting network metrics and making them available for users, virtual organizations, and higher level services, all aspects of implementation will be reviewed. In particular, several higher-level services were developed to bring the OSG network service to its full potential. These include a web-based mesh configuration system, which allows central scheduling and management of all the network tests performed by the instances; a set of probes to continually gather metrics from the remote instances and publish it to different sources; a central network datastore (esmond), which provides interfaces to access the network monitoring information in close to real time and historically (up to a year) giving the state of the tests; and a perfSONAR infrastructure monitor system, ensuring the current perfSONAR instances are correctly configured and operating as intended.

We will also describe the challenges we encountered in ongoing operations of the network service and how we have evolved our procedures to address those challenges. Finally we will describe our plans for future extensions and improvements to the service.

## 1. Introduction

### *1.1. Brief Description of the Worldwide LHC Computing Grid and Open Science Grid*

The Worldwide LHC Computing Grid (WLCG)[1] and the Open Science Grid (OSG)[2], as distributed E-Infrastructures, are heavily dependent on the existing research network. In the scope of this publication, WLCG refers to the ALICE, ATLAS, CMS, and LHCb computing resource providers. This includes Tier 1, Tier 2, and Tier 3 facilities and users of these resources. OSG refers to the resources that provide computing cycles to the Open Science Grid and the

researchers utilizing these resources. While there is some overlap in these communities they should be considered different environments with different, though similar, needs.

Both E-infrastructures have a critical dependency on the underlying network. Hence, finding network issues early and acting upon them make the operations of these E-infrastructures more reliable and productive for end-users.

To this end, the Open Science Grid is hosting a persistent datastore which collects these network metrics and makes them available for collaborators to gather and create visualizations based on their project needs.

### *1.2. Motivation for Network Monitoring*

As the WLCG and OSG function by sending workflows and data to hundreds of resources around the world, the criticality of the network for producing science results is obvious. Having reliable network connections from each resource to all other resources within this environment are not only critical to science workflows, but are also a requirement for data analysis and thereforeThe OSG Network Datastore includes bandwidth and latency tests triggered locally, metric collection, metric storage, presentation and publication. These peices are all in place and fully functional. scientific discovery. The basic unit of work for both environments is a job; these jobs require transfer of application software and datasets along with the return of outputs to the submission source. Identifying and correcting network issues leads to increased productivity and analysis speed.

To accomplish this the OSG, in partnership with WLCG and LHCONE[3], made the decision to host a long term network metric datastore. This datastore would be used to investigate not only immediate network issues but historical trends over significant periods of time that might contribute to network interruptions and systemic degradation.

### *1.3. Technologies Leveraged*

A series of proven technologies are leveraged to tie together end-to-end network metric data collection for operational analysis. These technologies will be introduced at length in the Methods and Leveraged Software section below. They include the perfSONAR[4] network monitoring client, the Resource Service Validation (RSV) monitoring service, ESnet Monitoring Daemon (esmond)[5] tools, the Apache Cassandra[6] non-relational database, visualization interfaces Open Monitoring Distribution (OMD) and MaDDash, and the ActiveMQ[7] messaging service.

OSG and ESnet also developed a Graphical User Interface (GUI) Mesh Configuration tool that allows users to define network meshes composed of selected hosts which they are interested in (e.g. Tier 1s, USCMS, OSG, etc.) and control test scheduling and test parameters within those defined meshes.

### *1.4. Core Principles*

The OSG network datastore project was conceived on the principles of Open Science, Open Data, and community resource sharing.

Minimal software development was conducted by OSG and when possible, existing collaborator and open source software was integrated. The only major component that was directly developed by OSG is the Mesh Configuration GUI described below.

All data collected is made available for consumption and analysis publicly or, for archived data, by request for legitimate research related uses. OSG welcomes and encourages researchers to collect, analyze, and visualize this data in a way that is useful to their operation.

## 2. Methods and Leveraged Software

### *2.1. perfSONAR*

"perfSONAR is a network measurement toolkit designed to provide federated coverage of paths, and help to establish end-to-end usage expectations. There are thousands of perfSONAR instances deployed world wide, many of which are available for open testing of key measures of network performance. This global infrastructure helps to identify and isolate problems as they happen, making the role of supporting network users easier for engineering teams, and increasing productivity when utilizing network resources."[8]

OSG has packaged a pre-configured version of perfSONAR that can be easily installed for WLCG and OSG resources.

### *2.2. esmond*

esmond is described on the project's website as, "a system for collecting, storing, visualizing and analyzing large sets of timeseries data. It was driven by the needs of the ESnet engineering team but is likely useful to a much wider audience. esmond has a RESTful API which allows easy access to the data which is collected. The original focus was on collecting SNMP timeseries data which is still the systems forte, but there is support for generalized timeseries data.

esmond uses a hybrid model for storing data, relying on Cassandra and PostgreSQL for data management. Timeseries data such as interface counters is stored using Cassandra. esmond will save the raw data, and create summarizations similar to RRD. However, the system never discards data through summarization, which distinguishes it from RRD (and whisper/ceres). Metadata (such as interface description and interface types from SNMP) are stored in an PostgreSQL database. Storing this data in an SQL database allows us to use the full expressiveness of SQL to query this data."[5]

### *2.3. Mesh Configuration*

Multiple perfSONAR toolkit instances can be controlled through the use of Mesh Configuration. A "mesh" represents a set of hosts, their relationship to one another, a test type and test parameters and is typically associated with a particular science domain and/or geographical grouping of perfSONAR instances. A perfSONAR toolkit instance can be configured to use one or more mesh-configuration URLs to get configuration information from. This has some powerful benefits we will discuss below.

When first developed, mesh-configurations were built using a text file and a small program that validated the text-configuration and produced JSON output that could be served via a web-server. This was somewhat tedious to manage as the number of meshes increased and required someone to make manual updates each time hosts or tests were added, deleted or changed. In addition, if a new mesh was created, each perfSONAR instance the mesh referred to would need to update its mesh-configuration to add the URL for the new mesh.

To address the short-comings of the original text-file based configuration, the Open Science Grid implemented a Mesh Configuration GUI. This tool leveraged OSG's existing resource registration database called "OIM"[9]; the OSG Information System, which contains a list of all perfSONAR toolkit instances that are members of OSG and WLCG communities. Implementing this tool within OIM minimized the amount of development effort required to build this tool but did make it difficult to install outside of OSG. In addition to providing access to the registration information for all perfSONAR toolkit instances registered in OIM, it gathers registration details from GOCDB[10] for WLCG perfSONAR instances. This allows mesh administrators to easily build new meshes and reduces typographical errors by auto-completing host information as entries are typed. Another OSG application, MyOSG[11], was also employed as a mechanism to publish the actual Mesh Configurations to be consumed by various toolkit instances in a reliable and scalable manner.

To see the power of the mesh-configuration concept, consider the work typically required to setup a 10 host mesh manually. Each of 10 perfSONAR administrators would need to separately configure tests to each of the other 9 host involved. Ensuring the tests are consistently defined is not easy. What happens when a host leaves the mesh? The other 9 administrators must reconfigure to remove that host. If a host joins, all other mesh admins must reconfigure to add the new host. Similarly any change in testing parameters or scheduling needs to be coordinated with all the perfSONAR admins. Note that for OSG/WLCG we have over 250 perfSONAR toolkit instances and some meshes include up to 80 hosts. Without the mesh-configuration we would be unable to manage a deployment at this scale.

The mesh-configuration capability allows for a single location to define all the participating hosts, tests and schedules, ensuring consistency and allowing quick updates to configurations. There are some additional benefits from having the registration databases as the basis of the mesh-config GUI: any change in registration can be quickly, correctly and automatically propagated to any mesh which includes that host.

There is another benefit from centralizing the configuration: the OSG mesh-configuration knows about **all** the hosts and meshes that are in use. In the original text-based system, if a new mesh was added, the participating hosts would need to update their configuration to use this new mesh configuration. With the central GUI we can instead create a single "auto-mesh" URL which uses the Fully Qualified Domain Name (FQDN) of the perfSONAR instance. The perfSONAR toolkit admin just needs to configure a single URL and any changes, additions or deletions of meshes will result in the correct, host-specific configuration being provided

Recently, there has been interest from communities outside the OSG to deploy a similar Mesh Configuration administration tool. However, packaging and distribution of this software as a general standalone tool for the perfSONAR community is difficult due to its dependencies to various OSG systems. A new, improved stand-alone version, called MeshConfig Administrator (MCA)[12], is currently being developed to allow anyone to install their own version without requiring MyOSG and OIM. It is targeted for release as part of the perfSONAR Toolkit v4.0 version.

### *2.4. Resource Service Validation Probes (RSV)*

At the heart of the network data collection obtained from the worldwide distribution of perfSONAR servers (See section 2.1) lay RSV network probes. Current RSV[13] software was leveraged from its original use case. Although RSV has been extensively used throughout OSG grid sites to monitor health of their own resources[14], computing gateways, storage and availability; its architecture of Perl and Python independent and idempotent scripts, that run at given intervals via a cronjob mechanism based on HTCondor[15], was flexible enough to be adapted for this task.

Two kinds of Python RSV probes were developed. The first probe gathers the host names of the perfSONAR servers defined in the different meshes described in 2.3. Afterwards it instructs RSV to activate a second type of probe for every host in the mesh configuration, and deactivates the host names that no longer appear in any of the meshes.

Second type of probes are the central part of the system, where one probe is running for each perfSONAR host. Each probe queries a perfSONAR host over an esmond API and receives all new measurements from a certain point in time ensuring independence from each probe. Esmond API works over http/https protocol, and is also utilized to trasfer data to the Central OSG Cassandra database. Once, the probe uploads the information to the Central OSG Cassandra database it stores the timestamps of the latest successfully posted data to guarantee the probe is idempotent. Finally, due to the agreement between WLCG and OSG, those probes also post some of the gathered data to a messaging queue (see Section 2.6), which is designated

for later consumption rather than long term storage.

### *2.5. OMD and MadDash*

OSG has a responsibility to its constituents and partners to ensure that network metrics are being consistently and correctly gathered. Given the large number of perfSONAR toolkits, measuring network metrics for OSG and WLCG, it is not trivial to ensure they are all operating appropriately.

OSG has developed a two-fold approach to verifying that network data metrics are current and that the underlying services support making the network measurements and providing those measurements when queried continue to function.

A sample of the OMD display is presented in Figure 1.

**Figure 1.** Open Monitoring Distribution Nagios[16] based monitoring tools

### *2.6. ActiveMQ*

In addition to the standard esmond programmatic interface (API), the network datastore supports publishing of the metrics via messaging system. This makes it possible for end-users to subscribe to events as they're collected and process them in near real-time without performing iterated requests to the datastore. We're relying on the Apache ActiveMQ to broker the messages between the collector (publisher) and the end-users (subscribers) using simple text-oriented message protocol (STOMP). A high-availability cluster has been setup for this purpose at CERN, which provides both testing and production brokers. The actual mechanism for publishing from the collector uses an intermediate directory to store the messages as files that are later processed by an open source tool called stompclt, which handles connection to the brokers, sending and removing the messages from the file system.

As the data rates of the collected measurements can be quite significant, the data is structured into set of topics, which can be subscribed to on an individual basis. The topics are organized based on the esmond event types (data types) and are provided in two distinct sets, i.e. raw and summary. Some examples of the topics published by the collector are:

- *perfsonar.raw.histogram-owdelay* - Histograms describing the observed one-way delays between source and destination over a time period (typically 60 seconds). Source and destination as well as other metadata would be included in the header of the message.

- *perfsonar.summary.histogram-owdelay* - Summaries computed by esmond for the one-way delay histograms. Typically this would include histograms aggregated per 5 minutes, 1 hour and 1 day as well as common statistical measures for same periods (min, max, avg, mean and percentiles).
- *perfsonar.raw.throughput* - Measured amount of data sent over a period of time from source to destination (typically using iperf3 in bps). The raw data would be a simple set of tuples with timestamp and throughput values.

For a full list of supported event types, which are generally accessible and published under topics described above, please refer to the esmond API documentation at `http://software.es.net/esmond/perfsonar_client_rest.html#full-list-of-event-types`.

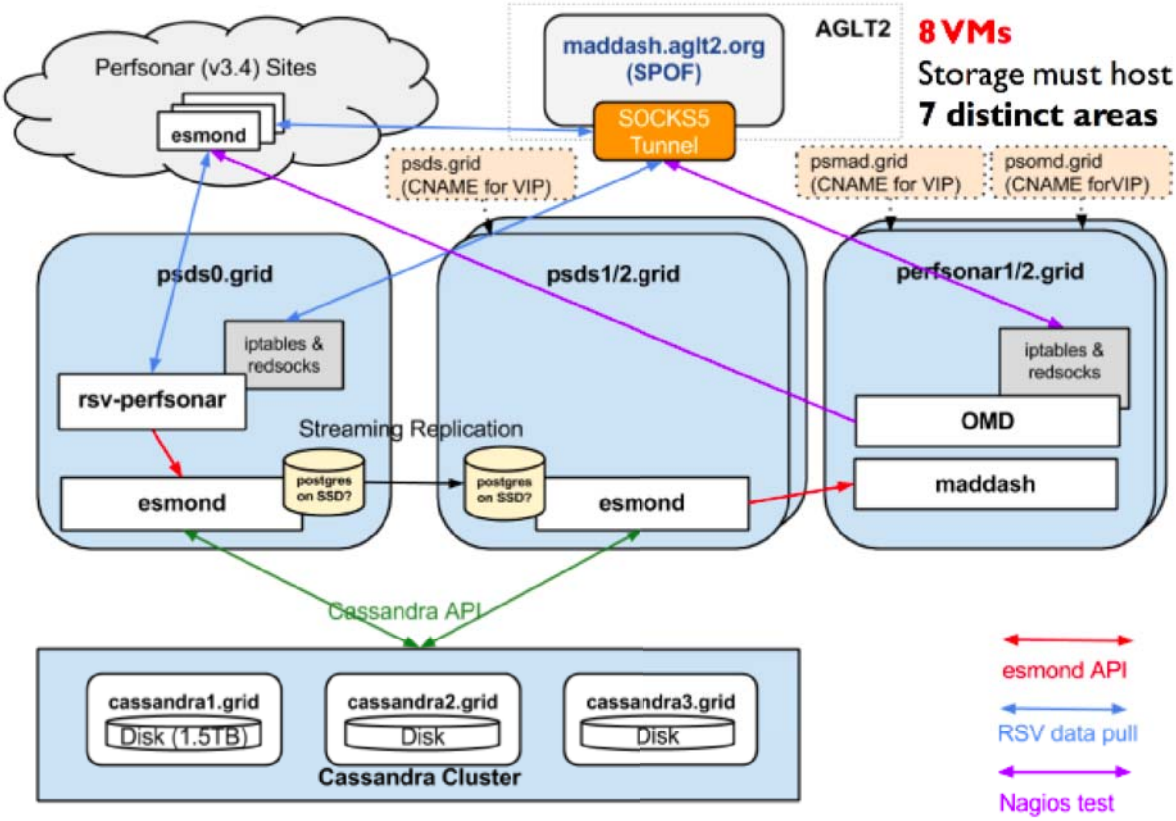
### *2.7. Operational Services Leveraged*

In addition to the previously described RSV[13] and MyOSG[11] services this system uses the already existing OSG ticketing system[17] and the OSG information manager OIM[9]. Ticketing is a well known means for reporting, tracking and documenting the means to resolve operational problems. OIM is the OSG implementation of a mysql database server with a web-UI and is used in this context to provide access to slowly changing topology data.

### *2.8. End to End Overview*

The OSG Network Datastore includes bandwidth and latency tests triggered locally, metric collection, metric storage, presentation and publication. These pieces are all in place and fully functional. A logical diagram of the components is shown in Figure 2.

**Figure 2.** OSG Network Datastore Logical Diagram



## 3. Service Operation

### *3.1. Availability Monitoring*

Check_MK[18] is used as the mechanism to determine the state of the system. Check_MK can be configured to perform many tests evaluating the status of the underlying services, operating system, and hosting hardware. Some tests implemented in this instance include CPU load, context switching rate, total number of processes, used memory, disk use and access rate, and network interface status. In addition to these system level diagnostics, specific applications required by the service are required to be running. If any of these tests are in a state outside specified ranges defining a warning or critical state an appropriate status is asserted by Check_MK.

## 4. Discussion

### *4.1. Problems and Challenges*

Gathering network metrics is just the first step to solving operational issues that affect the OSG and WLCG. The end goal is to provide information, visualization, and tools that allow OSG and WLCG Network Operations to identify, gather important diagnostic data, and involve network engineers in fixing network issues. This leading to improved workflow efficiency and possible smart job routing.

At this point the initial data and visualization components are in place. Satellite projects[19][20] are working on tools for alerting and alarming. OSG and WLCG Operations is learning how to learn and act on the data produced by the reported network metrics.

One main challenge will be to prove the local value of sending perfSONAR metrics to the datastore and thus build the user base. While the WLCG has wide adoption, OSG resources not related to WLCG are still not well represented. A push for this adoption by a wider OSG resource pool is underway.

The other main challenge is related to compiling and acting on the information that is collected. Policy and procedure need to be developed to allow end-to-end troubleshooting. In this case, end-to-end troubleshooting refers to the detection, alarming, information gathering, issue tracking, and finally solution to network problems affecting the WLCG and/or the OSG.

A final challenge lies in the summarization and archival of data for long term trend analysis. Today, no mechanism exists that would allow the OSG to summarize useful data and discard individual records. We will soon add disk capacity to keep records current live, but need to find a long-term solution to this issue.

### *4.2. Next Steps*

OSG efforts have enabled a global network monitoring infrastructure capable of "seeing" the state of the networks that interconnect distributed science centers worldwide. We are able to visualize the results of measurements of tens of thousands of network paths daily and understand the long-term behavior of the networks we monitor. Archiving and long-term storage mechanisms will need to be found, either by a change of technology, adopting an ESNet technology, or creating a custom product for summarization, compression, and archiving. The next step is to use this data to alarm and alert when deficiencies in our network infrastructure are found. In the distributed science collaboration that is the WLCG and OSG end-users, site administrators and network engineers face difficulties identifying network problems involving their domains. Our goal is to clarify, localize, and report, in a timely way, network problems wherever they occur in our distributed infrastructure.

## Acknowledgments

The authors would like to acknowledge the contributions of the perfSONAR project, namely staff from ESnet, GEANT, Indiana University, Internet2, the University of Michigan, and all

the contributors shown at http://www.perfSONAR.net/about/who-is-involved/.

We gratefully acknowledge the support of the National Science Foundation (grant 1148698) which supported the work in this paper.

## References

[1] Bird I 2011 Computing for the large hadron collider vol 61 (Annual Reviews) pp 99–118 URL http://www.annualreviews.org/doi/abs/10.1146/annurev-nucl-102010-130059
[2] Altunay M, Avery P, Blackburn K, Bockelman B, Ernst M, Fraser D, Quick R, Gardner R, Goasguen S, Levshina T, Livny M, McGee J, Olson D, Pordes R, Potekhin M, Rana A, Roy A, Sehgal C, Sfiligoi I and Würthwein F 2011 A science driven production cyberinfrastructure – the open science grid vol 9 (Springer Netherlands) pp 201–218 ISSN 1570-7873 URL http://dx.doi.org/10.1007/s10723-010-9176-6
[3] Martelli E and Stancu S 2015 *Journal of Physics: Conference Series* **664** 052025 URL http://stacks.iop.org/1742-6596/664/i=5/a=052025
[4] Hanemann A, Boote J, Boyd E, Durand J, Kudarimoti L, Lapacz R, Swany D M, Trocha S and Zurawski J 2005 Perfsonar: A service oriented architecture for multi-domain network monitoring vol 3826 (Springer Berlin Heidelberg) pp 241–254 URL http://link.springer.com/chapter/10.1007%2F11596141_19
[5] Esnet monitoring daemon URL http://software.es.net/esmond/
[6] Apache cassandra URL https://en.wikipedia.org/wiki/Apache_Cassandra
[7] Apache activemq URL http://activemq.apache.org/
[8] Description of perfsonar and its purpose URL http://www.perfsonar.net/about/what-is-perfsonar/
[9] Osg information management system URL https://oim.grid.iu.edu/oim/home
[10] Mathieu G, Richards D A, Gordon D J, Novales C D C, Colclough P and Viljoen M 2010 Gocdb, a topology repository for a worldwide grid infrastructure vol 219 p 062021 URL http://stacks.iop.org/1742-6596/219/i=6/a=062021
[11] Gopu A, Hayashi S and Quick R 2009 Myosg: A user-centric information resource for osg infrastructure data sources *Proceedings of the 5th Grid Computing Environments Workshop* GCE '09 (New York, NY, USA: ACM) pp 12:1–12:10 ISBN 978-1-60558-887-2 URL http://doi.acm.org/10.1145/1658260.1658276
[12] Meshconfig administrator URL http://docs.perfsonar.net/mca.html
[13] Emmanuel Udoh e 2011 Cloud, grid and high performance computing: Emerging applications: Emerging technologies
[14] Quick R, Gopu A and Hayashi S 2009 Rsv: Osg fabric monitoring and interoperation with wlcg monitoring systems URL https://indico.cern.ch/event/35523/contributions/840119/
[15] Thain D, Tannenbaum T and Livny M 2005 *Concurrency - Practice and Experience* **17** 323–356
[16] 2010 Nagios core 3.x documentation URL https://www.nagios.org/
[17] Hayashi S, Gopu A and Quick R 2011 Goc-tx: A reliable ticket synchronization application for the open science grid vol 331 p 082013 URL http://stacks.iop.org/1742-6596/331/i=8/a=082013
[18] Check_mk an extension to the nagios monitoring system URL https://en.wikipedia.org/wiki/Check_MK
[19] Batista J, Dovrolis C, Lee D and McKee S 2015 Identifying and localizing network problems using the pundit project vol 664 p 052027 URL http://stacks.iop.org/1742-6596/664/i=5/a=052027
[20] Vukotic I and Gardner R 2017 Big data analytics tools as applied to atlas event data vol 22nd International Conference on Computing in High Energy and Nuclear Physics (CHEP2016)