Identifying and localizing network problems using the PuNDIT project

Jorge Batista[1], Constantine Dovrolis[2], Danny Lee[2] and Shawn McKee[1]

[1] Randall Laboratory, Physics Department, University of Michigan, 450 Church Street, Ann Arbor, Michigan, 48109-1040, USA

[2] Klaus Advanced Computing Building, College of Computing, Georgia Institute of Technology, 266 Ferst Drive, Atlanta, Georgia, 30332-0765, USA

E-mail: `dovrolis@cc.gatech.edu, smckee@umich.edu`

Abstract. In today's world of distributed collaborations of scientists, there are many challenges to providing effective infrastructures to couple these groups of scientists with their shared computing and storage resources. The Pythia Network Diagnostic InfrasTructure (PuNDIT[1]) project is integrating and scaling research tools and creating robust code suitable for operational needs addressing the difficult challenge of automating the detection and location of network problems.

PuNDIT is building upon the de-facto standard perfSONAR[2] network measurement infrastructure deployed in Open Science Grid(OSG)[3] and the Worldwide LHC Computing Grid(WLCG)[4]to gather and analyze complex real-world network topologies coupled with their corresponding network metrics to identify possible signatures of network problems from a set of symptoms. The PuNDIT Team is working closely with the perfSONAR developers from ESnet and Internet2 to integrate PuNDIT components as part of the perfSONAR Toolkit. A primary goal for PuNDIT is to convert complex network metrics into easily understood diagnoses in an automated way.

We will report on the project progress to-date in working with the OSG and WLCG communities, describe the current implementation including some initial results and discuss future plans and the project timeline.

1. Introduction

Global research networks supporting data intensive science are rapidly moving towards terabit-per-second technologies; however, increases in bandwidth and network capacity may not necessarily translate to improved performance. Complexity is introduced when the end-to-end paths cross several autonomous systems, relying on diverse link and network technologies (e.g., hybrid packet and circuit switching), and the use of demanding scientific applications (e.g., remote visualization or distributed analysis of massive datasets). Performance problems may not manifest themselves as outright failures, but sporadic issues that go undetected, reducing the maximum possible performance. When performance problems are noticed by scientists they face significant challenges in finding the root cause and location in a multi-domain wide-area network.

To help address those challenges, the Open Science Grid(OSG)[3] and the World-wide LHC Computing Grid(WLCG)[4] have adopted a network monitoring framework based on perfSONAR [5] as a means to measure the networks that interconnect their broadly distributed resources and users. To do this they deployed an end-to-end monitoring infrastructure comprised of a number of commodity nodes co-located with data and compute facilities at their end-sites. These nodes perform active measurements amongst each other. A typical end-site monitoring deployment using perfSONAR consists of two commodity nodes that perform a full mesh of one-way delay, loss, reordering and throughput measurements between each other; perfSONAR uses active measurement tools such as OWAMP and BWCTL to do these measurements. In addition, perfSONAR nodes periodically record the IP-layer network topology using the traceroute or tracepath tool.

We note that network operators typically also deploy passive monitoring infrastructure such as SNMP traps, Netflow and syslogs from network devices. While active and passive methods are useful, they each have limitations. Passive monitoring gives detailed performance data, which can be used perform detailed diagnosis and localization (e.g., AT&T's G-RCA [6], NICE [7] and Giza [8]; SyslogDigest [9] and Sherlock [10]). Active monitoring infrastructure, on the other hand, scales to large networks and works when paths in the ISP traverse other administrative domains such as transit networks.

While OSG and WLCG have implemented a perfSONAR infrastructure and have very capable network operators running the research and education networks they use to interconnect their sites, they still face significant challenges in identifying real problems in the wide-area network, localizing those problems and getting them resolved in a timely way. That is the primary motivation for our project, PuNDIT[1], which is building upon the de-facto standard perfSONAR[2] network measurement infrastructure deployed in OSG and WLCG to gather and analyze complex real-world network topologies coupled with their corresponding network metrics to identify possible signatures of network problems from a set of symptoms. A primary goal for PuNDIT is to convert complex network metrics into easily understood diagnoses in an automated way.

PuNDIT is based upon Pythia[11], an earlier project that developed a distributed network monitoring system incorporating novel techniques to identify and explain problems in inter-domain paths, attributing probable causes to the router interfaces responsible. We intend that PuNDIT will augment Pythia with additional tools and data analysis components that help identify problems in the network and contribute to our ability to locate them. PuNDIT is an OSG satellite project and has collaboration agreements with the perfSONAR development team to work on integrating PuNDIT into future versions of the perfSONAR toolkit, as well as with OSG and WLCG to use their sites and infrastructure to help develop and deploy PuNDIT at-scale.

In this paper, we describe the PuNDIT project, its current status and how it is working with the OSG and WLCG communities to develop a tool to find, localize and eventually alert on network problems in the wide area research and eduction networks. This paper is organized as follows. We cover the system architecture in Section 2. We then cover the current work in a number of specific areas in Section 3 followed by a summary of our future plans in Section 4.

2. System Architecture

The PuNDIT architecture will build upon Pythia as its core component, updating Pythia to work with the current perfSONAR v3.4 toolkit components. PuNDIT will integrate an agent, based upon Pythia code which analyzes OWAMP raw packets, and some additional needed tools into the perfSONAR toolkit as an optional component sites can choose to enable. Each PuNDIT enabled perfSONAR toolkit instance will analyze data from perfSONAR measurements, trying to identify signatures of network problems in near real-time. If problems are found, the problem-signature and current network path information is reported to a central PuNDIT service whose task is to gather and correlate data from PuNDIT agents to identify and localize network

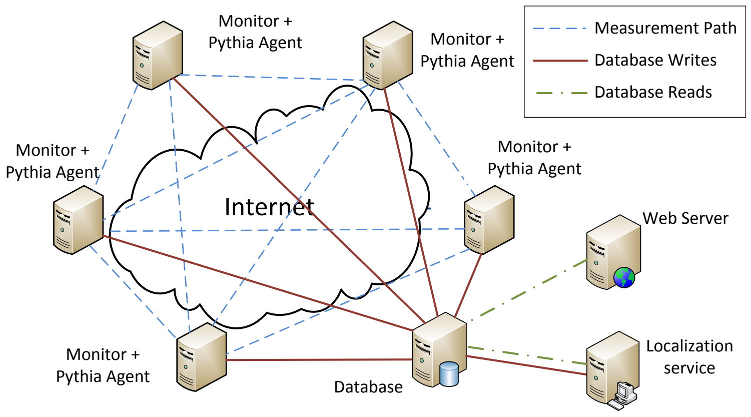


Figure 1. Architecture of a PuNDIT/Pythia deployment over a legacy monitoring infrastructure. We deploy lightweight agents on the monitors, and delegate storage, indexing and compute intensive tasks to nodes outside the monitoring infrastructure.

problems.

Pythia algorithms identify problems that include standard ones like link failures and route changes, but also less obvious problems like router misconfiguration, intermittent congestion and underprovisioned buffers. Specifically, Pythia works on top of legacy monitoring infrastructure deployments to solve three objectives: *(i)* detect performance problems, *(ii)* diagnose root cause(s) of detected problems, and *(iii)* localize problems to network interfaces.

Pythia was designed to augment existing troubleshooting procedures that network operators use. Pythia provides operators with a near real time performance problem diagnosis and localization. Pythia focuses on short-term performance problems (shorter than five minutes). This is an important goal, since short-term problems may not be evident to the operator, unlike problems that last for hours; moreover, short-term problems may indicate a potential failure. Pythia provides operators with a visualization of problem summaries across the monitored network. Operators can extend the diagnosis functionality of Pythia by adding new definitions for performance problems based on experience or knowledge of the network.

Pythia works with legacy monitoring infrastructure. Pythia was designed to scale to hundreds (potentially thousands) of monitoring nodes, without affecting the accuracy and timing of measurement processes on the monitoring nodes. For PuNDIT we have deployed a testbed to validate that our PuNDIT agent doesn't adversely impact any of the measurements a perfSONAR toolkit node makes.

There are design constraints when we work with legacy monitoring infrastructure. For PuNDIT, we decentralize computation (detection, diagnosis and localization) as much as possible, and at the same time, we implement efficient algorithms as agents at monitoring nodes. We implement computation that requires a view of measurements across multiple monitors using a central *bulletin board*, instead of communications between monitors.

Agent: We introduce a *lightweight* PuNDIT agent process on each monitoring node. Recall that each monitoring node records measurements as it receives probing packets generated by other monitoring nodes. The agent interfaces with these measurements in real time to: *(i)* detect if there is a performance problem at any point of time on each monitored path, *(ii)* diagnose root cause(s) of detected performance problems, and *(iii)* write real time diagnosis and summaries of path performance to the central database. We designed efficient algorithms and optimize CPU-bound logic in the agent process. On an average, the agent process takes about 60 μs to process each measurement; this allows the agent to scale to a large number of monitored paths.

We note that the typical path sampling rate in perfSONAR deployments is 10 Hz.
Database: The agents write problem diagnosis and timeseries summaries to a central PuNDIT server hosting a relational database. The database is used as a bulletin board for agents to interface diagnosis information from other monitoring nodes, for localization and for generating the front end. We require data from other agents to diagnose certain classes of pathologies. The timeseries summaries are used for localization of performance problems. In our initial implementation, we have used a MySQL database; we are looking at methods to horizontally scale the data and to optimize the read paths.
Front end: PuNDIT will use a browser-based front end to show summary statistics and visualizations of performance problems. We are implementing this functionality using a PHP engine on a web server; the PHP code interfaces with the database. The dashboard shows an overview of the health of the network, and allows operators to examine specific results or network paths in more detail using filters. The dashboard includes charts showing diagnoses aggregated by type and number of occurrences, and heatmaps visualizing frequency of diagnoses. The front end shows localization data using a network map, with a heatmap overlay for problem frequency. Operators will be able to select paths or IP interfaces to view problem summaries. While the information provided is not sensitive we do expect to authorize access via x509 credentials from IGTF certifed CAs to track access and limit visibility.

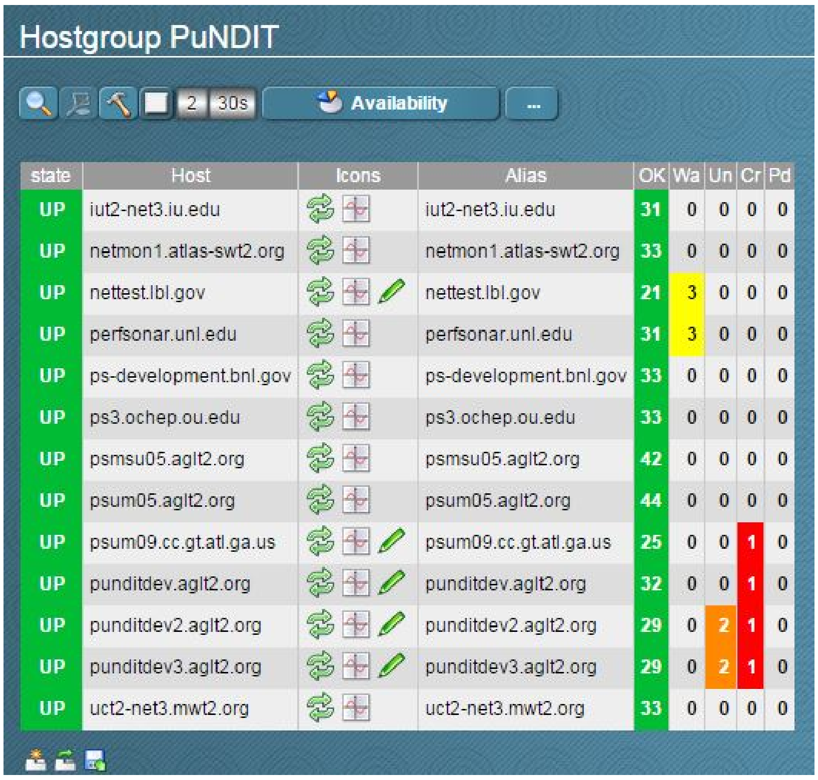
3. Current Work
The PuNDIT project started in September 2014 and is currently working in a number of areas targeted at reaching our project goals. We are hosting the project code in GitHub (see `https://github.com/pundit-project/pundit`) and have setup a web page to provide project tracking and organization. In the following sub-sections we will briefly describe our ongoing work.

3.1. Pythia Adaptation for perfSONAR 3.4
One of the first tasks we needed to undertake was updating the Pythia code, originally developed with perfSONAR v3.3, to work as the core of a new PuNDIT agent on the new v3.4 release of perfSONAR. In v3.4 there were many changes to various toolkit components as well as a migration from web-services to a RESTful API. The measurement archive technology was changed to use a new ESnet developed datastore called Esmond, which couples a PostgreSQL relational database with Cassandra. The code changes required to work with v3.4 are mostly implemented and are being tested on the PuNDIT testbed (see Subsection 3.2).

As part of this adaptation, we are also verifying the the resulting PuNDIT agent doesn't impact the primary purpose of the toolkit to make reliable, consistent network measurements. Of course we will also be evaluating the network problem sensitivity and accuracy at the same time.

3.2. Testbed and Development Systems
A very important part of PuNDIT is verifying that the older Pythia algorithms continue to work correctly with newer perfSONAR components while ensuring that PuNDIT doesn't adversely impact perfSONAR services. To do this we have setup a PuNDIT testbed composed of perfSONAR testbed instances from OSG and WLCG. This testbed was originally created from older perfSONAR instances at various institutions to test new releases of perfSONAR at a representative scale. PuNDIT was fortunate to get most of those sites to agree to extend their role to also test PuNDIT as we develop it.

The testbed currently has 13 members, 10 of which are running a full mesh of OWAMP tests between each other. Figure 2 shows the list of our testbed instances which span the United States, east coast to west coast and north to south. The 3 members not running OWAMP tests are used to develop and prototype PuNDIT changes and host various project related services we mention below.



state	Host	Icons	Alias	OK	Wa	Un	Cr	Pd
UP	iut2-net3.iu.edu	⟳ ⊞	iut2-net3.iu.edu	31	0	0	0	0
UP	netmon1.atlas-swt2.org	⟳ ⊞	netmon1.atlas-swt2.org	33	0	0	0	0
UP	nettest.lbl.gov	⟳ ⊞ ✎	nettest.lbl.gov	21	3	0	0	0
UP	perfsonar.unl.edu	⟳ ⊞	perfsonar.unl.edu	31	3	0	0	0
UP	ps-development.bnl.gov	⟳ ⊞	ps-development.bnl.gov	33	0	0	0	0
UP	ps3.ochep.ou.edu	⟳ ⊞	ps3.ochep.ou.edu	33	0	0	0	0
UP	psmsu05.aglt2.org	⟳ ⊞	psmsu05.aglt2.org	42	0	0	0	0
UP	psum05.aglt2.org	⟳ ⊞	psum05.aglt2.org	44	0	0	0	0
UP	psum09.cc.gt.atl.ga.us	⟳ ⊞ ✎	psum09.cc.gt.atl.ga.us	25	0	0	1	0
UP	punditdev.aglt2.org	⟳ ⊞ ✎	punditdev.aglt2.org	32	0	0	1	0
UP	punditdev2.aglt2.org	⟳ ⊞ ✎	punditdev2.aglt2.org	29	0	2	1	0
UP	punditdev3.aglt2.org	⟳ ⊞ ✎	punditdev3.aglt2.org	29	0	2	1	0
UP	uct2-net3.mwt2.org	⟳ ⊞	uct2-net3.mwt2.org	33	0	0	0	0

Figure 2. The summary OMD/Check_mk monitoring of our PuNDIT testbed instances that the PuNDIT developers use to monitor testbed status and track the impact of various PuNDIT components as they are incorporated into the testbed.

As shown in Figure 2 we have setup an installation of the Open Monitoring Distribution (`http://omdistro.org/`) to allow us to monitor and measure the performance of the services and hosts involved in the testbed. The Check_mk component of OMD makes it very easy for us to automatically find, monitor and graph services and operating system components. One of our primary use-cases for this is to measure the impact of adding PuNDIT components to the baseline perfSONAR toolkit instances. Using Check_mk, we are able to track many components: host memory, cpu, load, network use, disk I/O, disk storage, thread use as well as the state of the various perfSONAR and PuNDIT services.

We are monitoring the PuNDIT testbed results using MaDDash (ESnet's Monitoring and Diagnostic Dashboard) which tracks the perfSONAR metrics produced by the testbed so we can verify measurements are not impacted as we add-to and modify the testbed.

To track our project milestones and work plan we have deployed OpenProject on a virtual-machine host at Michigan. This is important to track progress and identify where we need to focus to keep the project on track.

3.3. Localization

One of the PuNDIT project's primary goals it the ability to localize problems along wide-area network paths. The OSG/WLCG perfSONAR deployment gives us access to approximately 150 end-sites, distributed around the world. Each of the instances is performing network measurements with a large fraction of the total sites resulting in a very large number (up to

150^2) of network paths being monitored. This provides us with a unique dataset capable of being mined to identify problematic network-hops by correlating network metrics indicating problems with their corresponding traceroutes. To analyze this in PuNDIT, we delegate localization to a central server, since our localization algorithms require measurement data from the complete monitoring mesh (this simplifies the design by eliminating inter-agent communication). We are exploring message queuing systems to serve as the mechanism to get data reliable sent to the localization server.

The localization process interfaces with timeseries summaries for each monitored path, and with topology data (collected from periodic traceroutes between monitors). The timeseries summary consists of a (delay, loss, reordering) metric-tuple for back-to-back 5s time windows of measurements. The localization service periodically queries monitors for the most recent topology data and caches it before running localization.

3.4. Paris-Traceroute

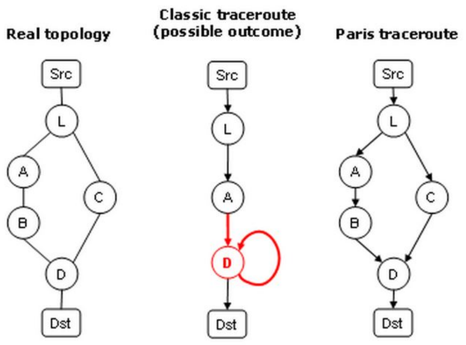


Figure 3. For a given network topology using load-balancing, traceroute can provide a misleading topology as shown above. Paris traceroute was designed to deliver an accurate topology in this case. Figure from `http://www.paris-traceroute.net/`.

To properly localize problems PuNDIT needs to rely upon accurate network topologies associated to each set of network metric measurements. The typical tool to identify network topology at the IP layer is traceroute. However, traceroute has a few shortcomings that can lead to mis-represented topologies as shown in Figure 3.

We have contacted the Paris traceroute[12] developers about updating their tool so it could be incorporated as an optional topology tool distributed with the perfSONAR toolkit. As a result of this contact and subsequent discussions with the perfSONAR developers, the Paris traceroute developers have made a new effort to update their code so it will be suitable to include in perfSONAR.

3.5. Lightweight Bandwidth Estimation

A new tool to be developed in PuNDIT is a lightweight bandwidth estimator, which aims to be a non-intrusive application-layer method of estimating the available bandwidth between a client and server using TCP streams. Additionally, it may be able to infer whether the transfer is limited by a TCP buffer, or whether an intermediate buffer along the path is over-provisioned (bufferbloat).

Our approach leverages the fast increase in congestion window size during TCP slow start. We initiate a new TCP stream in one direction, sending data as fast as possible. Simultaneously, a loop performing RTT estimation will be started, measuring the time to exchange a single byte between sender and receiver. On the receiver side, the received bytes will be sampled every 100ms to generate a throughput estimate. We infer a loss when the timeseries of the throughput shows a decrease by half or more, and stop the transfer, producing a bandwidth estimate. Alternatively, if we find the throughput timeseries remains approximately constant for several seconds, we stop the transfer and examine the RTT timeseries: If it shows an increasing trend for that same time duration, there is a possibility of bufferbloat, otherwise it is possible that the receiver window is under provisioned. We repeat this test multiple times and perform clustering on the bandwidth estimates to produce a possible range for the available bandwidth.

The main issue with current methods of estimating bandwidth is the intrusiveness of the tests required when measuring high bandwidth-delay network paths. Tests like Iperf [13] and nuttcp [14] operate on the concept of saturating a network path for several seconds to generate a bandwidth estimate. For high speed research networks, this means transferring a large amount of data which can affect other transfers [15]. As a result, these tests are generally performed infrequently, reducing their effectiveness. We avoid this by stopping the transfer when a loss is inferred or a throughput plateau is detected, minimizing the amount of data transferred.

3.6. Bandwidth Analysis via Adaptive Plateau Detection

The Pythia component of PuNDIT only analyzes OWAMP data to find signatures of network problems and, as noted in the Architecture section above, is focused on near real-time problem detection. However there are other classes of network problems which are much more persistent yet still can escape detection for a long time. To allow PuNDIT to also be able to identify that class of problem, we are working on incorporating Adaptive Plateau Detection[16] processing on all the bandwidth data created within a PuNDIT deployment. Note the latency data doesn't need APD because it already has a simple shift detection in its diagnosis modules.

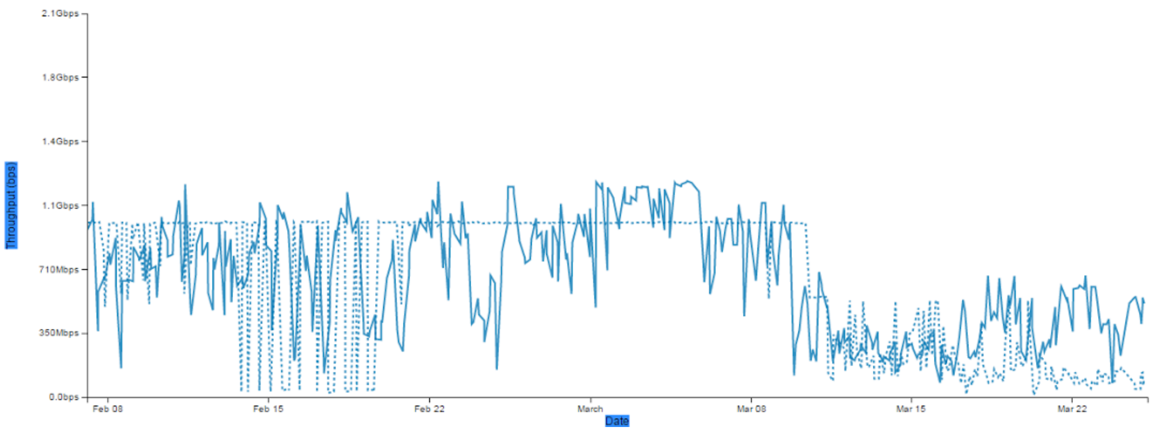


Figure 4. An example plot of measured bandwidth using iperf between two sites showing a significant change the could be easily identified by an Adaptive Plateau Detection algorithm.

As show in Figure 4, there is useful information about long-term changes in the wide-area network performance as measured by iperf bandwidth tests in the perfSONAR toolkit. Performance degradations, like the one shown, can often go undetected for weeks or months. When they are finally noticed the memory of changes made that may have contributed to the problem is often lost. We are working to automate the processing of all bandwidth measurements involved in a PuNDIT-monitored infrastructure so that we can quickly identify such issues which should lead to much quicker problem resolution.

4. Future Plans and Summary

The PuNDIT project has another 1.3 years to complete our work. We are targeting an initial integration with the perfSONAR Toolkit for the 3.6 release currently scheduled for January 2016. By that time we should have a robust PuNDIT agent that can be enabled through the regular perfSONAR toolkit interfaces and a well developed central server that can analyze topology and problem reports to localize their origin in the wide-area network.

An important topic we will handle in 2016 (the last year of the project) is developing an interface to third party alerting and alarming software, so that PuNDIT can provide customized notification of problems to the appropriate destinations at any scale.

Acknowledgments

The authors gratefully acknowledge the support of the National Science Foundation and our collaborators from the perfSONAR team and the OSG and WLCG communities. This work is supported by NSF Grants 1440571 and 1440585.

References

[1] PuNDIT project homepage. `http://pundit.gatech.edu/`, 2015. [Accessed 15-May-2015].

[2] perfSONAR primary website. `http://www.perfsonar.net/`, 2015. [Accessed 15-May-2015].

[3] Ruth Pordes, Don Petravick, Bill Kramer, Doug Olson, Miron Livny, Alain Roy, Paul Avery, Kent Blackburn, Torre Wenaus, Frank Wrthwein, Ian Foster, Rob Gardner, Mike Wilde, Alan Blatecky, John McGee, and Rob Quick. The open science grid. *Journal of Physics: Conference Series*, 78(1):012057, 2007.

[4] Jamie Shiers. The worldwide LHC computing grid (worldwide LCG). *Computer Physics Communications*, 177(1-2):219–223, July 2007.

[5] A. Hanemann, J. Boote, E. Boyd, J. Durand, L. Kudarimoti, R. Łapacz, D. Swany, S. Trocha, and J. Zurawski. PerfSONAR: A service oriented architecture for multi-domain network monitoring. *Service-Oriented Computing-ICSOC 2005*, pages 241–254, 2005.

[6] H. Yan, L. Breslau, Z. Ge, D. Massey, D. Pei, and J. Yates. G-RCA: a generic root cause analysis platform for service quality management in large IP networks. In *ACM SIGCOMM CoNEXT*, 2010.

[7] A. Mahimkar, J. Yates, Y. Zhang, A. Shaikh, J. Wang, Z. Ge, and C.T. Ee. Troubleshooting chronic conditions in large IP networks. In *ACM SIGCOMM CoNEXT*, 2008.

[8] A.A. Mahimkar, Z. Ge, A. Shaikh, J. Wang, J. Yates, Y. Zhang, and Q. Zhao. Towards automated performance diagnosis in a large IPTV network. In *ACM SIGCOMM CCR*, volume 39, pages 231–242, 2009.

[9] T. Qiu, Z. Ge, D. Pei, J. Wang, and J. Xu. What happened in my network: mining network events from router syslogs. In *ACM SIGCOMM IMC*, 2010.

[10] P. Bahl, R. Chandra, A. Greenberg, S. Kandula, D.A. Maltz, and M. Zhang. Towards highly reliable enterprise network services via inference of multi-level dependencies. In *ACM SIGCOMM CCR*, volume 37, pages 13–24, 2007.

[11] P. Kanuparthy, D. Lee, W. Matthews, C. Dovrolis, and S. Zarifzadeh. Pythia: detection, localization, and diagnosis of performance problems. *Communications Magazine, IEEE*, 51(11):55–62, November 2013.

[12] Brice Augustin, Xavier Cuvellier, Benjamin Orgogozo, Fabien Viger, Timur Friedman, Matthieu Latapy, Clémence Magnien, and Renata Teixeira. Avoiding traceroute anomalies with paris traceroute. In *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, IMC '06, pages 153–158, New York, NY, USA, 2006. ACM.

[13] iperf3: A tcp, udp, and sctp network bandwidth measurement tool. `https://github.com/esnet/iperf`, 2015. [Accessed 14-May-2015].

[14] NutTCP Welcome Page. `http://www.nuttcp.net/Welcome%20Page.html`, 2014. [Accessed 14-May-2015].

[15] Ajay Tirumala, Les Cottrell, and Tom Dunigan. Measuring end-to-end bandwidth with iperf using web100. In *Web100, Proc. of Passive and Active Measurement Workshop.* Citeseer, 2003.

[16] P. Calyam, Jialu Pu, W. Mandrawa, and A. Krishnamurthy. Ontimedetect: Dynamic network anomaly notification in perfsonar deployments. In *Modeling, Analysis Simulation of Computer and Telecommunication Systems (MASCOTS), 2010 IEEE International Symposium on*, pages 328–337, Aug 2010.