# $S^3$: Scalable Network Sensing Service

Praveen Yalagandula, Puneet Sharma, Sujata Banerjee, Sung-Ju Lee, and Sujoy Basu
Mobile & Media Systems Lab
Hewlett-Packard Laboratories
Palo Alto, CA 94304

## I. INTRODUCTION

Efficient operation of large-scale distributed networked systems such as enterprise networks and grid systems remains a challenge for a number of reasons. Most commercial management systems monitor only a subset of system/application metrics and at relatively coarse grain timescales to be able to collect and process the measurements at a central location. This impedes decision making at very fine timescales, resulting in the inability to perform agile control and adaptation to network dynamics which is important for several emerging applications like interactive multi-media services and early-anomaly detection systems. What is needed is a scalable sensing service that the management subsystem as well as individual applications/services can subscribe to and securely get customized information at whatever timescale or periodicity necessary for their purposes.

Scalable operation requires the sensing service to work in a fully decentralized manner, with no perceivable downtime, without conducting unnecessary duplicate measurements, and without measuring every resource/application metric continuously. Such sensing can be used for a number of management tasks such as -

- Detecting failures or anomalous behavior: link failures, node failures, service failures, root cause diagnosis for poor application/service performance, suspicious network activity such as DoS attacks, viruses, worms, and spam.
- Resource placement: adaptive placement based on observed performance (e.g., choosing a host with better availability for running a key service), placement based on observed loads (replicate services based on usage conserving key resources such as power).
- Better performance: fast network path selection for quick content transfers.

In this demo, we will be showing a Scalable network Sensing Service ($S^3$) that monitors state of large-scale networked systems and allows the network management software as well as individual applications/services to securely obtain customized monitoring information at time scales required by the respective systems. For scalable operation, S3 provides the sensing service in a decentralized manner, eliminates unnecessary duplicate measurements by consolidating sensing requirements of different applications, and provides inference engines to estimate network metrics with high accuracy while avoiding quadratic all-pair measurements load.

## II. SYSTEM ARCHITECTURE

The key features of our approach are:

- Embedded Sensing: Embedding the sensing in the network itself is different from the current approach collecting data at a central location and issuing control actions based on that data. Our approach not only allows faster detection of faults but also restricts the proliferation of a problem from one domain to another.
- Web-services Based Access: We argue that the standard SNMP based approach for monitoring networks increases the response time. Sensing based on secure web-services can leverage WSDM's MUWS standards. This approach is extensible and configurable as the requirements change with new networking technologies being introduced. Figure 1 shows the web-based output from a deployed sensing pod.
- Sensing Inference Engines The task of collecting complete information about network metrics is an immense task both in terms of the infrastructure requirements as well as the measurement traffic. As the overall system grows, the total number of individual metrics to be tracked increases exponentially. There are a large number of attributes (network and machine) that need to be tracked. Some attributes are highly dynamic (e.g., available bandwidth on an end-to-end path or current load on a machine) while others are slowly varying or static (processor type or OS version on a machine). Measuring everything at the smallest possible timescale is impossible and smart estimation techniques are needed that are able to give an accurate approximation of the global state while incurring low measurement overhead. Scalable inference engines are used to infer and estimate complete information about the relevant network metrics based on partial information collected from the sensing pods. We have built some of the inference engines for scalable network proximity/latency estimation (NetVigator) [3] and bandwidth inference [1]. Figure 2 shows the accuracy of the proximity estimation tool NetVigator. While triangular inequality based inference algorithms work well for latency estimation, this heuristic does not hold for all metrics.

## III. DEMONSTRATION DETAILS

In this demonstration, we show novel scalable latency and proximity estimation algorithms as well as the distributed sensing infrastructure deployed on PlanetLab [2]. $S^3$ architecture is a loosely coupled Service Oriented Architecture

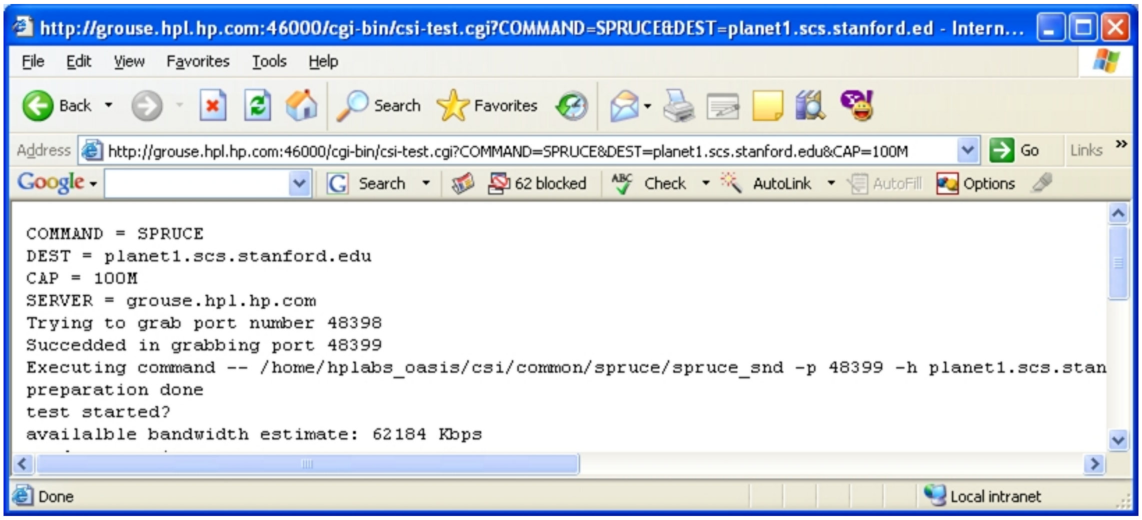


Fig. 1. Sensor Pod: Available Bandwidth measurement

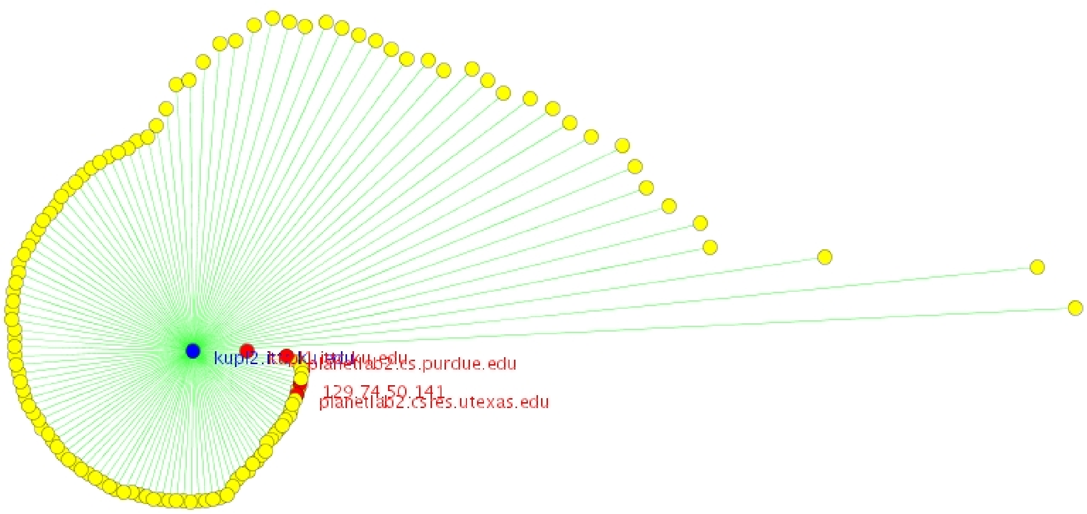


Fig. 2. Proximity estimation visualization

(SOA) based on principles similar to those of MUWS (Management using Web Services). This architecture comprises of three components: Sensing Pods, Sensing Inference Engines, and Sensing Information Management Backplane. Sensor pods are implemented as `cgi` scripts accessible through any webserver that supports `cgi`. We currently use Boa (`http://www.boa.org`), a light-weight open source webserver. Our current implementation has a wide variety of sensors (latency, proximity, capacity, available bandwidth, lossrate, etc.), that leverage several open source network monitoring tools for measuring these different network path metrics. The proximity and latency information has been computed using NetVigator, a scalable network proximity and latency estimation tool developed at HP Labs. NetVigator uses information obtained from probing a small number of landmark nodes and intermediate routers (termed milestones) that are discovered en route to the landmarks, to identify the closest nodes. With very little additional probing overhead, Netvigator uses distance information to the milestones to accurately locate the closest nodes. Performance of the Netvigator prototype has been evaluated on PlanetLab and in HP's intranet. We will demonstrate the ability of $S^3$ to provide personalized measurements to applications (Figure 1). We will also demonstrate the accuracy of the latency and proximity estimation algorithm using the Guess graph visualization tool (Figure 2). Finally, using live measurements from PlanetLab, we will demonstrate how the $S^3$ infrastructure enables fast proximity estimation. We will also demonstrate the data being collected from $S^3$ deployment

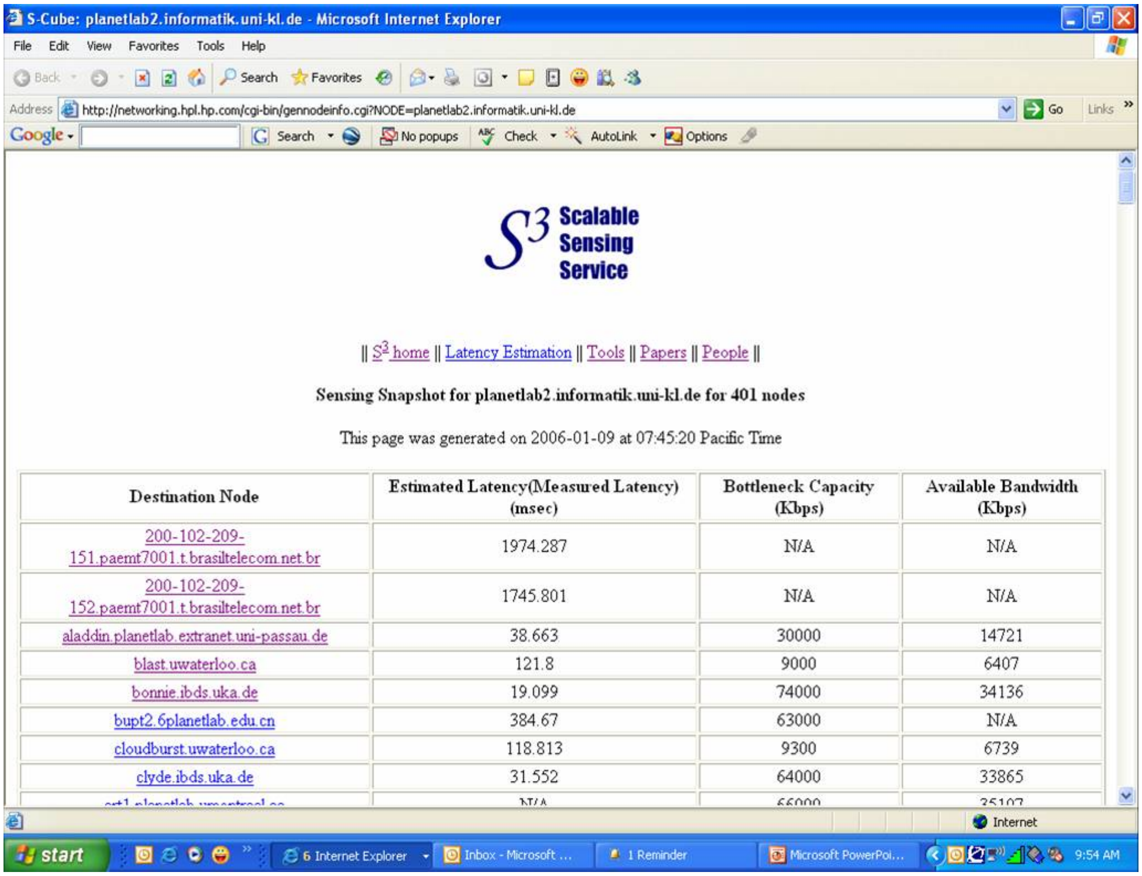


Fig. 3. S3 data page

on planetlab as shown is Figure 3.

## IV. Conclusion

S$^3$ is being used in a recently funded DARPA Internet Control Plane project called CHART(Control for High Throughput Adaptive and Resilient Transport). In CHART, the goal is to achieve high throughput TCP transfers and various networking metrics such as delay, loss, and bandwidth information at very fine time scales are needed to build high throughput network overlay paths. The demo and poster will form an interesting backdrop for discussing issues in scalable network sensing such as measurement, management, inference, and data dissemination. Measurement related issues deal with probing the network for gathering information about different network properties such as pairwise latency between different nodes. Inference techniques are used to model/estimate complete information from the partial information collected during the probing/measurement phase.

## References


[1] S.-J. Lee, P. Sharma, S. Banerjee, S. Basu, and R. Fonseca, “Measuring bandwidth between PlanetLab nodes,” in *Proceedings of the PAM 2005*, Boston, MA, March 2005, pp. 292–305.

[2] PlanetLab, http://www.planet-lab.org.

[3] Z. Xu, P. Sharma, S.-J. Lee, and S. Banerjee, “Netvigator: Scalable network proximity estimation,” HP Laboratories, Tech. Rep., 2005.