Anatomy of an International Exchange Point: Distributed Network Monitoring Using MonALISA and NetFlow[1]

Xun Su[1], Jose Fernandez[2], Ernesto Rubi[2], Iosif Legrand[1], Heidi Alvarez[2], Julio Ibarra[2]

[1]High Energy Physics Department, California Institute of Technology, Pasadena, CA. 91125
xsu@hep.caltech.edu, Iosif.Legrand@cern.ch

[2]CIARA, Florida International University, Miami, FL. 33199
josef@fiu.edu, ernesto@cs.fiu.edu, heidi@fiu.edu, Julio@fiu.edu

Abstract: In this paper we present a distributed network monitoring system, which exploits MonALISA (Monitoring Agents in A Large Integrated Services Architecture), a distributed web service delivery infrastructure designed to collect and process the network monitoring information. We augment the capability of MonALISA with FlowTools, the popular NetFlow data analysis toolset. We demonstrate how to integrate MonALISA and FlowTools via an UDP-listening agent ApMon, and highlight a case study of AMPATH, an international exchanging point located in Miami and serving a number of South American National Research and Education Networks (NRENs). Our experience showcases the elegant design philosophy of a scalable distributed service deployment platform coupled with the open-source traffic analysis tools and its impact on the daily operation of the production networks.

Keywords: NetFlow, MonALISA, FlowTools, Network monitoring.

1. INTRODUCTION

As the Internet expands both in its scope, reach and capacity, it becomes evident that there is a strong need to develop a distributed network monitoring infrastructure that can be scaled to support various network topology, traffic granularity and user applications. NetFlow[1] is a widely deployed router-based traffic monitoring mechanism. FlowTools[2] is an open-source NetFlow analysis toolset underlying the data gathering and analysis infrastructure of our project. It is our main motivation to effectively use NetFlow to gain crucial understanding of the traffic characteristics of the networks we operate. In particular, we are interested in understanding how to exploit the key advantages and avoid drawbacks of NetFlow-based traffic analysis by augmenting it with a distributed service-deployment platform. Indeed the focus of our work is to integrate MonALISA[3], a distributed monitoring system based on JINI/JAVA and WSDL/SOAP technologies. MonALISA's flexibility as a framework to gather, store and distribute network data collected was crucial to the success of our investigation and it shall become apparent throughout the course of this paper. The MonALISA framework provides a distributed monitoring service that not only is closely integrated with our monitoring and

[1]This work is supported by National Science Foundation's Strategic Technologies for the Internet (STI): Research Experience for Undergraduates Award No. 331112 and a Cisco Systems University Research Program (URP) grant.

data distribution philosophy but also acts as a dynamic service system. The goal is to provide the monitoring information from large and distributed systems in a flexible and self-describing way as part of a loosely coupled service architectural model to perform effective resource utilization in large, heterogeneous distributed centers. A salient feature of the MonALISA framework lies in its capability to integrate existing monitoring tools and procedures to collect parameters related to computational nodes, storage devices and network performance. A critical part to the research that we undertook is to integrate the parsed NetFlow data into MonALISA, which would make available the information in a distributed manner through mobile service agents. We will incorporate real-time as well as historical information in our system to improve the understanding of the traffic statistics data from the networks being monitored.

A significant part of the paper is dedicated to the use of our system to analyze network traffic behavior across a real-world production network, AMPATH. The AMericasPATH (AMPATH) network is an FIU project sponsored in part by the US National Science Foundation CISE directorate, in collaboration with Global Crossing and other telecommunications product and service providers. Using Global Crossing's terrestrial and submarine optical-fiber networks, AMPATH is interconnecting the research and education networks in South and Central America and the Caribbean to US and non-US research and education networks via Internet2's Abilene network. The purpose of the AMPATH project is to allow participating countries to contribute to the research and development of applications for the advancement of Internet technologies. The mission of AMPATH is to serve as the pathway for Research and Education networking in the Americas and to the world and to be the International Exchange Point for Latin America and the Caribbean research and education networks. Additionally AMPATH fosters collaboration for educational outreach to underserved populations both in the US and abroad. The AMPATH pathway serves as the bridge between Central and South American National Research Networks (NRENs) and the world's research and education networks. With the multiplicity of complex networked systems and educational activities served by AMPATH's wide-ranging infrastructure a strong demand for high availability and engineering collaboration arises, which is met through the use of various monitoring agents to provide a strong factual foundation to troubleshooting. Likewise, deciphering the everyday activities of our peers is achieved with a distributed approach to data gathering and dissemination.

2. ANATOMY OF AN INTERNATIONAL EXCHANGE POINT

Figure 1 illustrates AMPATH's current architecture. We will use simple NREN names to identify our international peers.[2] It is this network that served as the backdrop for our study. Two core routers exist: a Cisco GSR 12012 as well as a Juniper M10, both routers have NetFlow accounting enabled and are designed to export this data to a collection workstation.

[2] For more detailed information please visit http://mrtg.ampath.net.

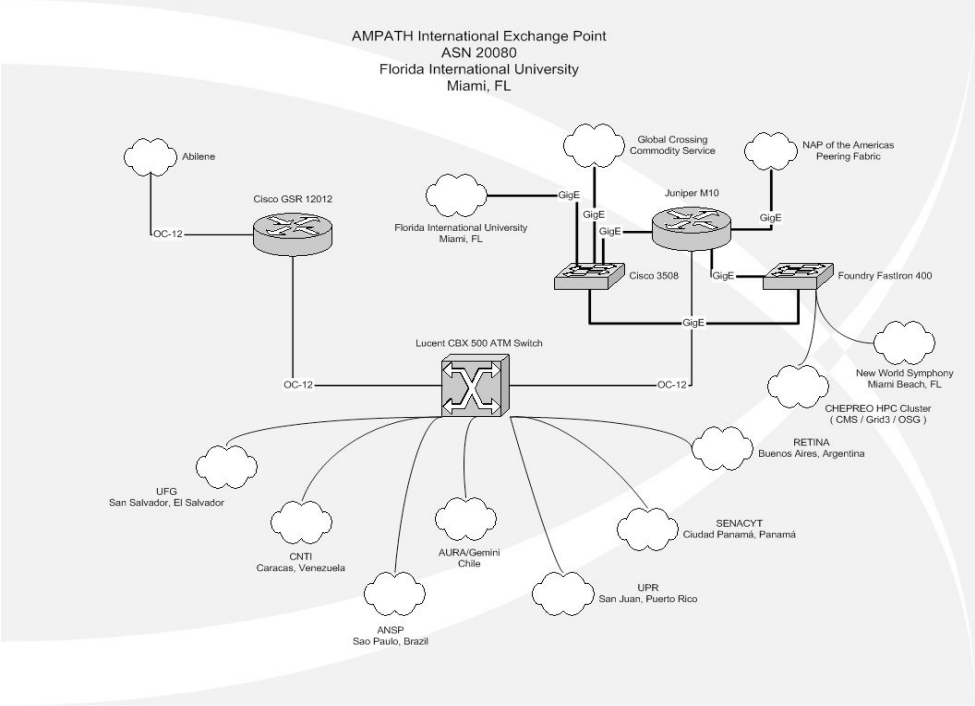

Figure 1: AMPATH schematic architecture.

3. NETFLOW AND FLOWTOOLS

NetFlow was originally developed as a switching path and today it is primarily used for network accounting. Flow records are generated and exported by a router. Each flow record contains information about all packets that are categorized to have the same combination of the following IP header fields:

- Source IP address
- Destination IP address
- Source port
- Destination port
- Layer 3 protocol type
- TOS byte
- Input logical interface

NetFlow records only unidirectional traffic in-bound to any interface on the router. Note however, even though this traffic is unidirectional NetFlow accounts for all traffic going in and out of the router by recording both transit traffic and traffic destined for the router. By storing only the router's flow-level information and neglecting payload it becomes feasible to summarize and "sketch" large amount of data traffic.

A key technical note is the sampling mode at which the router is running. In the GSR's case a 1-100 sampling rate is specified. This means that for every 100 packets processed by the forwarding engine or route processor there will be one packet extracted and reported to the NetFlow process running at the router. This is the lowest allowed sampling rate on our GSR running NetFlow v5 and clearly we can see that this causes

limitations on the analysis of network data. It is not uncommon to have short host-to-host sessions where the overall transmission does not exceed 100 packets. It is beyond the scope of this paper to discuss NetFlow sampling algorithms but based on our experience it is likely that if the transmission is 100 packets or less NetFlow will not account for it. This introduces a margin of error to any analysis of data flows but especially to those UDP flows transmitted over our core. With their inherent error correction mechanisms TCP flows are less prone to being ignored by the collection process.

FlowTools is a collection of programs and libraries used to collect and process NetFlow data. These tools allow users to process stored flow data from a series of command line interfaces. Commands like flow-filter and flow-sort allow the user to filter and sort NetFlow data by IP address, port, AS number and any other parameter present in that data collected. The data is presented on the command line in a table format. However these tools do not provide a dynamic way of dynamically monitoring flow data. Through the use of MonALISA we have used NetFlow data collected and processed by Flow Tools to create a graphical interface by which to view certain characteristics of the AMPATH network in a near-real time fashion. For our analysis we implemented flow-tools version 0.66 on a dual Xeon 2.66 GHz system with a copper Gigabit Ethernet network connection to FIU's campus network. The operating system of choice was Fedora Core 2.

Having discussed the data gathering techniques in the following we focus on the data dissemination mechanism. For the rationales that we detail in the next section MonALISA is our platform of choice for this purpose.

4. MONALISA AND APMON

MonALISA (Monitoring Agents in A Large Integrated Services Architecture) is a distributed services architecture to collect, process and act upon real-time monitoring information. While its initial target field of application is networks and Grid systems used by the global high energy and nuclear physics collaborations, MonALISA is broadly applicable to many fields of data intensive science, and to the monitoring and management of major research and education networks. MonALISA is based on a scalable dynamic distributed services Architecture, and is implemented in Java using JINI[21] and WSDL[22] technologies. The scalability of the system derives from the use of a multi-threaded engine to host a variety of loosely coupled self-describing dynamic services, and the ability of each service to register itself and then to be discovered and used by other services or clients that require such information. The framework integrates many existing monitoring tools and procedures to collect parameters describing computational nodes, applications and network performance. Specialized mobile agents are used in the MonALISA framework to perform global optimization tasks or help improve the operation of large distributed system by performing supervising tasks for different applications. MonALISA is currently running around the clock monitoring several Grids and distributed applications on approximately 150 sites.

The core of the MonALISA monitoring service is based on a modular system design used to perform the data collection tasks in parallel, independently. The modules used for collecting different sets of information, or interfacing with other monitoring tools, are

dynamically loaded and executed in independent threads. In order to reduce the load on systems running MonALISA, a dynamic pool of threads is created once, and the threads are then reused when a task assigned to a thread is completed. This allows one to run concurrently and independently a large number of monitoring modules, and to dynamically adapt to the load and the response time of the components in the system. If a monitoring task fails or hangs due to I/O errors, the other tasks are not delayed or disrupted, since they are executing in other, independent threads. A dedicated control thread is used to properly stop the threads in case of I/O errors, and to reschedule those tasks that have not been successfully completed. A priority queue is used for the tasks that need to be performed periodically.

A schematic view of this mechanism of collecting data is shown in Figure 2. This approach makes it relatively easy to monitor a large number of heterogeneous nodes with different response times, and at the same time to handle monitored units that are not responding without affecting other measurements. The clients, other services or agents can get any real-time or historical data by using a predicate mechanism for requesting or subscribing to selected measured values. These predicates are based on regular expressions to match the attribute description of the measured values a client is interested in. They may also be used to impose additional conditions or constraints for selecting the values. In case of requests for historical data, the predicates are used to generate SQL queries to the local database. The subscription requests create a dedicated thread, to serve each client. This thread performs a matching test for all the predicates submitted by a client with the measured values in the data flow. The same thread is responsible to send the selected results back to the client as compressed serialized objects. Having an independent thread per client allows sending the information they need, in a fast and reliable way, and it is not affected by communication errors that may be occurring at other clients. In case of communication problems these threads will try to reestablish the connection or to clean up the subscriptions for a client or a service that is no longer active.

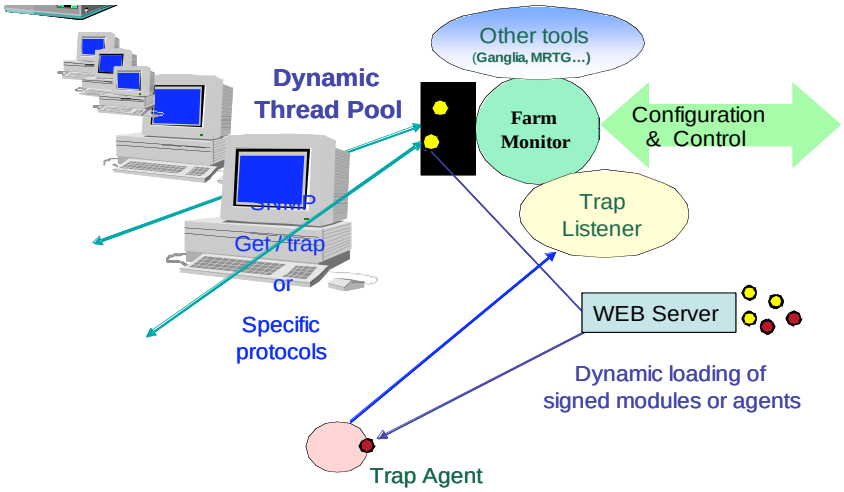

Figure 2: A schematic view of the MonALISA data collection mechanism based on a multi-threaded engine.

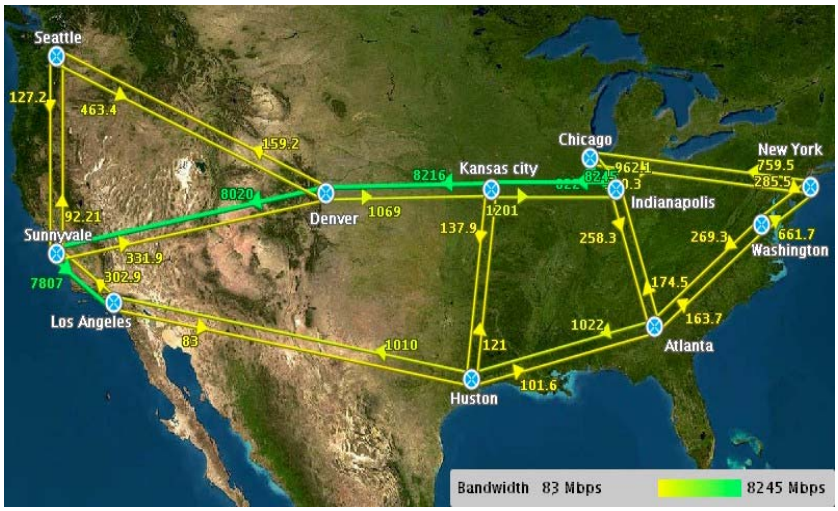

Figure 3: The MonALISA monitoring service for Abilene, shown at a time we injected more than 8 Gigabits/sec.

Figure 3 is a snapshot of the MonALISA monitoring network for Abilene network of the Internet2. It shows all the active nodes running MonALISA services for this particular "farm", discovered automatically by a graphical MonALISA client. The client can display the real time global views and connectivity, as well as the usage and load of the farms. In this particular instance we captured a highly intensive data transfer event on June 19th, 2004 where a group of 12 disk servers in CERN concurrently sent TCP traffic via LHCNet and Abilene to their destinations in Caltech. Note that in this case MonALISA reported a throughput reaching 8.4 Gbps on the Abilene links from Chicago → Kansas City → Denver → Sunnyvale → Los Angeles.

A salient feature of the MonALISA design is its extensibility. It allows user-defined monitoring modules, specific to user-specified system and network information, to be easily implemented and integrated in the MonALISA framework. This facilitates the work reported in this paper, as well as our ongoing project to integrate NLANR PMA[5] real-time packet trace analysis and MonALISA. From a systems point of view, MonALISA provides scalable architectural support for collecting, visualizing and responding to the operating conditions of large-scale distributed systems, and it is especially suited for monitoring and controlling large computing systems and networks used in Grid applications.

ApMon[4] is an Application Programming Interface that facilitates user-specific applications to interact with the MonALISA services. ApMon allows any application to send parameterized monitoring information to MonALISA. The data can be sent as UDP packets to multiple hosts running the MonALISA service. Through the use of ApMon MonALISA services can receive parameterized data in (name, type, value) tuples. When transmitting a data point to MonALISA the application specifies the name of the parameter about to be sent, the type of the parameter (string, object, integer, double) and the actual value of the parameter. The ApMon module on the MonALISA service will then receive this data and create any needed fields on its database for new parameters or populate existing fields if a particular parameter name already exists. The resulting data stored in MonALISA is a set of parameters and the values of those parameters over time.

MonALISA then provides an interface by which to view one or multiple parameters in a real-time or historical graph.

5. INTEGRATING MONALISA AND FLOWTOOLS VIA APMON

NetFlow allows for the monitoring of a large number of parameters. For this project we decided to limit the parameters monitor to the following:

- UDP/TCP port destination/source traffic
- IP destination/source traffic
- IP protocol traffic
- IP Next Hop traffic
- AS destination/source traffic
- Prefix destination/source traffic

For most of these parameters we will be monitoring that total traffic in octets over a period of time. The initial period of time was 5 minutes. In the flow-capture startup script above the parameter –n288 indicates that we want flow-capture to generate 288 files per day which results in a new file generated very 5 minutes.

Our application will use the most recent file to retrieve the desired parameters. This will result in parameterized NetFlow data being retrieved by our application reflecting 5-minute averages for each of the monitored parameters. Hence the value of each parameter will represent the total number of octets associated with that particular parameter over a 5-minute period. We will use flow-stat to collect all of the parameters specified above. Below is a sample output using flow-stat. As we can see for this particular 5-minute interval the HTTP port (80) was most heavily used at 11274635 octets roughly 10.75 MB.

```
$ flow-cat ft-v05.2005-01-19.135207-0500 | flow-stat -f 5 -S 2
#  --- ---- ---- Report Information --- --- ---
#
# Fields:    Total
# Symbols:   Disabled
# Sorting:   Descending Field 2
# Name:      UDP/TCP destination port
#
# Args:      flow-stat -f 5 -S 2
#
#
# port      flows             octets            packets
#
80          42816             11274635           82997
25          2877              3770113           7623
2010        28                2543195           1747
4662        2095               2290946           3770
2009        28                2091493           1413
```

The application will parse the flow-stat report and generate a (name, value) pair which will be sent to MonALISA via ApMon. This name will be one of the following: (1) a port number or name; (2) an IP address; (3) a protocol number or name; (4) an AS number; (5) a Prefix.

6. CASE STUDY– MONITORING PEER TRAFFIC

In this section we demonstrate the use of our monitoring system in practice. We choose to examine a peer, the NWS Internet2 Gigabit Ethernet link that carries all HPC traffic to and from New World Symphony (NWS), a post-doctorate institution at Miami Beach that has AMPATH be its upstream provider of Internet2 traffic as well as commodity internet. Once MonALISA starts the screen presented in Figure 4 is a global view of all running sites / farms.

Figure 4 - MonALISA startup screen with 'test' group.

The MonALISA service that pertains to our particular study is titled I2Monitor. Choosing the I2Monitor farm, we are then presented with the AMPATH specific data we have chosen to integrate into our analysis, as shown in Figure 5. Two core routers (a Cisco GSR and a Juniper M10) are monitored. As an example we will present the data for the Juniper – NWS Internet2 connection. In Figure 6 by choosing Egress Source AS we view the current set of stored AS numbers belonging to flows leaving our Juniper router and destined to the New World Symphony network. With the "Parameters" corresponding to the AS numbers traversing our router destined to NWS we have a clear top-level view of flows of the network, and we can delve deeper into this data by showing a real-time plot or history plot of the AS data gathered. This is given in Figure 7. In Figure 8 we show the utilization of IPv6. We note that IPv6 data has not been a significant load on AMPATH during the time period specified.

7. DISCUSSIONS AND FUTURE WORKS

NetFlow data contains a rich amount of network information with a multitude of applications. Through the use of the distributed monitoring environment provided by

MonALISA and the reporting flexibility embedded in the FlowTools API it is possible to encapsulate and summary this data in visually friendly manner. In doing saw were able to create dynamic and real time views of the AMPATH Internet 2 network traffic and its behaviors. Direct application of this technology could be used to further understand network traffic behavior and trends in complex research networks.

The traffic views generated by MonALISA and the NetFlow application focused on the "top talking" flows. That is MonALISA only received the top x talking flows for a particular period of time. We saw that this technique produced a seemingly random setup of results. Further study should be made to understand this trend and determine its inherent properties.

One further technology that we intended to explore was that of the National Laboratory for Applied Network Research (NLANR) PMA (Passive Monitoring Agent) [5]. There are key differences between PMA data and NetFlow data worthy of serious research effort. With our results we hope to provide a stable platform from which networks of varying degrees can be closely monitored, their traffic patterns clearly identified and the appropriate decisions taken to rectify issues which negatively impact performance or augment those which have a positive impact on the delivery of service to an end user.

REFERENCES:

1. NetFlow Services Solution Guide, Cisco Systems Inc.
2. FlowTools: http://www.splintered.net/sw/flow-tools/.
3. MonALISA: http://monalisa.caltech.edu.
4. ApMon: http://monalisa.caltech.edu.
5. PMA: http://moat.nlanr.net

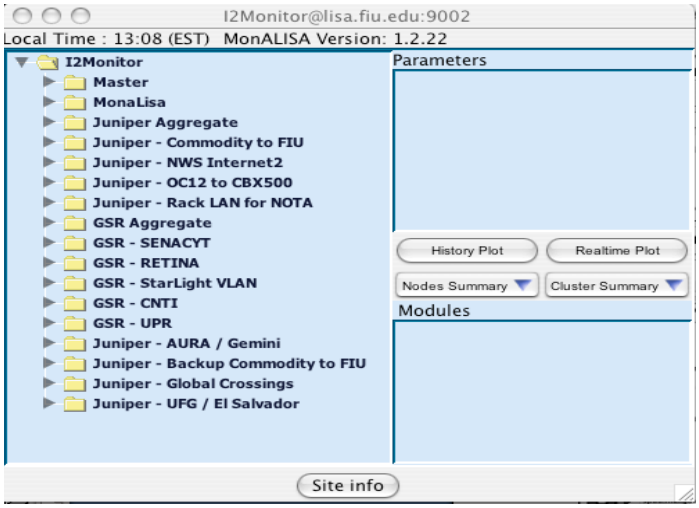

Figure 5– AMPATH Farm specific data

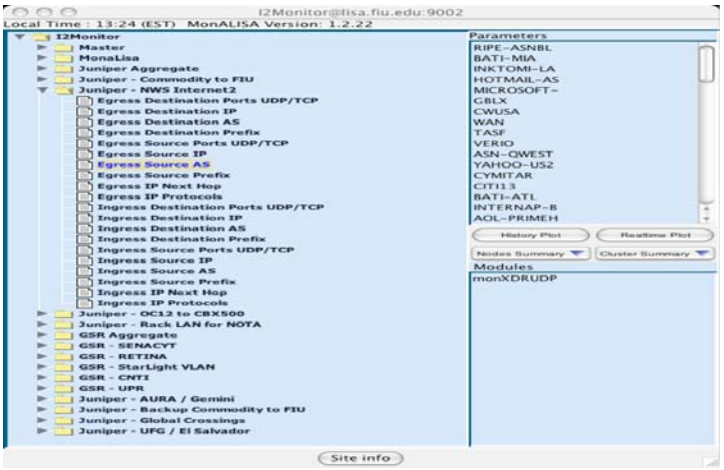

Figure 6 - ASNs traversing AMPATH towards NWS

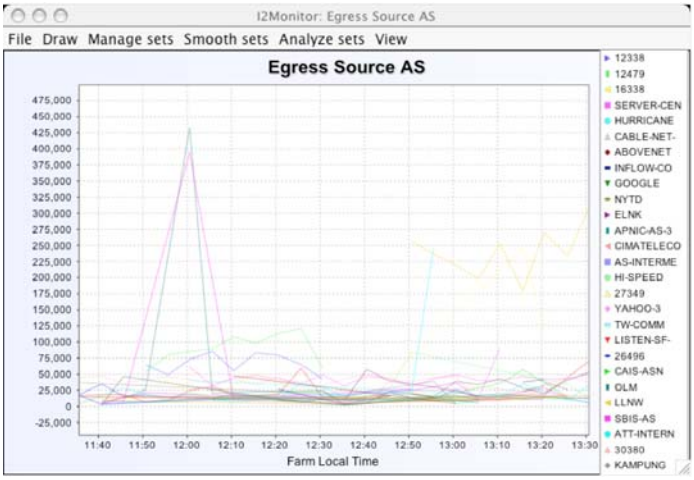

Figure 7 - ASN traffic destined to NWS

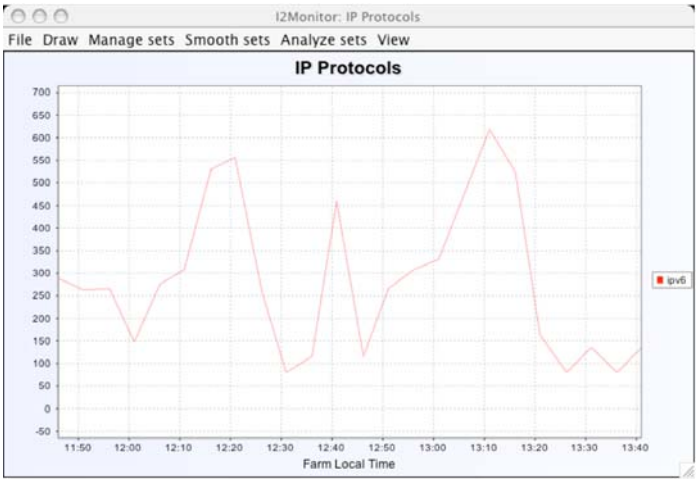

Figure 8 – Two-hour snapshot of IPv6 data traversing AMPATH