DISTRIBUTED NETWORK MONITORING MADE EASY - AN APPLICATION FOR ACCELERATOR CONTROL SYSTEM PROCESS MONITORING*

C.E. Peters, M. Power, ANL, Argonne, IL 60439, USA

Abstract

As the complexity and scope of distributed control systems increase, so does the need for an ever increasing level of automated process monitoring. The goal of this paper is to demonstrate one method whereby the SNMP protocol combined with open-source management tools can be quickly leveraged to gain critical insight into any complex computing system. Specifically, we introduce an automated, fully customizable, web-based remote monitoring solution which has been implemented at the Argonne Tandem Linac Accelerator System (ATLAS). This collection of tools is not limited to only monitoring network infrastructure devices, but also to monitor critical processes running on any remote system. The tools and techniques used are typically available pre-installed or are available via download on several standard operating systems, and in most cases require only a small amount of configuration out of the box. High level logging, level-checking, alarming, notification and reporting is accomplished with the open source network management package OpenNMS, and normally requires a bare minimum of implementation effort by a non-IT user.

BACKGROUND

Ever since the advent of reliable Internet Protocol (IP) communication, the control system at ATLAS has become more heterogeneous [1]. This is in part due to the pervasiveness of the protocol, and because even legacy devices and tools can be networked via Ethernet with the addition of extra hardware. The Simple Network Monitoring Protocol (SNMP) is part of a group of protocols known as the Internet Protocol Suite defined in the early 1980's, and was standardized as RFC2261 (now RFC3411) in the early 1990's [2]. Its application to network device monitoring would then seem to make it an obvious and pervasive choice for control system monitoring. However, the openness and extendibility of the protocol can tend to make implementation appear difficult to a non-networking or information technology oriented user. To date in the author's experience, SNMP is used mainly to monitor network infrastructure devices like managed switches and routers.

The purpose of this work is to demonstrate our SNMP based implementation of control system network monitoring in a way that exemplifies the name 'simple'. There is a large body of previous work related to project specific implementation as full custom applications [3], but few papers are dedicated to the ease of use and utility of SNMP. There are several important pieces of any network monitoring package which are listed below.

* This work was supported by the U.S. Department of Energy, Office of Nuclear Physics, under Contract No. DE-AC02-06CH11357.

- **Operating System**: Any network monitoring implementation is only as good as the operating system chosen to run it. Since SNMP is an open protocol, any operating system on either the client or the server can implement its own SNMP libraries. In the end, the selection of operating system is often based primarily on either 1) User familiarity, and/or 2) Organization rules and norms.
- **SNMP Monitored Device**: Normally in a distributed system there is an SNMP managed device, which itself will consist of a master SNMP agent and several sub-agents (Fig. 1). While normally transparent to the user, the sub-agents are responsible for supplying data to the master, who then handles the task of managing incoming and outgoing SNMP packets. There is normally at least one default sub-agent installed with any SNMP server to respond to preconfigured commands and queries.
- **Centralized Monitoring Database**: This server acts as the single database and coordinator of the monitoring system. It issues all SNMP commands and requests, and stores historical values. Most often comes with graphing and archiving tools, along with several versions of alarming and notification utilities called a Network Monitoring System (NMS).
- **Management Information Base (MIB)**: This is the section where most non-IT users lose their way. A MIB is simply a tree-like definition of the data and its association attributes which a certain agent can handle. It is most often implemented as a text file on the host computer. There are often several default MIBs which come with any installation of SNMP libraries, and therefore the end user often needs to worry little about MIBs until they begin to customize their individual system.

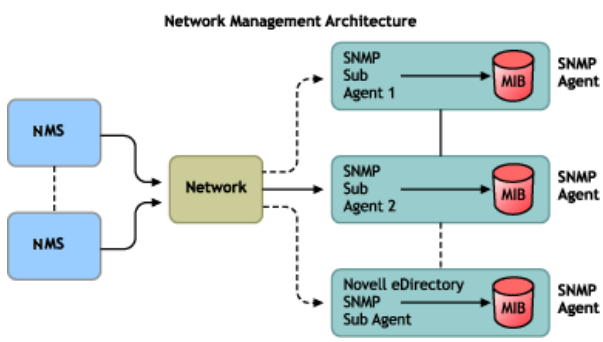

Figure 1: Example of a SNMP architecture [4].

ISBN 978-3-95450-139-7

Example SNMP Transaction

1) Network Management System (client) sends a 'GET' request to a master agent on a remote device for a piece of information defined in the remote MIB.
2) The remote device's Master agent receives SNMP packet, and determines which sub-agent owns the requested data. Calls sub-agent to gather data.
3) Subagent retrieves data and passes back to Master Agent on the device.
4) Master Agent (server) replies back to requestor (client) with the request header and accompanying data.

ACCELERATOR DEVICE SNMP MONITORING AT ATLAS

Process Overview

During the review of network process monitoring at ATLAS, there were two main questions to be answered: 1) What devices support monitoring via SNMP, and 2) What centralized database should be used to coordinate the monitoring process?

Deciding Which Devices can be Monitored

The first step in the implementation of any SNMP enabled monitoring system is to ensure that all the devices to be monitored can support the SNMP protocol itself. As an example of even legacy systems supporting SNMP, ATLAS uses both current and 1980-1990s hardware in its control system design. The legacy portion consists of OpenVMS Alphaservers running a centralized real-time database. This database and its associated running processes are critical to the operation of ATLAS. Any interruption in either the server itself or the running processes could lead to a halt in the experiment. However even though the operating system dates to the early 1980's, OpenVMS has a pre-compiled SNMP library installed in the operating system as a service, along with two default MIBs. OS_MIBS contains information regarding the TCP/IP stack and networking activity, and HR_MIB which provides information about the host system itself. This default installation, while not fully inclusive of all the desired parameters to be monitored, is available with little configuration.

Installation of an SNMP service on Scientific Linux 6 proved to be even easier, and should be similar in operation to any other Linux distribution. The open source package net-snmp is available via the standard repositories, and was installed via the package manager. Both the OpenVMS and Linux versions of the SNMP library are now installed, but often require a few additional configuration changes. In Linux, these settings are stored in *'/etc/snmp/*snmpd.*conf'*, and this file allows the user to set access limits, the system location, the manager's name, and other important parameters which take effect after a service restart.

Using this technique at the time of writing, ATLAS now has 2 OpenVMS Alphaservers and several more Scientific Linux IOCs which are running a configured SNMP daemon and can respond to SNMP packets. In addition, several UPS systems have been added as alarm levels using customized SNMP extension modules, as discussed in the sections that follow.

Selection and Implementation of the Network Monitoring System Software Package

In a distributed SNMP monitoring installation, the software package responsible for collecting, aggregating, and responding to all the individual data points is referred to as the Network Monitoring System (NMS). There are several options to choose from in this category of software packages. In many cases, the NMS can perform much more than simple SNMP transactions. However, the decision should be made early on as to how much support the organization requires, and compare those costs. There are many large enterprise grade systems that are open source and free of any licensing cost, apart from service and support, for example Nagios [5], Zenoss [6].

In the end, ATLAS chose OpenNMS for its installation due to its reputation as a large scale, enterprise network monitoring software. In addition, OpenNMS's installation instructions are very straightforward, and all the instructions are in the form of a large collection of wiki pages what are easy to understand and grouped into logical sections. There is also the option to test out a 'demo' of the software using a sandbox website set up by the company: http://demo.opennms.com/demo, which runs the actual software and not a pre-filmed movie stream. In addition, the package is fully available via package managers, and most of the time does not require any manual compilation of binaries in order to install.

CUSTOM ATLAS SUB-AGENT IMPLEMENTATIONS

The initial results at ATLAS were that the default installation of both the SNMP managed devices and OpenNMS were sufficient to monitor the low level performance of the systems themselves, but did require some additional customization. For example on OpenVMS, data points like number of current TCP connections, total number of processes, and number of users are all available by default. However, due to the large amount of remotely hosted displays which often consume large amounts of memory on legacy systems the controls group desired more information about memory management and specific processes. Also, there is a 'Direct I/O' parameter which can be used to monitor all CAMAC (Computer Automated Measurement and Control) I/O transactions throughout the accelerator.

In order to extend the functionality of the SNMP service, a custom sub-agent was developed. This effort was guided by an HP OpenVMS manual for the SNMP service itself, and several example programs [7]. Having this documentation made it easy for a single developer to

ISBN 978-3-95450-139-7

extend the MIB on these machines to include these critical parameters.

In comparison, an effort was also undertaken to expand some of the MIBs located on several Linux machines located throughout the facility which are connected via USB to Uninterruptable Power Supplies (UPS). Normally, SNMP monitoring of UPS systems is accomplished via a separately installed network gateway on the UPS itself. However, with SNMP sub-agents it is possible to extend the default MIB on the host system and include application specific information from custom scripts and executables. There are several methods to accomplish these customizations, described below.

- **Using the SNMP Configuration:** The snmpd.conf configuration file allows the user to specify several basic customizations. This includes the '*proc*' directive which will monitor the number of processes running with a specific process name. It is also possible, via the 'extend' directive, to specify custom scripts to be executed whenever a certain data point is requested, and the results of that script will be passed back within a standard SNMP transaction.
- **Via the SNMP-Perl Module Extension:** The net-snmp Perl module is available as a separate install. By installing this package, the user can specify a locally installed Perl script which will be executed in a similar manner as the 'extend' directive above. However, there is more low level programmatic access to parameters using the Perl add-on than there might be when using regular Bash scripts.
- **Compiling a Custom Developed Sub-Agent:** This is the most complex method to extend SNMP and requires programming knowledge. The user is also responsible for creating a custom MIB, if desired. The advantage is that the developer has full access to the libraries of that language to interact with the host system.

Table 1 describes the advantages and disadvantages to some of these methods.

Table 1: Summary of SNMP Extension Methods

Method	Advantages	Disadvantages
SNMP Configuration	No compiling, very easy to implement, well documented	Limited to net-snmp tools, libraries and functions
SNMP-Perl Module	No low level programming or compiling required, access to many pre-installed Perl libraries.	Requires knowledge of Perl and does not allow low level access to the machine.
Custom SNMP Agent running as separate process	Complete customized extensions, almost no limits to SNMP data collection	Requires expert knowledge of system and programming experience

EXAMPLE USE CASE & DATA ANALYSIS

The best way to illustrate the power of distributed network monitoring over a wide range of devices in the system is by example. The ability to quickly recall and visualize data makes it sometimes trivial to recognize and implement corrective actions that would otherwise go unnoticed. By using provided utilities such as level checking and automated email notifications, this data can propagate through an organization even before an issue is noticed by the end users.

The legacy OpenVMS systems maintained by ATLAS function as centralized real-time databases and also host several X-window displays throughout the facility. Several times a month, the systems would experience drastic slowdowns and all displays would freeze. The cause was not immediately obvious. However, after implementing SNMP monitoring of many critical parameters of the systems, two revelations emerged. First, as can be seen on the following graph (Fig. 2), the total available free memory was at a constant state of near zero. The green section of the graph is very low, and even hits zero in some cases in 2012. Second, at some point the installed memory pagefile had been disabled from the system. This extra secondary bank of pagefile memory is the orange section of the graph, and re-installing this pagefile (after January 2013) assured the system would have a reserve of memory if the system became highly utilized. Fixing these problems has led to zero downtime due to memory management on the legacy systems. In addition, during times when there are issues, OpenNMS automatically sends email notifications to administrators to notify them of the problem. These level checking and notification utilities are all available by default and easily configurable via the OpenNMS web site.

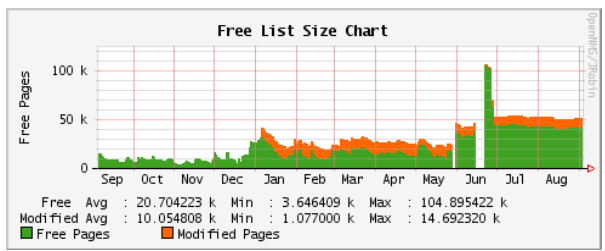

Figure 2: OpenNMS memory chart showing very low free memory (green), and the installation of a secondary page file (orange). Gaps are due to system downtime.

In addition to memory monitoring, Linux based CPU monitoring can be useful to diagnose slowdowns or areas of extra capacity (Fig. 3). Once again, these graphs are available by default and with almost no configuration once net-snmp is installed and OpenNMS is monitoring the system. In addition, OpenNMS comes out of the box with certain default limits for CPU usage and hard disk space, and will notify any users of an issue before it begins to affect the experiment itself. The addition of user-requested automation increased the CPU usage on the below system after July.

ISBN 978-3-95450-139-7

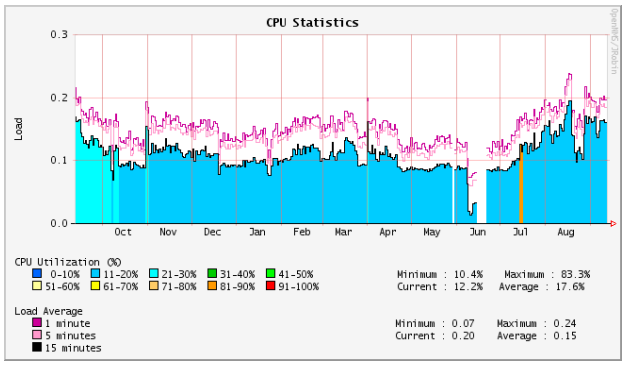

Figure 3: Example of Linux CPU statistics (gaps due to reboots or OpenNMS monitoring restarts).

The net-snmp Perl module was used in order to implement UPS monitoring into many sections of the accelerator. The local utility power is not perfect and sometimes becomes interrupted during an experiment. It was desired to have an alert automatically issued whenever the charge on a battery became low, or the battery itself went out of life and ceased operation. In this case, the default SNMP MIBs did not include the vendor specific UPS monitoring parameters since there are multiple methods to interface with each UPS itself. A custom Perl script was written to query a third program which monitors and displays vendor specific UPS information. The results from this query were then parsed by the Perl module, and passed back to the SNMP agent as a normal response.

Figure 4 shows an example of UPS charge monitoring. The few drop outs represent a battery 'self-test', and when they drop too low can be an indicator of a failing battery.

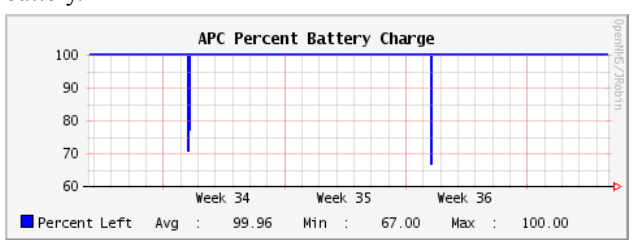

Figure 4: Example of UPS charge monitoring.

The last example of custom process monitoring is using the '*proc*' directive within the snmpd.conf file. This allows the user to specify that the SNMP library should automatically make available a parameter as part of the default MIB indicating how many of the named processes are currently running. This allows ATLAS to monitor specific update and monitoring processes running on every remote node throughout the facility. The failure of one of these processes will change the number currently executing, and generate an email notification once the value falls below a specified limit (Fig. 5).

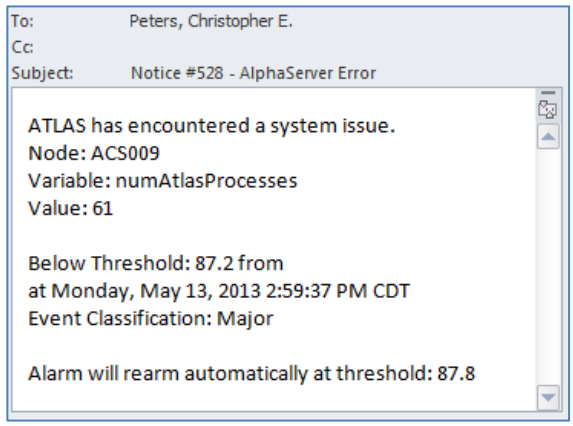

Figure 5: Example of a custom automated email alert.

CONCLUSION

This work to implement distributed monitoring of a heterogeneous control system was undertaken in 2012 and required only two engineers approximately two weeks of effort to first implement. The data which can be collected from this type of automated logging cannot be fully appreciated until it can be presented in a searchable, graphical system along with automated alerts and notifications. The tools used are all open-source and come with complete documentation either online, or via normal library resources.

This same work could be applied to any size control system for minimal cost or effort. Many of the parameters discussed in this work are available after installing exactly 2 packages (net-snmp and OpenNMS) and performing only basic configuration. The added customizations are available in a range from default SNMP directives, to basic scripting languages, all the way up to large and complex standalone sub-modules. The insight gathered as a result of this effort should prove to be extremely valuable to any systems engineer as compared to the time and cost trade-off to implement.

REFERENCES

[1] F. Munson, D. Quock, B. Chapin, and J. Figueroa, "Argonne's ATLAS Control System Upgrade", International Conference on Accelerator and Large Experimental Physics Control Systems, ICALEPCS '99, Trieste, Italy, October 4-8, 1999.

[2] D.O. Savu, B. Martin, A. Al-Shabibi, R. Sjoen, S.M. Batraneanu, S.N. Stancu, "Efficient Network Monitoring for Large Data Acquisition Systems", Proceedings of ICALEPCS 2011, Grenoble, France, 10-14th October 2011.

[3] http://tools.ietf.org/rfcmarkup?doc=3411

[4] http://www.novell.com/documentation/edir873/edir873/data/ag7hbgr.html

[5] http://www.nagios.org/

[6] http://www.zenoss.com/

[7] Hewlet-Packard Company, "HP TCP/IP Services for OpenVMS, SNMP Programming and Reference", Palo Alto, California, January 2005.

ISBN 978-3-95450-139-7