

# Enforcing Model Network Citizenship by Remote Administration

Prasun Gupta  
Institutional Technology  
Ringling School of Art and Design  
Sarasota, FL 34234  
(941) 359-7633  
pgupta@ringling.edu

Mahmoud Pegah  
Institutional Technology  
Ringling School of Art and Design  
Sarasota, FL 34234  
(941) 359-7633  
mpegah@ringling.edu

## ABSTRACT

Higher education institutions have been striving to improve services and keep pace with new technologies. In a Higher education environment, the users utilize the available computing resources 24 hours a day 7 days a week. Although we cannot have the 24 by 7 uptime guaranteed, due to issues like budget constraints, we could certainly reduce the downtime by deploying a cost effective open source network monitoring solution such as Big Brother. Commercially available network management systems are complex and expensive systems with a steep learning curve associated with them. Network management systems are popular in organizations that maintain large and complex networks. Therefore, we enhanced Big Brother solution by extending its service to support SNMP based devices.

## Categories and Subject Descriptors: C.2.3

[Computer-Communication Networks]: Network  
Operations - *Network monitoring, Network Management*

**General Terms:** Management, Measurement, Reliability,  
Performance, Verification.

**Keywords:** Network Monitoring, Network Management,  
System Monitoring, Monitoring, System Reliability, SNMP, Big  
Brother, RMON.

## 1. INTRODUCTION

Ringling School of Art and Design has been striving to improve its services. Therefore Ringling school decided to deploy a system to proactively monitor IT delivered services. Automatic detection of failures would then be possible, reducing downtime and inconvenience to the user base. BB (*Big Brother*) monitoring system was chosen, as it is an open source solution and free for educational institutions.

BB is an elegant solution for monitoring servers, disk spaces on file systems, processes, and cpuload. BB automatically sends notification in the case of a failure via email or pages. BB unfortunately cannot monitor non-server based devices, for instance printers, routers, and switches. Network elements are critical components of an IT infrastructure that were not getting monitored.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

*SIGUCCS'03*, September 21–24, 2003, San Antonio, Texas, USA.  
Copyright 2003 ACM 1-58113-665-X/03/0009...\$5.00.

Organizations such as AT&T, Verizon, and Bellsouth monitor large networks using network management systems. NMS (*Network Management Systems*) are complex and expensive systems with a steep learning curve associated with them. Our goal was to deploy a simple portal like monitoring solution that could monitor network components therefore we decided to extend the functionality of BB to provide the needed network monitoring capability. This paper discusses our experiences with an extension of BB with SNMP for monitoring SNMP based devices.

## 2. BIG BROTHER

BB is an open source system monitoring solution, which polls services to check for availability. The program is written mostly in C, shell scripts, and Perl. It can easily be customized to an individual's needs and preferences. BB is broken down into several components, which makes it easier to manage. The major components are BBNET, BBDISPLAY, BBCLIENT, and BBPAGER. There is a central monitoring station or display server, which receives incoming messages from clients, processes them and makes them available in the form of web pages. The major components of the system include:

**BBNET** - is the server that performs different tests over the network using pings on remote systems and collects the data. Scripts then run on the collected data to generate a dynamic web page that gets displayed on the BBDISPLAY.

**BBCLIENT** - is the client that collects information locally from the system and transmits it to the BBNET.

**BBDISPLAY** - is the host running web server on it and it displays the web pages received from BBNET. BBDISPLAY shows current status of the systems being monitored.

**BBPAGER** - is the host, which takes care of notification/paging task for BB. When correctly configured, BB can send emails, initiate a call to a beeper, and leave alerts for recipients. BB is based on the client/server computational model. BBNET points to a host that runs the BB process called *bbd*. This process is designed to register itself and listen on port 1984 for incoming messages from BBCLIENTS. The BBNET also checks for standard services on hosts configured for monitoring.

BBCLIENT runs a process called *bbbrun* which performs the local system checks on system memory, disk space availability for filesystems, system processes, configured processes, message log checking, and also other configured tests. The client then transmits that information in a fairly simple format to BBNET, which collects the information. BBNET builds web pages out of the collected information and sends these files to the BBDISPLAY server.

BBDISPLAY server places the files in its web directory, which are accessible through the web.

### 3. SNMP

SNMP (*Simple Network Management Protocol*) was developed in 1988, and since then has been widely accepted and implemented in network devices [1]. SNMP is the communication protocol between a NMS and a network device. SNMP is the only real standard protocol used for network management. Hence all network devices have built in support for SNMP. SNMP is extensible by design because of the interface definition structure known as MIB (*Management Information Base*). Generally SNMP devices are almost guaranteed to support IETF (*Internet Engineering Task Force*) based MIBs. Proprietary MIB extensions may or may not be incorporated in the devices. SNMP is composed of *manager/agent* based interaction paradigm where NMS is the manager and network device is the agent. The network administrator uses manager to perform network management functions. Agents are the protocol units that reside in the actual device being managed. Bridges, Switches, Routers or network servers are examples of managed devices that contain managed objects.

These managed objects might be hardware, configuration parameters, performance statistics, etc, that directly relate to the current operation of the device in question. These objects are arranged in what is known as a virtual information database, called a MIB. SNMP allows managers and agents to communicate for the purpose of accessing these objects. The high level model of network management architecture is shown in figure 1.

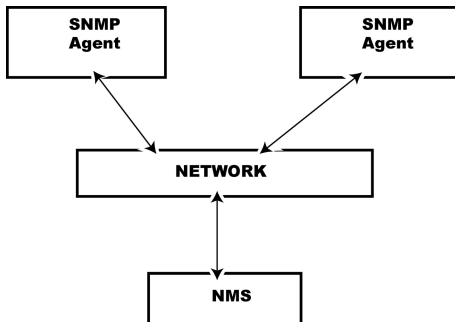


Figure 1. Network Management Architecture

In the *manager/agent* paradigm for network management, managed network objects must be logically accessible. Logical accessibility means that management information must be stored somewhere within and therefore that the information must be retrievable and modifiable. SNMP actually performs the retrieval and modification.

The SMI (*Structure of Management Information*), which is given in RFC 1155, is based on the OSI SMI model proposed in Draft proposal 2684 [1]. The SMI organizes, names, and describes information so that logical accesses can occur. The SMI states that each managed object must have a name, syntax, and an encoding scheme. The name, an OID (*object identifier*), uniquely identifies the object. The SMI defines the data type, such as an integer or a string of octets. The encoding scheme describes how the information associated with the managed objects is serialized for transmission between machines. The encoding scheme used for SNMP is the BER (*Basic Encoding Rules*). MIBs are represented in an OSI standard called ASN.1 (*Abstract Syntax Notation One*). Each object whether it's a device or a characteristic of a device, must have a name by which it can be uniquely identified. That name is the OID. It is written as a sequence of integers, separated by periods. For

instance, the sequence 1.3.6.1.2.1.1.1.0 specifies the system description within the system group, of the *mgmt* subtree.

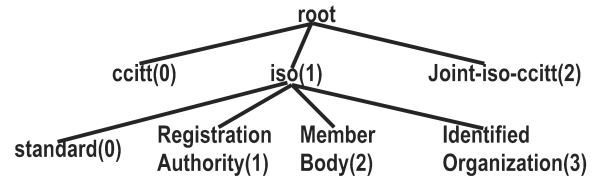


Figure 2. Root of the MIB Tree

Figure 2 shows the top cross section of the global MIB tree. MIBs are files that define the communication interface between agent and manager. The MIB as a data structure is represented as a tree, whose nodes are addressed as OIDs. IETF ensures that there is a globally unique MIB tree for smooth operation of SNMP, it also maintains and publishes all the MIBs. Theoretically all the MIBs in the world combine to form one large unique and unambiguous tree. IETF also maintains proprietary MIBs. For instance a company like Cisco can pay required fees to IETF and get a node from a list of nodes reserved for private enterprises. The root OID for the private enterprises subtree is:

*iso(1).org(3).dod(6).internet(1).private(4).enterprises(1)* and Cisco OID is:

*iso(1).org(3).dod(6).internet(1).private(4).enterprises(1).cisco(9).*

All of the Cisco's proprietary extensions are defined under the Cisco OID.

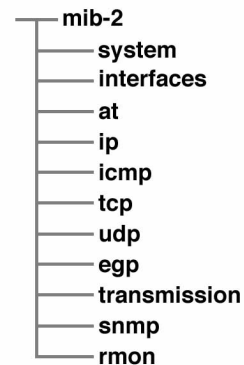


Figure 3. Tree hierarchy of MIB-II (RFC 1213)

Figure 3 shows MIB-II and it was created, by taking a screen shot in *Mgsoft MIB-Browser* software. The MIB-II (Figure 3) subtree is the actual mandatory Internet MIB that all agents must implement (currently MIB-II RFC 1156 is the only subtree of *mgmt* node) [1]. Enterprise subtree (of private) are MIBs of proprietary objects and are not mandatory for agents to implement. (subtree registered with Internet Assigned Numbers Authority) For instance: Cisco router OID: 1.3.6.1.4.1.9.1.1 [1]. An agent is not allowed to have a partial implementation of a group.

#### 3.1 SNMP Protocol

SNMP is based on the *manager/agent* model. Most of the processing power and the data storage resides on the management system such as *HP Openview* [3], while a complementary subset of those functions resides in the managed system such as router. SNMP is a very simple protocol with a limited set of management commands and responses. The management system issues *Get*,

*GetNext* and *Set* messages to retrieve single or multiple object variables or to establish the value of a single variable. The managed agent sends a response message to complete the *Get*, *GetNext* or *Set*. The managed agent sends an asynchronous event notification called a *trap* to the management system to identify the occurrence of certain conditions like on a reboot of a router.

### 3.2 RMON

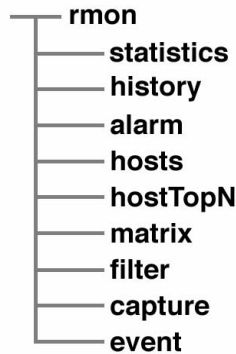


Figure 4. RMON Groups

Figure 4 shows a screen shot of RMON groups taken in *Mgsoft MIB-Browser* software. In 1991, the Remote Monitoring (RMON) protocol was created to augment SNMP in networks [12]. RMON was developed to add the capability of remotely monitoring components of a LAN (*Local Area Network*). RMON added more intelligence to the agents, which monitored its interfaces, stored the filtered data and also provided information to NMS on getting queried. RMON reduced the polling that had previously hampered the use of SNMP on larger networks and extended the range of information that can be sent back to the SNMP manager. By allowing the network manager to set thresholds, RMON also enables probes to measure network performance. When the threshold for acceptable network behavior is exceeded, the RMON probe alerts the SNMP management station to the problem. The RMON protocol reports statistics at layer two (data link) in the OSI reference model. However new extensions allow reporting of information at layer three (network layer) in the OSI reference model is possible. Approved in 1996, RMON II enables RMON probes to completely provide information at OSI layer three [12].

## 4. SNMP EXTENSION of BB

Once the core installation of BB is deployed, the next step is to add SNMP support. Packages needed to be installed in order to extend BB to monitor SNMP devices. This included Openssl (from <http://www.sunfreeware.com>) and SNMP library net-snmp (from <http://www.net-snmp.org>). Additional scripts needed were *bb-snmp.pl* and *BigBrother.pm* (from <http://www.deadcat.net>). Place *bb-snmp.pl* in the \$BBHOME/ext directory and *BigBrother.pm* in the \$BBHOME directory. Follow installation instructions within the scripts to configure them in BB. After successful installing of the packages and scripts, one should be able to setup SNMP devices with connectivity tests. We will now discuss the enhancements made by us to the existing framework.

### 4.1 Our Extensions to BB-SNMP Infrastructure

We enhanced existing monitoring functionality of BB by adding monitoring of network elements and printers. In order to add the monitoring functionality we created two Perl scripts *net.pl* and

*prn.pl*. In order to use these scripts place them in the \$BBHOME/ext directory. We also followed standard steps for enabling external scripts inside the BB framework. There are a few additional steps to get the scripts to work, which are detailed within the scripts.

### 4.2 Monitoring Network Elements

We created a script called *net.pl* that polls the configured network elements. The test would retrieve the results and display it in the BBDISPLAY web page under the heading *net*. One of our main goals was to build a platform independent solution because of heterogeneity in Ringling School's network and also in typical campus networks. Hence we only used standard MIBs for interfacing to SNMP devices. We built our solution using MIB-II (RFC-1213) and RMON. Ringling school's network equipment was comprised of components with support for only four RMON groups *Statistics*, *History*, *Alarm* and *Event*. Thus we limited the scope of our monitoring to these four groups. We will now briefly describe the steps taken in configuring RMON on Cisco switches.

Log on to the switch. Change to the configuration mode. Select an interface and enable RMON to begin gathering history with number of buckets set to one. Similarly enable history on other interfaces. This will enable RMON to run on the agent and also to start collecting the needed statistics. One can also setup alarms, which trigger on configured conditions. These alarms can be associated with user defined events. The events are configurable to log or send a trap. In our configuration, we just enabled the events to log. Cisco administration manuals provide detailed information on SNMP/RMON setup [9].

MIB-II [4] was used in our monitoring scheme. Here are the variables used from the MIB:

- System.sysuptime
- Interfaces.ifTable.ifEntry.ifOperStatus
- Interfaces.ifTable.ifEntry.ifInErrors
- Interfaces.ifTable.ifEntry.ifInUnknownProtos
- Interfaces.ifTable.ifEntry.ifOutErrors
- Ip.ipInHdrErrors
- Ip.ipInAddrErrors

RMON MIB was used in our monitoring. History group provided the needed parameters matching our requirements for LAN monitoring. The variables are listed below:

- EtherHistoryIntervalStart
- EtherHistoryDropEvents
- EtherHistoryOctets
- EtherHistoryPkts
- EtherHistoryBroadcastPkts
- EtherHistoryMulticastPkts
- EtherHistoryCRCAlignErrors
- EtherHistoryUndersizePkts
- EtherHistoryOversizePkts
- EtherHistoryFragments
- EtherHistoryJabbers
- EtherHistoryCollisions
- EtherHistoryUtilization

The *net.pl* script polls the *Event Log* table specified in RMON, which logs events, and serves as an alarm log. Unfortunately events logged in queues do not get deleted even after the device conditions reach normalcy. The logs do not represent the current state of the device. BB has three states of alarm *normal (green)*, *warning (yellow)*, and *critical (red)*, mapping polled information to BB's alarm states is a difficult problem because there is no severity field in the event structure.

We improvised by mapping BB's alarm states to an event's occurrence in time. The Alarm state for events logged within an hour was red. It would change to yellow after an hour and change to green after a day. In this way, the network administrator would be alerted by the change in status. Our solution was generic and effective in alerting the network administrator in case of a problem. Therefore network administrator should be careful in configuring network elements to only generate minimal and necessary events.

### 4.3 Monitoring Printers

Printers are also SNMP based devices and therefore were also integrated into the system. There is a standard MIB for printers called the *printer.mib*. We used this MIB for interfacing to printers running SNMP agents. We started monitoring the alarms table in the MIB. We created a script called *prn.pl* that polls the configured printers. The *prn.pl* script retrieves results and display it in BBDISPLAY web page under the heading *prn*.

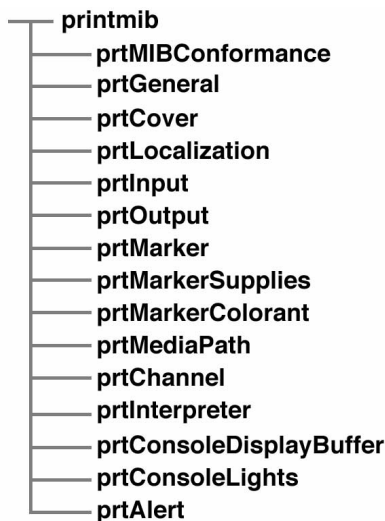


Figure 5. The Standard Printer MIB

Figure 5 shows Printer.mib groups, created by taking a screen shot in *Mgsoft MIB-Browser* software. We only used the *printMib.prtAlert* group for monitoring the printers. Listed are the MIB variables used for monitoring the printers. *Printer-MIB* [7] provide detailed information on the MIB variables. The MIB variables we utilized are as follows:

- PrtAlertSeverityLevel
- prtAlertTrainingLevels
- prtAlertGroup
- PrtAlertGroupIndex
- prtAlertLocation
- prtAlertCode Ether
- prtAlertDescription

## 5. FURTHER WORK

Additional improvements can be made to the current system which will make a significant difference to the system's overall viability, including:

- Integration of SNMP trap receiver that receives traps sent from SNMP devices triggered on specified error conditions.
- Inclusion of other classes of SNMP devices such as SNMP enabled power supply units, data center HVAC (*heating ventilating and air conditioning*) units, tape library systems, etc.
- Utilization of the added features of the more recently developed SMON (*switch monitoring*) published MIB, an improvement over the RMON MIB used in our scheme, which takes advantage of additional features for monitoring switched networks.

## 6. ACKNOWLEDGMENTS

We would like to acknowledge the online resources maintained by BB community, which helped us tremendously. To name a few of the resources *mailarchive*, *mailinglist* accessible from <http://www.bb4.com> and user contributed software at <http://www.deadcat.net>. We would also like to acknowledge the initial work done by Sean MacGuire (author of BB) and Craig Cook (author of *cisco\_cpu.pl* [2]) in the original integration of SNMP with BB.

## 7. REFERENCES

- [1] Cohen, Y., *SNMP - Simple Network Management Protocol*, <http://www2.rad.com/networks/1995/snmp/snmp.htm>, 1995.
- [2] Cook, C., *cisco\_cpu.pl*, <http://www.deadcat.net>, 2003.
- [3] Hewlett-Packard, *HP OpenView*, <http://www.openview.hp.com/>, 2003.
- [4] McCloghrie, K. and Rose, M., *Management Information Base for Network Management of TCP/IP-based internets: MIB-II*, <http://www.ietf.org>, 1991.
- [5] Mohr, J., *Big Brother*, <http://www.linux-tutorial.info/cgi-bin/display.pl?177&0&0&3>, 2002.
- [6] Perkins, D. T., *RMON: Remote Monitoring of SNMP-Managed LANs*: Prentice Hall, 1998.
- [7] Smith, R., Wright, F., Hastings, T., Zilles, S., and Gyllenskog, J., *Printer MIB*, <http://www.ietf.org>, 1995.
- [8] Stallings, W., *Data and Computer Communication*, Fourth Edition ed: MacMillan, 1994.
- [9] Systems, C., *RMON*, [http://www.cisco.com/en/US/tech/tk648/tk362/tk560/tech\\_protocol\\_home.html](http://www.cisco.com/en/US/tech/tk648/tk362/tk560/tech_protocol_home.html), 2003.
- [10] Trimmer, M., *Using Big Brother to Verify System Availability*, [http://www.sans.org/rr/sysadmin/big\\_brother.php](http://www.sans.org/rr/sysadmin/big_brother.php), 2000.
- [11] Waldbusser, S., *Remote Network Monitoring Management Information Base*, <http://www.ietf.org>, 2000.
- [12] Wong, E., *Network Monitoring Fundamentals and Standards*, [http://www.cis.ohio-state.edu/~jain/cis788-97/net\\_monitoring/index.htm](http://www.cis.ohio-state.edu/~jain/cis788-97/net_monitoring/index.htm), 1997.
- [13] Wright, D., Manros, C., and Zilles, S., *Major Companies Players Unite to Deliver Internet Printing Standard*, <http://www.pwg.org/ipp/releases/rel0497.html>, 1997.