Cisco − Baseline Process: Best Practices White Paper

Table of Contents

Baseline Process: Best Practices White Paper

Introduction

This document describes baselining concepts and procedures for highly available networks. It includes critical success factors for network baselining and thresholding to help evaluate success. It also provides significant detail for baseline and threshold processes and implementation that follow best practice guidelines identified by Cisco's High Availability Services (HAS) team.

This document takes you step−by−step through the process of baselining. Some current network management system (NMS) products can help automate this process, however, the baselining process remains the same whether you use automated or manual tools. If you use these NMS products, you must adjust the default threshold settings for your unique network environment. It is important to have a process to intelligently choose those thresholds so that they are meaningful and correct.

Why a Baseline?

A baseline process helps you to identify and properly plan for critical resource limitation issues in the network. These issues can be described as control plane resources or data plane resources. Control plane resources are unique to the specific platform and modules within the device and can be impacted by a number of issues including:

- Data utilization
- Features enabled
- Network design

Control plane resources include parameters such as:

- CPU utilization
- Memory utilization

- Buffer utilization

Data plane resources are impacted only by the type and quantity of traffic and include link utilization and backplane utilization. By baselining resource utilization for critical areas, you can avoid serious performance issues, or worse, a network meltdown.

With the introduction of latency−sensitive applications such as voice and video, baselining is now more important than ever. Traditional Transmission Control Protocol/Internet Protocol (TCP/IP) applications are forgiving and allow for a certain amount of delay. Voice and video are User Datagram Protocol (UDP) based and do not allow for retransmissions or network congestion.

Due to the new mix of applications, baselining helps you to understand both control plane and data plane resource utilization issues and to proactively plan for changes and upgrades to ensure continued success.

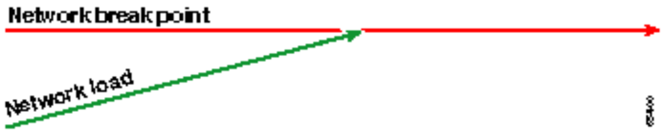
Data networks have been around for many years. Until recently, keeping the networks running has been a fairly forgiving process, with some margin for error. With the increasing acceptance of latency−sensitive applications such as Voice over IP (VoIP), the job of running the network is becoming harder and requires more precision. In order to be more precise and to give a network administrator a solid foundation upon which to manage the network, it is important to have some idea of how the network is running. To do this, you must go through a process called a baseline.

What is a Baseline?

A baseline is a process for studying the network at regular intervals to ensure that the network is working as designed. It is more than a single report detailing the health of the network at a certain point in time. By following the baseline process, you can obtain the following information:

- Gain valuable information on the health of the hardware and software
- Determine the current utilization of network resources
- Make accurate decisions about network alarm thresholds
- Identify current network problems
- Predict future problems

Another way of looking at the baseline is illustrated in the following diagram.



The red line, the network break point, is the point at which the network will break, which is determined through the knowledge of how the hardware and software perform. The green line, the network load, is the natural progression of load on the network as new applications are added, and other such factors.

The purpose of a baseline is to determine:

- Where your network is on the green line
- How fast the network load is increasing
- Hopefully predict at what point in time the two will intersect

By performing a baseline on a regular basis, you can find out the current state *and* extrapolate when failures will occur and prepare for them in advance. This also helps you to make more informed decisions about when, where, and how to spend budget money on network upgrades.

Baseline Objective

The objective of a baseline is to:

1. Determine the current status of network devices
2. Compare that status to standard performance guidelines
3. Set thresholds to alert you when the status exceeds those guidelines

Due to the large amount of data and the amount of time it takes to analyze the data, you must first limit the scope of a baseline to make it easier to learn the process. The most logical, and at times the most beneficial, place to start is with the core of the network. This part of the network is usually the smallest and requires the most stability.

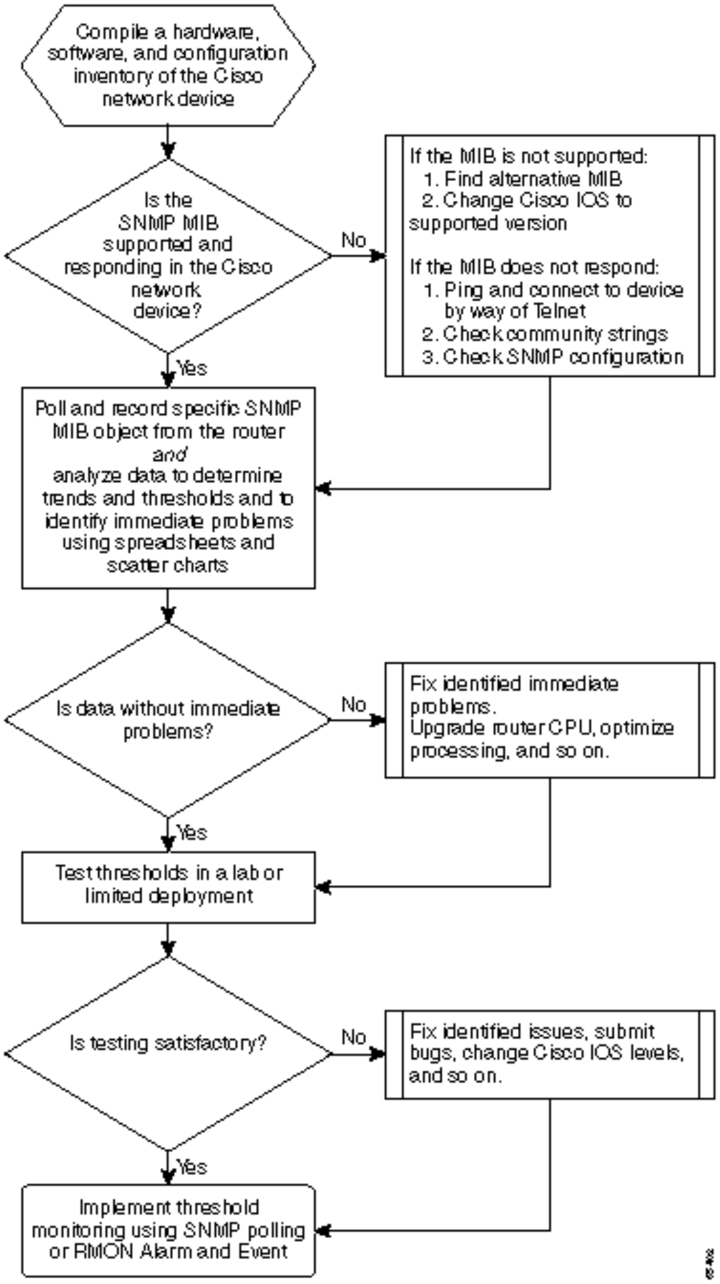
For the sake of simplicity, this document explains how to baseline one very important Simple Network Management Protocol Management Information Base (SNMP MIB): cpmCPUTotal5min. cpmCPUTotal5min is the five−minute decaying average of a Cisco router's central processing unit (CPU), and is a control plane performance indicator. The baseline will be performed on a Cisco 7000 series router.

Once you have learned the process, you can apply it to any data available in the vast SNMP database which is available in most Cisco devices, such as:

- Integrated Services Digital Network (ISDN) usage
- Asynchronous Transfer Mode (ATM) cell loss
- Free system memory

Core Baseline Flowchart

The following flow chart shows the basic steps of the core baseline process. While products and tools are available to perform some of these steps for you, they tend to have gaps in flexibility or ease of use. Even if you plan to use network management system (NMS) tools to perform baselining, this is still a good exercise in studying the process and understanding how your network really works. This process may also take some of the mystery out of how some NMS tools work since most tools essentially do the same things.


Compile a hardware, software, and configuration inventory of the Cisco network device
Is the SNMP MIB supported and responding in the Cisco network device?
No
If the MIB is not supported:
1. Find alternative MIB
2. Change Cisco IOS to supported version
If the MIB does not respond:
1. Ping and connect to device by way of Telnet
2. Check community strings
3. Check SNMP configuration
Yes
Poll and record specific SNMP MIB object from the router *and* analyze data to determine trends and thresholds and to identify immediate problems using spreadsheets and scatter charts
Is data without immediate problems?
No
Fix identified immediate problems. Upgrade router CPU, optimize processing, and so on.
Yes
Test thresholds in a lab or limited deployment
Is testing satisfactory?
No
Fix identified issues, submit bugs, change Cisco IOS levels, and so on.
Yes
Implement threshold monitoring using SNMP polling or RMON Alarm and Event

Baseline Procedure

Step 1: Compile a Hardware, Software, and Configuration Inventory

It is extremely important that you compile an inventory of hardware, software, and configuration for several reasons. First, Cisco SNMP MIBs are, in some cases, specific to the Cisco IOS release that you are running. Some MIB objects are replaced with new ones or are, at times, completely eliminated. The hardware inventory is most important after the data is collected since the thresholds you need to set after the initial baseline are often based on the type of CPU, amount of memory, and so on, on the Cisco devices. The configuration inventory is also important to make sure that you know the current configurations: You may want to change device configurations after your baseline to tune buffers, and so on.

The most efficient way to do this part of the baseline for a Cisco network is with CiscoWorks2000 Resource Manager Essentials (Essentials). If this software is installed correctly in the network, Essentials should have the current inventories of all devices in its database. You simply need to look at the inventories to see if there are any issues.

The following table is an example of a Cisco Router Class software inventory report exported from Essentials, and then edited in Microsoft Excel. From this inventory, notice that you have to use SNMP MIB data and Object Identifiers (OIDs) found in the 12.0x and 12.1x Cisco IOS releases.

Device Name	Router Type	Version	Software Version
field−2500a.embu−mlab.cisco.com	Cisco 2511	M	12.1(1)
qdm−7200.embu−mlab.cisco.com	Cisco 7204	B	12.1(1)E
voip−3640.embu−mlab.cisco.com	Cisco 3640	0x00	12.0(3c)
wan−1700a.embu−mlab.cisco.com	Cisco 1720	0x101	12.1(4)
wan−2500a.embu−mlab.cisco.com	Cisco 2514	L	12.0(1)
wan−3600a.embu−mlab.cisco.com	Cisco 3640	0x00	12.1(3)
wan−7200a.embu−mlab.cisco.com	Cisco 7204	B	12.1(1)E
172.16.71.80	Cisco 7204	B	12.0(5T)

If Essentials is not installed in the network, you can use the UNIX command line tool **snmpwalk** from a UNIX workstation to find the IOS version. This is shown in the following example. If you're not sure how this command works, type **man snmpwalk** at the UNIX prompt for more information. The IOS version will be important in Step 2 when you begin choosing which MIB OIDs to baseline, since the MIB objects are IOS dependent. Also notice that by knowing the router type, you can later make determinations as to what the thresholds should be for CPU, buffers, and so on.

```
nsahpov6% snmpwalk −v1 −c private 172.16.71.80 system
system.sysDescr.0 : DISPLAY STRING− (ascii): Cisco Internetwork Operating System Software
IOS (tm) 7200 Software (C7200−JS−M), Version 12.0(5)T, RELEASE SOFTWARE (fc1)
Copyright (c) 1986−2001 by cisco Systems, Inc.
Compiled Fri 23−Jul−2001 23:02 by kpma
system.sysObjectID.0 : OBJECT IDENTIFIER:
.iso.org.dod.internet.private.enterprises.cisco.ciscoProducts.cisco7204
```

Step 2: Verify that the SNMP MIB is Supported in the Router

Now that you have an inventory of the device you want to poll for your baseline, you can begin to choose the specific OIDs you want to poll. It saves a lot of frustration if you verify, ahead of time, that the data you want is actually there. The cpmCPUTotal5min MIB object is in the CISCO−PROCESS−MIB.

To find the OID you want to poll, you need a conversion table which is available on Cisco's CCO web site. To access this web site from a web browser, go to the Cisco MIBs page, and click the OIDs link.

To access this web site from an FTP server, type **ftp://ftp.cisco.com/pub/mibs/oid/**. From this site, you can download the specific MIB that has been decoded and sorted by OID numbers.

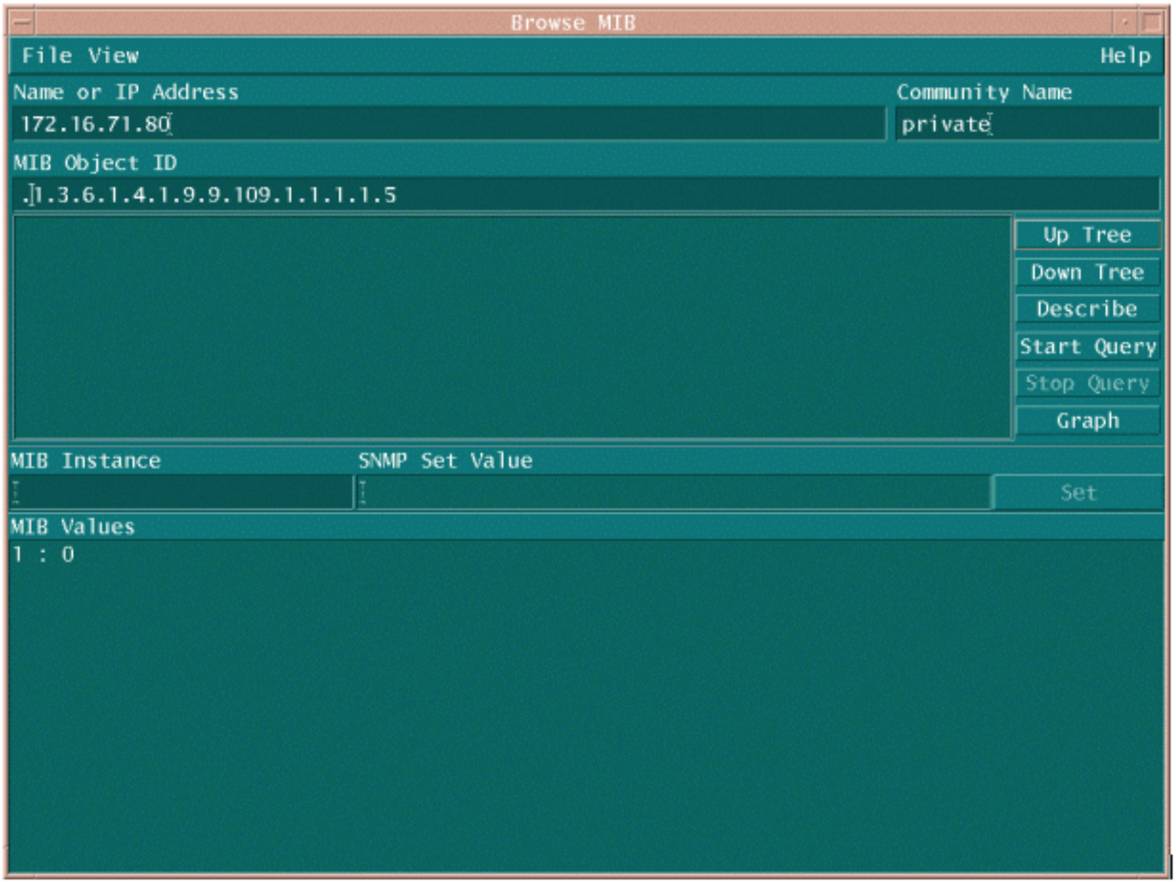
The following example is extracted from the CISCO−PROCESS−MIB.oid table. This example shows that the OID for the **cpmCPUTotal5min** MIB is .1.3.6.1.4.1.9.9.109.1.1.1.1.5.

NOTE: Don't forget to add a "." to the beginning of the OID or you'll get an error when you try to poll it. You also need to add a ".1" to the end of the OID to instantiate it. This tells the device the instance of the OID you are looking for. In some cases, OIDs have more than one instance of a particular type of data, such as when a router has multiple CPUs.

```
ftp://ftp.cisco.com/pub/mibs/oid/CISCO−PROCESS−MIB.oid
### THIS FILE WAS GENERATED BY MIB2SCHEMA
"org" "1.3"
"dod" "1.3.6"
"internet" "1.3.6.1"
"directory" "1.3.6.1.1"
"mgmt" "1.3.6.1.2"
"experimental" "1.3.6.1.3"
"private" "1.3.6.1.4"
"enterprises" "1.3.6.1.4.1"
"cisco" "1.3.6.1.4.1.9"
"ciscoMgmt" "1.3.6.1.4.1.9.9"
"ciscoProcessMIB" "1.3.6.1.4.1.9.9.109"
"ciscoProcessMIBObjects" "1.3.6.1.4.1.9.9.109.1"
"ciscoProcessMIBNotifications" "1.3.6.1.4.1.9.9.109.2"
"ciscoProcessMIBConformance" "1.3.6.1.4.1.9.9.109.3"
"cpmCPU" "1.3.6.1.4.1.9.9.109.1.1"
"cpmProcess" "1.3.6.1.4.1.9.9.109.1.2"
"cpmCPUTotalTable" "1.3.6.1.4.1.9.9.109.1.1.1"
"cpmCPUTotalEntry" "1.3.6.1.4.1.9.9.109.1.1.1.1"
"cpmCPUTotalIndex" "1.3.6.1.4.1.9.9.109.1.1.1.1.1"
"cpmCPUTotalPhysicalIndex" "1.3.6.1.4.1.9.9.109.1.1.1.1.2"
"cpmCPUTotal5sec" "1.3.6.1.4.1.9.9.109.1.1.1.1.3"
"cpmCPUTotal1min" "1.3.6.1.4.1.9.9.109.1.1.1.1.4"
"cpmCPUTotal5min" "1.3.6.1.4.1.9.9.109.1.1.1.1.5"
```

There are two common ways to poll the MIB OID to make sure it's available and functioning. It is a good idea to do this before you start the bulk data collection so that you don't waste time polling something that is not there and end up with an empty database. One way to do this is to use a MIB walker from your NMS platform such as HP OpenView Network Node Manager (NNM), or CiscoWorks Windows, and enter the OID you want to check.

The following is an example from HP OpenView SNMP MIB walker.



Name or IP Address: 172.16.71.80

Community Name: private

MIB Object ID: .1.3.6.1.4.1.9.9.109.1.1.1.1.5

MIB Instance

SNMP Set Value

MIB Values

1 : 0

Another easy way to poll the MIB OID is to use the UNIX command **snmpwalk** as shown in the following example.

```
nsahpov6% cd /opt/OV/bin
nsahpov6% snmpwalk −v1 −c private 172.16.71.80
.1.3.6.1.4.1.9.9.109.1.1.1.1.5.1

cisco.ciscoMgmt.ciscoProcessMIB.ciscoProcessMIBObjects.cpmCPU.cpmCPUTotalTable.cpmCPUTotal
: Gauge32: 0
```

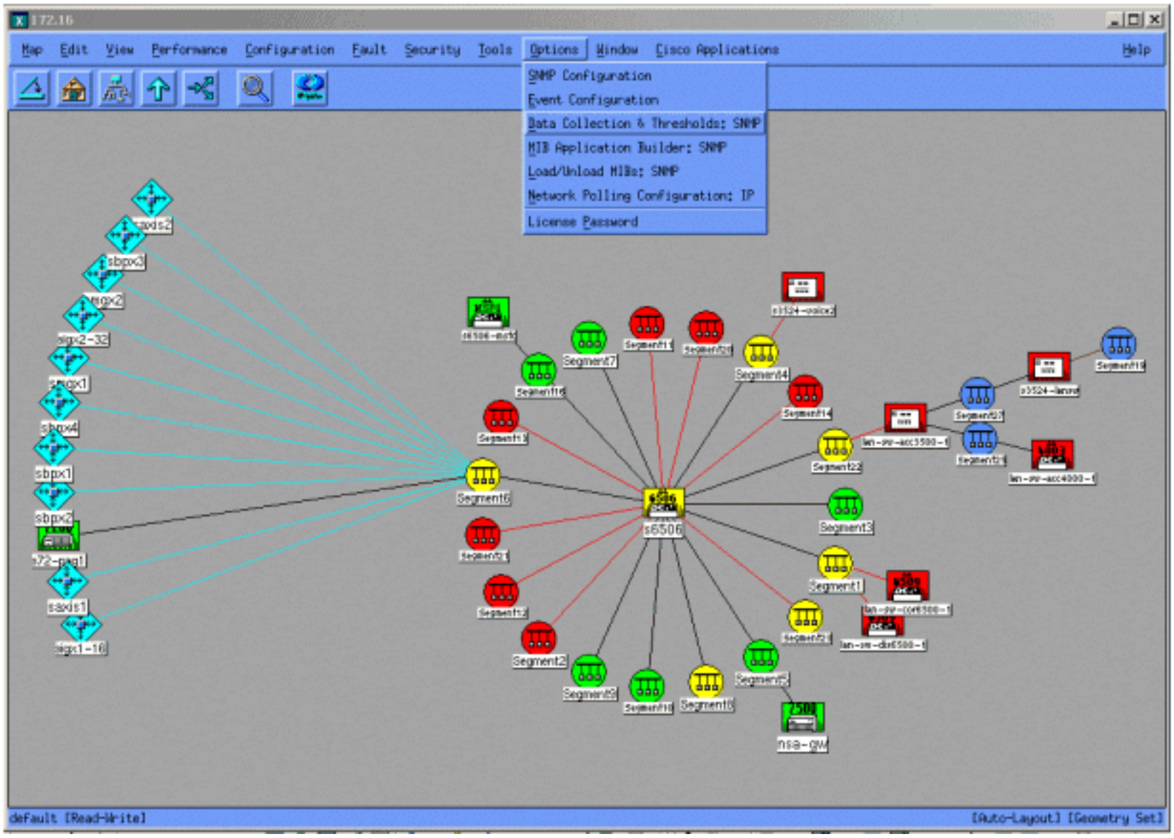
In both examples, the MIB returned a value of 0, meaning that for that polling cycle the CPU averaged 0 percent utilization. If you have difficulty getting the device to respond with the correct data, try pinging the device and accessing the device by way of Telnet. If you still have a problem, check the SNMP configuration and the SNMP community strings. You may need to find an alternative MIB or another version of IOS to make this work.

Step 3: Poll and Record Specific SNMP MIB Object from the Router

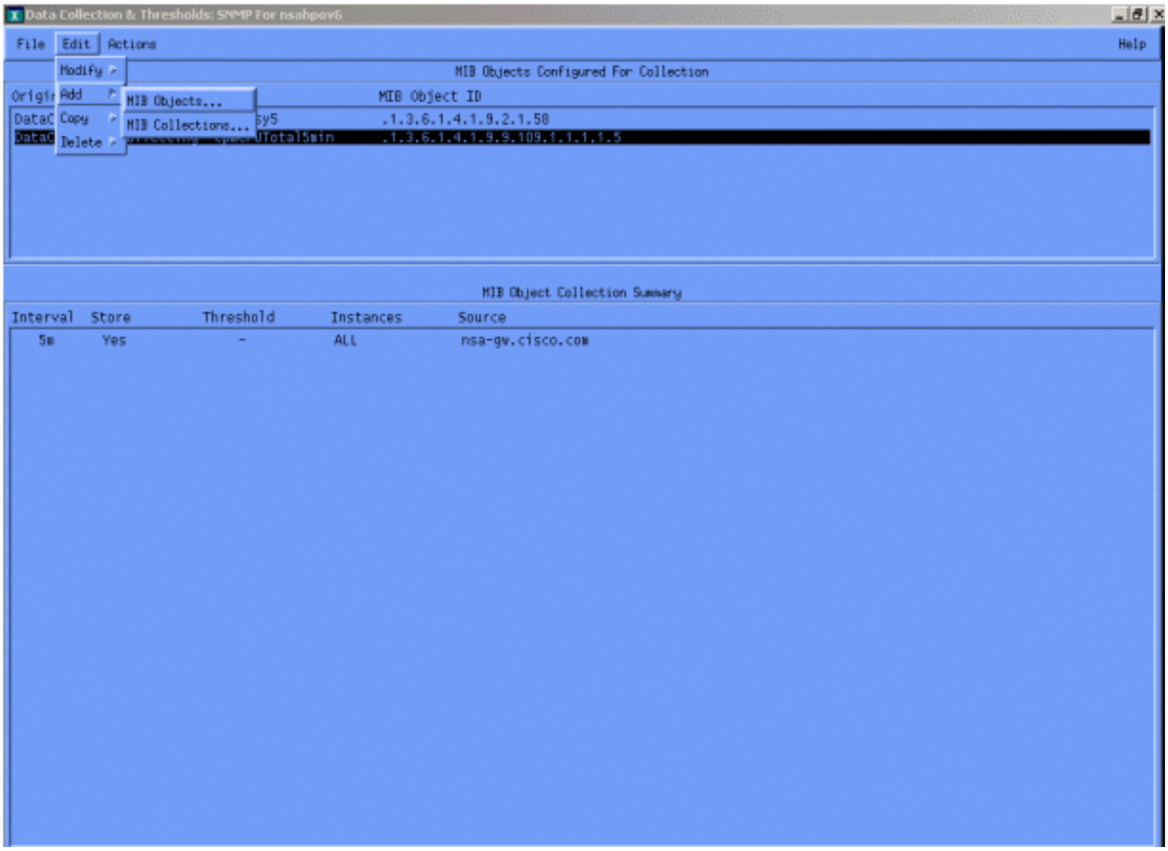
There are several ways to poll MIB objects and record the output. Off−the−shelf products, shareware products, scripts, and vendor tools are available. All front−end tools use the SNMP **get** process to obtain the information. The main differences are in the flexibility of the configuration and the way in which the data is recorded in a database. Again, look at the processor MIB to see how these various methods work.

Now that you know the OID is supported in the router, you need to decide how often to poll it and how to record it. Cisco recommends that the CPU MIB be polled at five−minute intervals. A lower interval would increase the load on the network or device, and since the MIB value is a five−minute average anyway it would not be useful to poll it more often than the averaged value. It is also generally recommended that baseline polling have at least a two−week period so that you can analyze at least two weekly business cycles on the network.

The following screens show you how to do add MIB objects with HP OpenView Network Node Manager version 6.1. From the main screen, select **Options** > **Data Collection & Thresholds**.



Then select **Edit** > **Add** > **MIB Objects**.



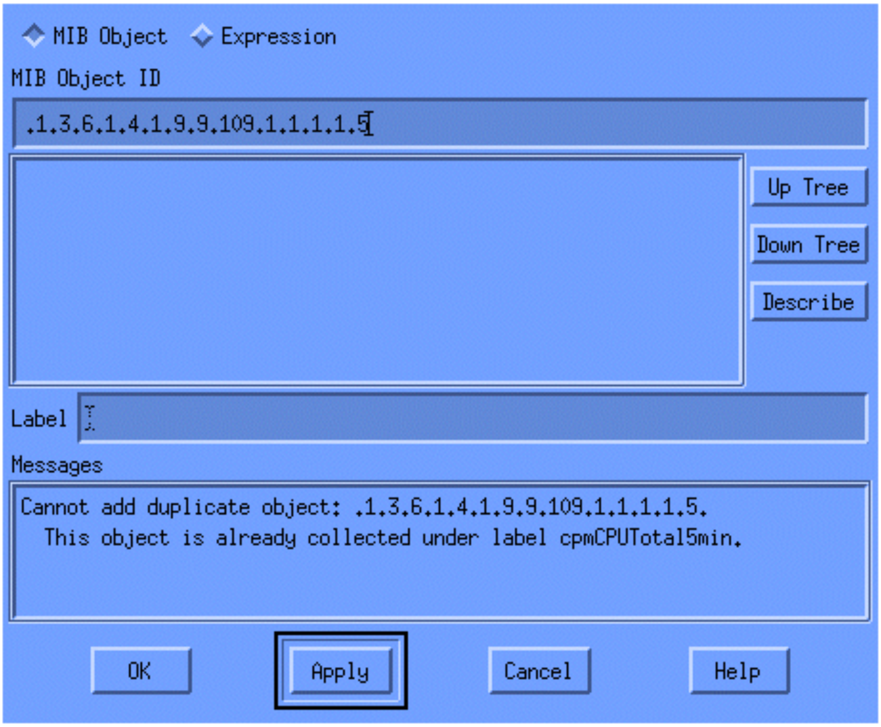
MIB Objects Configured For Collection

```
Origi              MIB Object ID
DataC        sy5       .1.3.6.1.4.1.9.2.1.58
DataC        UTotal5min  .1.3.6.1.4.1.9.9.109.1.1.1.1.5
```

MIB Object Collection Summary

Interval	Store	Threshold	Instances	Source
5m	Yes	-	ALL	nsa-gw.cisco.com

From the menu, add the OID string and click **Apply**. You have now entered the MIB object into the HP OpenView platform so that it can be polled.



MIB Object ID: .1.3.6.1.4.1.9.9.109.1.1.1.1.5

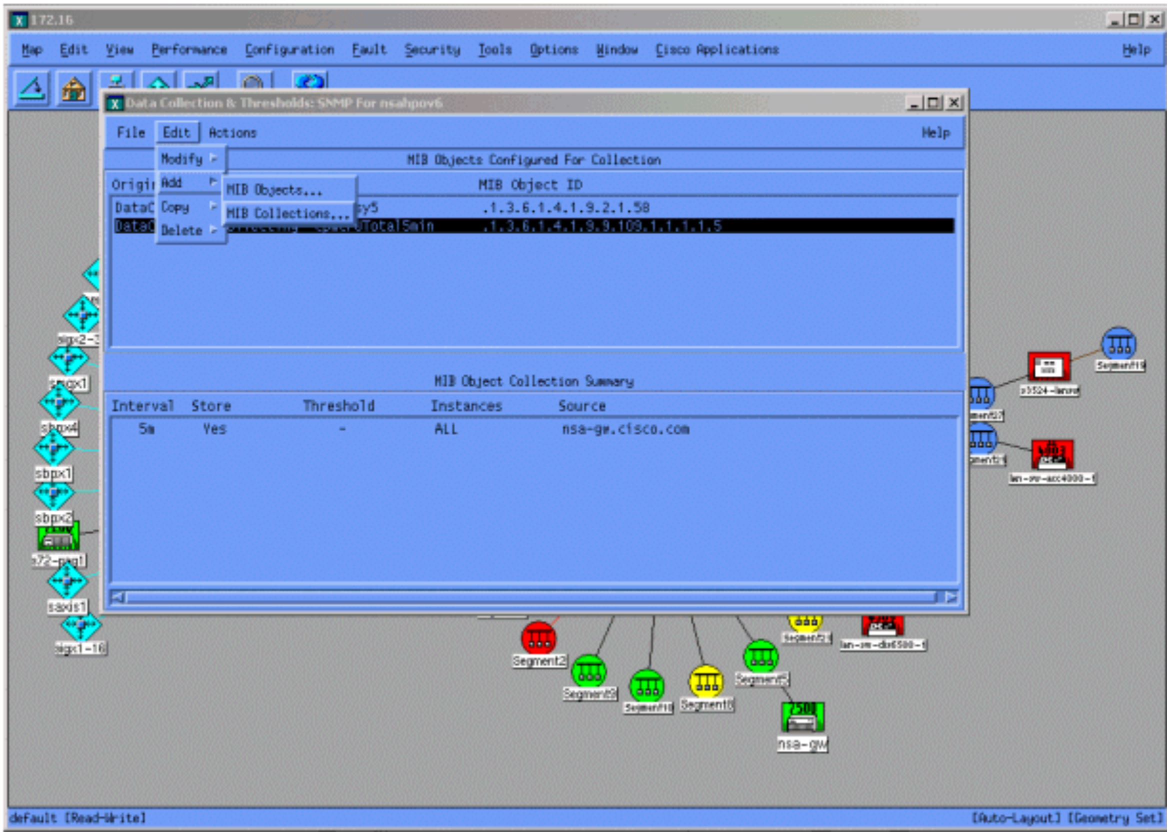
Label:

Messages

```
Cannot add duplicate object: .1.3.6.1.4.1.9.9.109.1.1.1.1.5.
  This object is already collected under label cpmCPUTotal5min.
```

You must next let HP OpenView know what router to poll for this OID.

From the Data Collection menu, select **Edit** > **Add** > **MIB Collections**.



MIB Objects Configured For Collection

Origi	MIB Object ID
DataC … y5	.1.3.6.1.4.1.9.2.1.58
DataC … TotalSmin	.1.3.6.1.4.1.9.9.109.1.1.1.1.5

MIB Object Collection Summary

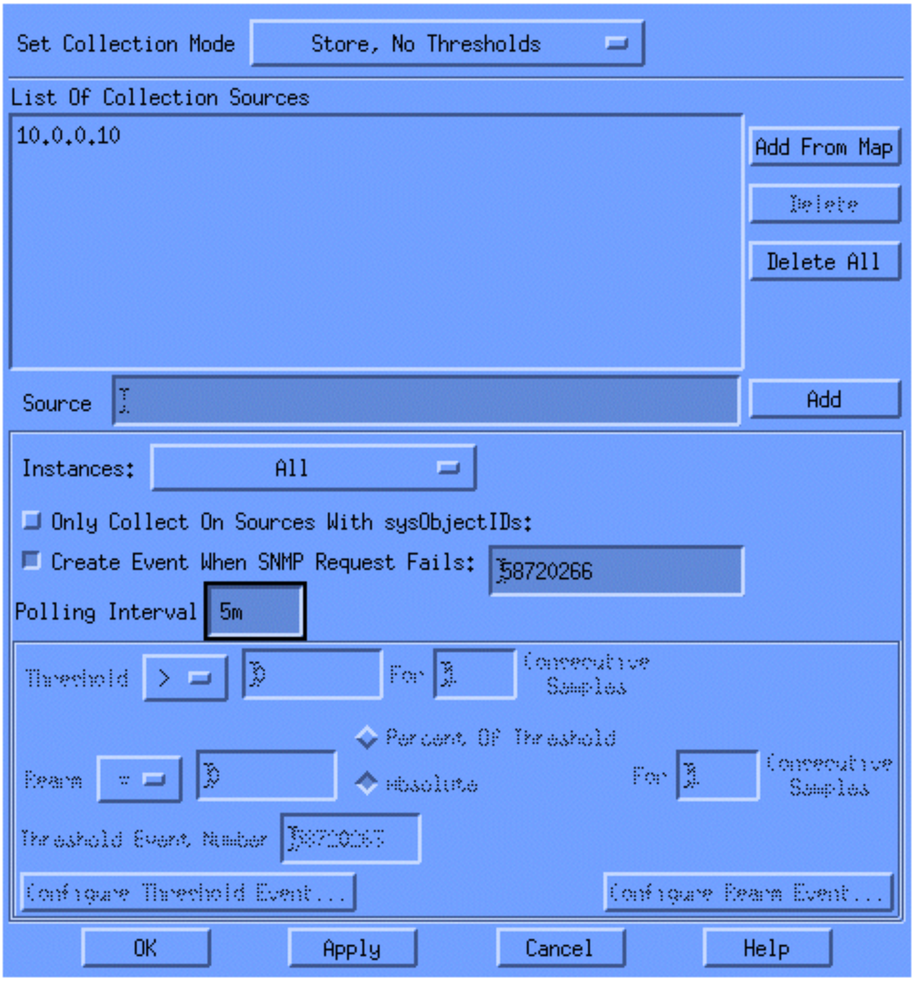
Interval	Store	Threshold	Instances	Source
5m	Yes	-	ALL	nsa-gw.cisco.com

In the Source field, enter the Domain Naming System (DNS) name or IP address of the router to be polled.

Select **Store, No Thresholds** from the Set Collection Mode list.

Set the Polling Interval to **5m**, for five minute intervals.

Click **Apply**.



Set Collection Mode: Store, No Thresholds

List Of Collection Sources: 10.0.0.10

Source:

Instances: All

□ Only Collect On Sources With sysObjectIDs:

□ Create Event When SNMP Request Fails: 58720266

Polling Interval: 5m

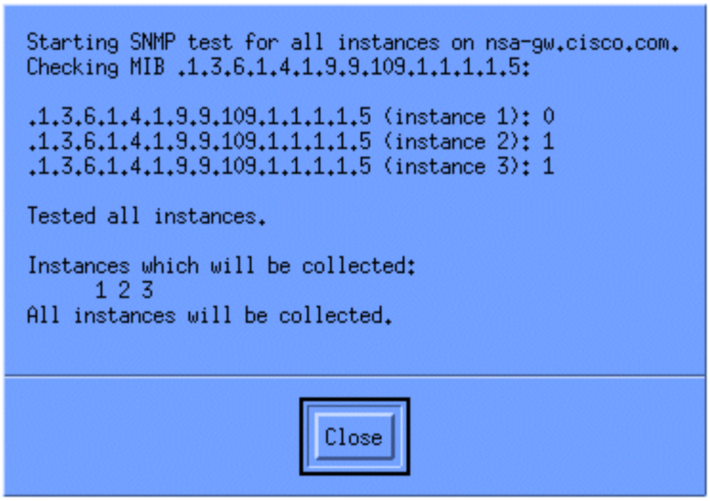
Threshold: > 0 For 1 Consecutive Samples

Rearm: = 0 ◇ Percent Of Threshold ◆ Absolute For 1 Consecutive Samples

Threshold Event Number: 58720267

You must select **File** > **Save** for the changes to take affect.

To verify that the collection is set up properly, highlight the collection summary line for the router and select **Actions** > **Test SNMP**. This checks to see if the community string is correct and will poll for all instances of the OID.



```
Starting SNMP test for all instances on nsa-gw.cisco.com.
Checking MIB .1.3.6.1.4.1.9.9.109.1.1.1.1.5:

.1.3.6.1.4.1.9.9.109.1.1.1.1.5 (instance 1): 0
.1.3.6.1.4.1.9.9.109.1.1.1.1.5 (instance 2): 1
.1.3.6.1.4.1.9.9.109.1.1.1.1.5 (instance 3): 1

Tested all instances.

Instances which will be collected:
      1 2 3
All instances will be collected.
```

Click **Close**, and let the collection run for a week. At the end of the weekly period, extract the data for analysis.

The data is more easily analyzed if you dump it to an ASCII file and import it into a spreadsheet tool such as Microsoft Excel. To do this with HP OpenView NNM, you can use the command line tool, **snmpColDump**. Each collection configured writes to a file in the /var/opt/OV/share/databases/snmpCollect/ directory.

Extract the data to an ASCII file called **testfile** with the following command:

```
snmpColDump /var/opt/OV/share/databases/snmpCollect/cpmCPUTotal5min.1 >
testfile
```

NOTE: cpmCPUTotal5min.1 is the database file that HP OpenView NNM created when the OID polling began.

The test file generated appears similar to the following example.

```
03/01/2001 14:09:10 nsa−gw.cisco.com 1
03/01/2001 14:14:10 nsa−gw.cisco.com 1
03/01/2001 14:19:10 nsa−gw.cisco.com 1
03/01/2001 14:24:10 nsa−gw.cisco.com 1
03/01/2001 14:29:10 nsa−gw.cisco.com 1
03/01/2001 14:34:10 nsa−gw.cisco.com 1
03/01/2001 14:39:10 nsa−gw.cisco.com 1
03/01/2001 14:44:10 nsa−gw.cisco.com 1
03/01/2001 14:49:10 nsa−gw.cisco.com 1
03/01/2001 14:54:10 nsa−gw.cisco.com 1
03/01/2001 14:59:10 nsa−gw.cisco.com 1
03/………
```

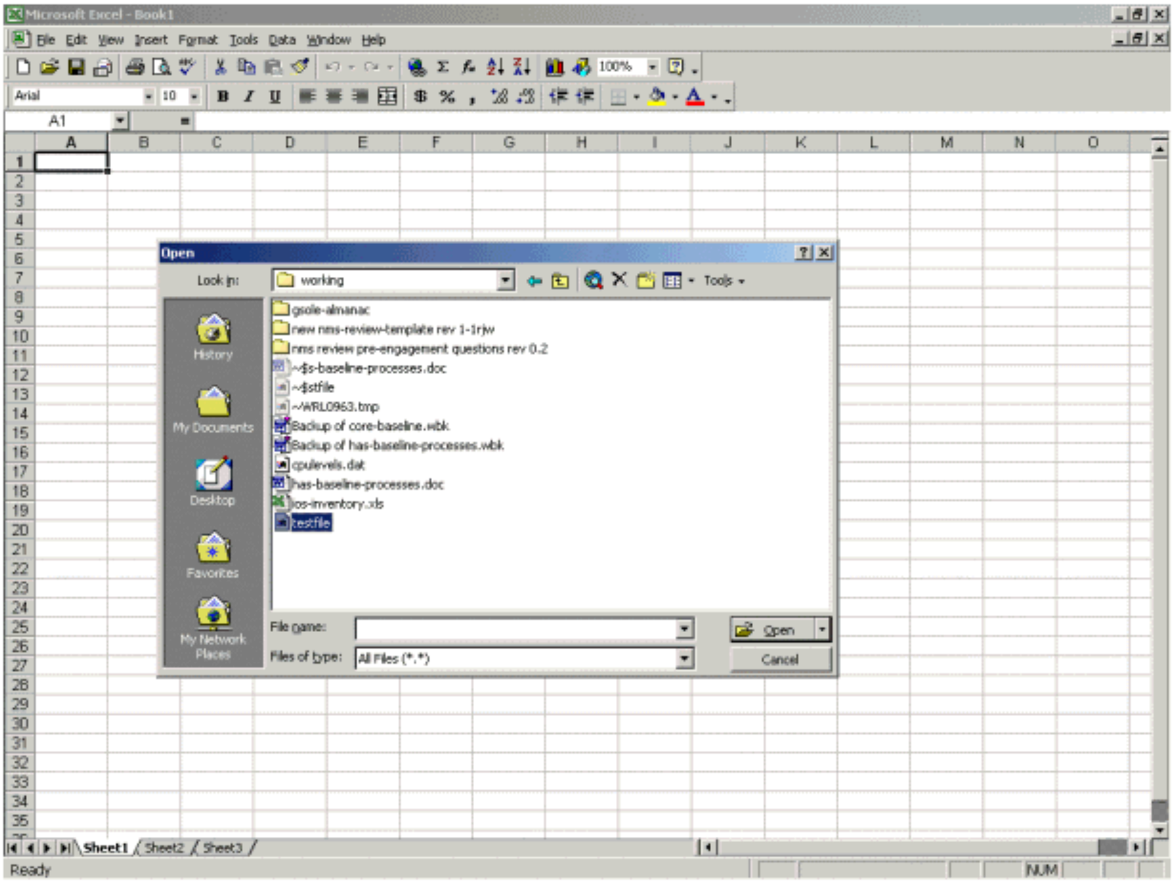
Once the test file output is on your UNIX station, you can transfer it to your PC using File Transfer Protocol (FTP).

You can also gather the data using your own scripts. To do this, perform an **snmpget** for the CPU OID every five minutes and dump the results into a .csv file.

Step 4: Analyze Data to Determine Thresholds

Now that you have some data, you can begin to analyze it. This phase of the baseline determines the threshold settings you can use that are an accurate measure of performance or fault and will not set off too many alarms when you turn on threshold monitoring. One of the easiest ways to do this is to import the data into a spreadsheet such as Microsoft Excel and plot a scatter chart. This method makes it very easy to see how many times a particular device would have created an exception alert if you were monitoring it for a certain threshold. It is not advisable to turn on thresholds without doing a baseline, since this may create alert storms from devices that have exceeded the threshold you have chosen.

To import the test file into an Excel spreadsheet, open Excel and select **File > Open** and select your data file.



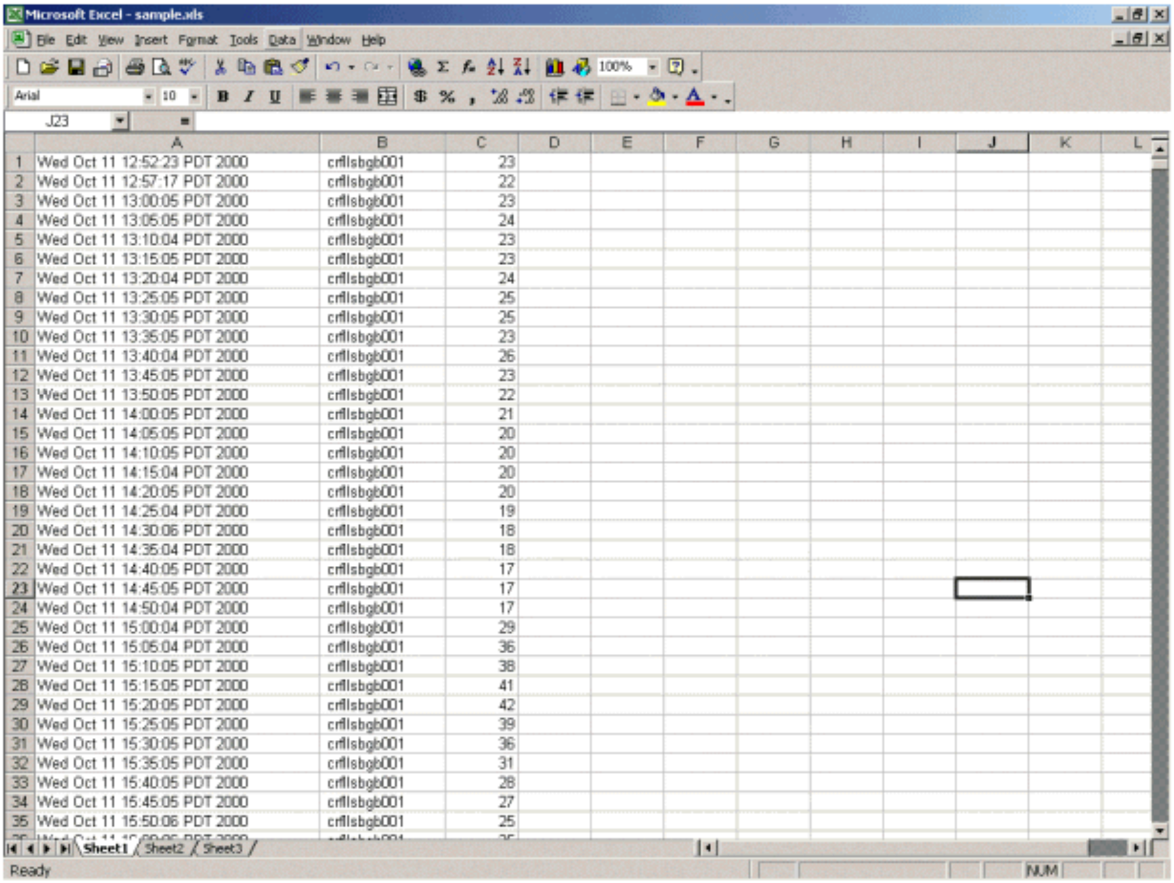
Look in: working

gsole-almanac
new nms-review-template rev 1-1rjw
nms review pre-engagement questions rev 0.2
~$s-baseline-processes.doc
~$stfile
~WRL0963.tmp
Backup of core-baseline.wbk
Backup of has-baseline-processes.wbk
cpulevels.dat
has-baseline-processes.doc
ios-inventory.xls
testfile

File name:
Files of type: All Files (*.*)

The Excel application then prompts you through importing the file.

When finished, the imported file should look similar to the following screen.



Microsoft Excel - sample.xls
File Edit View Insert Format Tools Data Window Help
Arial
10
100%
J23
Sheet1
Sheet2
Sheet3
Ready
NUM

	A	B	C
1	Wed Oct 11 12:52:23 PDT 2000	crfllsbgb001	23
2	Wed Oct 11 12:57:17 PDT 2000	crfllsbgb001	22
3	Wed Oct 11 13:00:05 PDT 2000	crfllsbgb001	23
4	Wed Oct 11 13:05:05 PDT 2000	crfllsbgb001	24
5	Wed Oct 11 13:10:04 PDT 2000	crfllsbgb001	23
6	Wed Oct 11 13:15:05 PDT 2000	crfllsbgb001	23
7	Wed Oct 11 13:20:04 PDT 2000	crfllsbgb001	24
8	Wed Oct 11 13:25:05 PDT 2000	crfllsbgb001	25
9	Wed Oct 11 13:30:05 PDT 2000	crfllsbgb001	25
10	Wed Oct 11 13:35:05 PDT 2000	crfllsbgb001	23
11	Wed Oct 11 13:40:04 PDT 2000	crfllsbgb001	26
12	Wed Oct 11 13:45:05 PDT 2000	crfllsbgb001	23
13	Wed Oct 11 13:50:05 PDT 2000	crfllsbgb001	22
14	Wed Oct 11 14:00:05 PDT 2000	crfllsbgb001	21
15	Wed Oct 11 14:05:05 PDT 2000	crfllsbgb001	20
16	Wed Oct 11 14:10:05 PDT 2000	crfllsbgb001	20
17	Wed Oct 11 14:15:04 PDT 2000	crfllsbgb001	20
18	Wed Oct 11 14:20:05 PDT 2000	crfllsbgb001	20
19	Wed Oct 11 14:25:04 PDT 2000	crfllsbgb001	19
20	Wed Oct 11 14:30:06 PDT 2000	crfllsbgb001	18
21	Wed Oct 11 14:35:04 PDT 2000	crfllsbgb001	18
22	Wed Oct 11 14:40:05 PDT 2000	crfllsbgb001	17
23	Wed Oct 11 14:45:05 PDT 2000	crfllsbgb001	17
24	Wed Oct 11 14:50:04 PDT 2000	crfllsbgb001	17
25	Wed Oct 11 15:00:04 PDT 2000	crfllsbgb001	29
26	Wed Oct 11 15:05:04 PDT 2000	crfllsbgb001	36
27	Wed Oct 11 15:10:05 PDT 2000	crfllsbgb001	38
28	Wed Oct 11 15:15:05 PDT 2000	crfllsbgb001	41
29	Wed Oct 11 15:20:05 PDT 2000	crfllsbgb001	42
30	Wed Oct 11 15:25:05 PDT 2000	crfllsbgb001	39
31	Wed Oct 11 15:30:05 PDT 2000	crfllsbgb001	36
32	Wed Oct 11 15:35:05 PDT 2000	crfllsbgb001	31
33	Wed Oct 11 15:40:05 PDT 2000	crfllsbgb001	28
34	Wed Oct 11 15:45:05 PDT 2000	crfllsbgb001	27
35	Wed Oct 11 15:50:06 PDT 2000	crfllsbgb001	25

A scatter chart allows you to more easily visualize how various threshold settings would work on the network.

To create the scatter chart, highlight column **C** in the imported file and then click the **Chart Wizard** icon. Then follow the steps through Chart Wizard for building a scatter chart.

In Chart Wizard step 1, as shown below, select the **Standard Types** tab, and select the **XY (Scatter)** chart type. Then click **Next**.

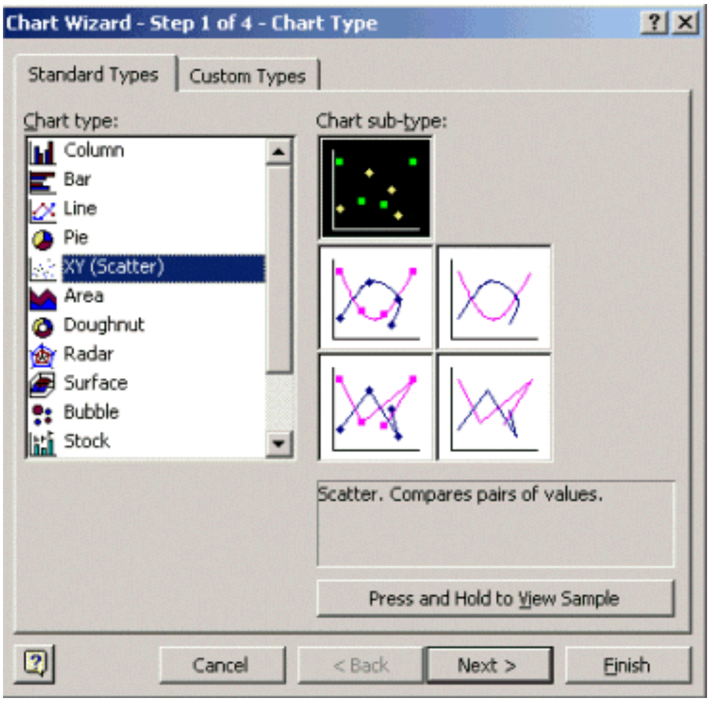
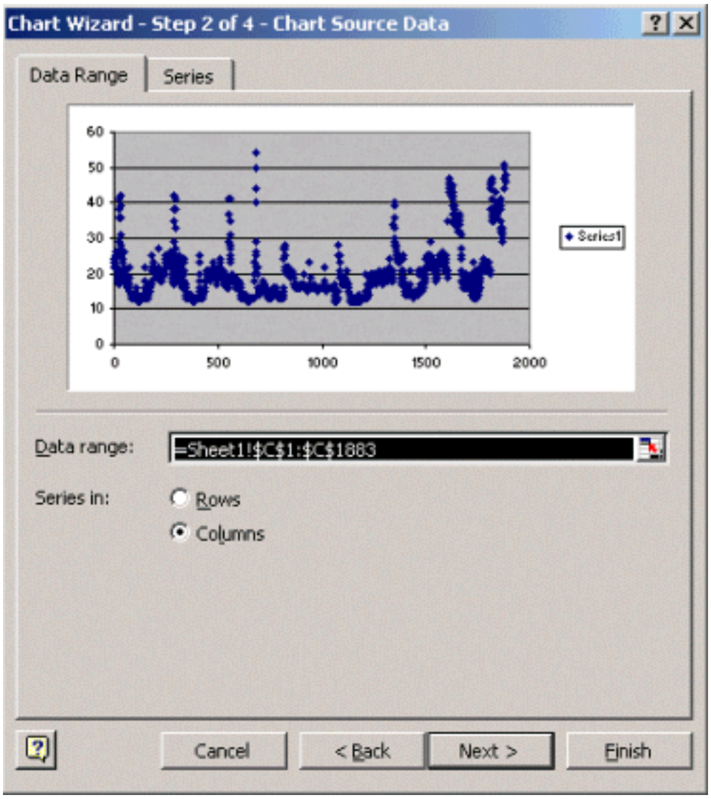


Chart type: XY (Scatter)

Chart sub-type:

Scatter. Compares pairs of values.

In Chart Wizard step 2, as shown below, select the **Data Range** tab and select the data range and the **Columns** option. Click **Next**.



Data range: =Sheet1!$C$1:$C$1883

Series in: ○ Rows ● Columns

In Chart Wizard step 3, as shown below, enter the chart title and the X and Y axis values, and then click **Next**.

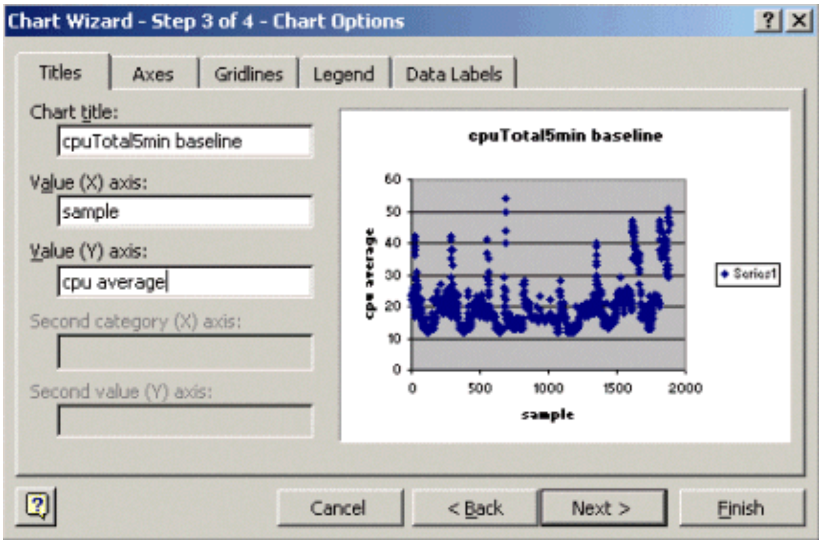


Chart Wizard - Step 3 of 4 - Chart Options

Chart title: cpuTotal5min baseline

Value (X) axis: sample

Value (Y) axis: cpu average

Second category (X) axis:

Second value (Y) axis:

In Chart Wizard step 4, select whether you want the scatter chart on a new page or as an object in the existing page.

Click **Finish** to place the chart in your desired location.

"What If?" Analysis

You can now use the scatter chart for analysis. However, before proceeding, you need to ask the following questions:

- What does the vendor (in this example the vendor is Cisco) recommend as a threshold for this MIB variable?

  In general, Cisco recommends that a core router does not exceed 60 percent average CPU utilization. Sixty percent was chosen because a router needs some overhead in case it experiences trouble or the network has some failures. Cisco estimates that a core router needs approximately 40 percent CPU overhead in case a routing protocol has to recalculate or reconverge. These percentages vary based on the protocols you use and the topology and stability of your network.

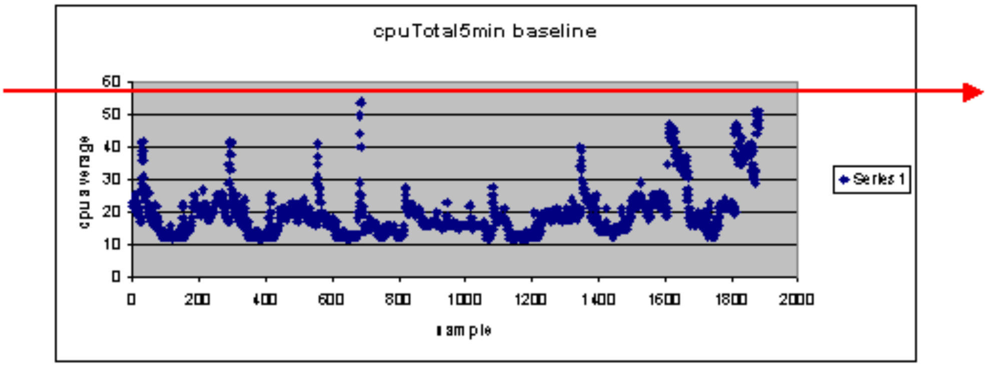
- What if I use 60 percent as the threshold setting?

  If you draw a line across the scatter chart horizontally at 60, you'll see that none of the data points exceed 60 percent CPU utilization. So a threshold of 60 set on your network management system (NMS) stations will not have set off a threshold alarm during the polling period. A percentage of 60 is acceptable for this router. However, notice in the scatter chart that some of the data points are close to 60. It would be nice to know when a router is nearing the 60 percent threshold so you can know ahead of time that the CPU is approaching 60 percent and have a plan for what to do when it reaches that point.

- What if I set the threshold to 50 percent?

  It is estimated that this router reached 50 percent utilization four times during this polling cycle and would have generated a threshold alarm each time. This process becomes more important when you look at *groups of routers* to see what the different threshold settings would do. For example, "What if I set the threshold at 50 percent for the entire core network?" You see, it is very difficult to choose just one number.

CPU Threshold "What If" Analysis



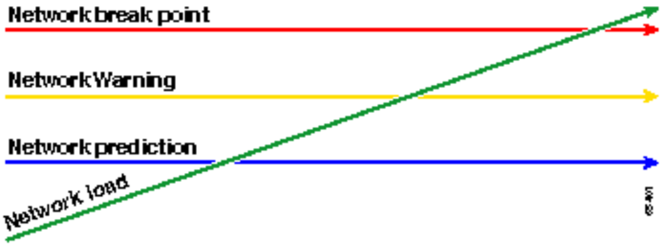
One strategy you can use to make this easier is the Ready, Set, Go threshold methodology. This methodology uses three threshold numbers in succession.

- Ready—the threshold you set as a predictor of what devices will likely need attention in the future
- Set—the threshold that is used as an early indicator, which alerts you to begin planning for a repair, reconfiguration, or upgrade
- Go—the threshold that you and/or the vendor believe is a fault condition and requires some action to repair it; in this example it is 60 percent

The following table shows the strategy of the Ready, Set, Go strategy.

Threshold	Action	Result
45 percent	Investigate further	List of options for action plans
50 percent	Formulate action plan	List of steps in action plan
60 percent	Implement action plan	Router no longer exceeds thresholds. Back to Ready mode

The Ready, Set, Go methodology changes the original baseline chart discussed earlier. The following diagram shows the changed baseline chart. If you can identify the other intersection points on the chart, you now have more time to plan and react than you did before.



Notice that in this process, attention is focused on the exceptions in the network and is not concerned with other devices. It is assumed that as long as devices are below thresholds, they are fine.

If you have these steps thought out from the beginning, you will be well prepared for keeping the network healthy. Performing this type of planning is also extremely useful for budget planning. If you know what your top five **go** routers, your middle **set** routers, and your bottom **ready** routers are, you can easily plan on how much budget you will need for upgrades based on what kind of routers they are and what your action plan options are. The same strategy can be used for wide−area network (WAN) links or any other MIB OID.

Step 5: Fix Identified Immediate Problems

This is one of the easier parts of the baseline process. Once you have identified which devices exceed the **go** threshold, you should make an action plan to get those devices back under threshold.

You can open a case with Cisco's Technical Assistance Center (TAC) or contact your Systems Engineer for available options. You should not assume that getting things back under threshold will cost you money. Some CPU issues can be resolved by changing the configuration to ensure that all processes are running in the most efficient manner. For example, some Access Control Lists (ACLs) can make a router CPU run very high due to the path the packets take through the router. In some cases, you can implement NetFlow switching to change the packet switching path and reduce the impact of the ACL on the CPU. Whatever the issues are, it is necessary to get all routers back under threshold in this step so you can implement the thresholds later without the risk of flooding the NMS stations with too many threshold alarms.

Step 6: Test Threshold Monitoring

This step involves testing the thresholds in the lab using the tools you will use in the production network. There are two common approaches to monitoring thresholds. You must decide which method is best for your network.

- Poll and compare method using an SNMP platform or other SNMP monitoring tool

  This method uses more network bandwidth for polling traffic and takes up processing cycles on your SNMP platform.
- Use Remote Monitoring (RMON) Alarm and Event configurations in the routers so they send an alert only when a threshold is exceeded

  This method reduces network bandwidth usage but also increases memory and CPU utilization on the routers.

Implementing a Threshold using SNMP

To set up the SNMP method using HP OpenView NNM, select **Options > Data Collection & Thresholds** as you did when you set up the initial polling. This time, however, select **Store, Check Thresholds** rather than Store, No Thresholds in the collections menu. After you set up the threshold, you can raise the CPU utilization on the router by sending it multiple pings and/or multiple SNMP walks. You may have to lower the threshold value if you can't force the CPU high enough to trip the threshold. In any case, you should ensure that the threshold mechanism is working.

One of the limitations of using this method is that you cannot implement multiple thresholds simultaneously. You would need three SNMP platforms to set three different simultaneous thresholds. Tools such as Concord Network Health and Trinagy TREND allow multiple thresholds for the same OID instance.

If your system can only handle one threshold at a time, you may consider the Ready, Set, Go strategy in serial fashion. That is, when the **ready** threshold is reached continually, begin your investigation and raise the threshold to the **set** level for that device. When the set level is reached continually, begin to formulate your action plan and raise the threshold to the **go** level for that device. Then when the go threshold is reached continually, implement your action plan. This should work just as well as the three simultaneous threshold method. It just takes a little more time changing the SNMP platform threshold settings.

Implementing a Threshold using RMON Alarm and Event

Using RMON alarm and event configurations, you can have the router monitor itself for multiple thresholds. When the router detects an over−threshold condition, it sends an SNMP trap to the SNMP platform. You must have an SNMP trap receiver set up in your router configuration for the trap to be forwarded. There is a correlation between an alarm and event. The alarm checks the OID for the given threshold. If the threshold is reached, the alarm process fires the event process which can either send an SNMP trap message, create an RMON log entry, or both. For more detail on this command, refer to the RMON Alarm and Event Configuration Commands section.

The following router configuration commands has the router monitor cpmCPUTotal5min every 300 seconds. It will fire event 1 if the CPU exceeds 60 percent and will fire event 2 when the CPU falls back to 40 percent. In both cases, an SNMP trap message will be sent to the NMS station with the community private string.

To use the Ready, Set, Go method, use all of the following configuration statements.

```
rmon event 1 trap private description "cpu hit60%" owner jharp
rmon event 2 trap private description "cpu recovered" owner jharp
rmon alarm 10 cpmCPUTotalTable.1.5.1 300 absolute rising 60 1 falling 40 2
owner jharp


rmon event 3 trap private description "cpu hit50%" owner jharp
rmon event 4 trap private description "cpu recovered" owner jharp
rmon alarm 20 cpmCPUTotalTable.1.5.1 300 absolute rising 50 3 falling 40 4
owner jharp

rmon event 5 trap private description "cpu hit 45%" owner jharp
rmon event 6 trap private description "cpu recovered" owner jharp
rmon alarm 30 cpmCPUTotalTable.1.5.1 300 absolute rising 45 5 falling 40 6
owner jharp
```

The following example shows the output of the **show rmon alarm** command that was configured by the above statements.

```
zack#sh rmon alarm
Alarm 10 is active, owned by jharp
 Monitors cpmCPUTotalTable.1.5.1 every 300 second(s)
 Taking absolute samples, last value was 0
 Rising threshold is 60, assigned to event 1
 Falling threshold is 40, assigned to event 2
 On startup enable rising or falling alarm
Alarm 20 is active, owned by jharp
 Monitors cpmCPUTotalTable.1.5.1 every 300 second(s)
 Taking absolute samples, last value was 0
 Rising threshold is 50, assigned to event 3
 Falling threshold is 40, assigned to event 4
 On startup enable rising or falling alarm
Alarm 30 is active, owned by jharp
 Monitors cpmCPUTotalTable.1.5.1 every 300 second(s)
 Taking absolute samples, last value was 0
 Rising threshold is 45, assigned to event 5
 Falling threshold is 40, assigned to event 6
 On startup enable rising or falling alarm
```

The following example shows the output of the **show rmon event** command.

```
zack#sh rmon event
Event 1 is active, owned by jharp
 Description is cpu hit60%
 Event firing causes trap to community private, last fired 00:00:00
Event 2 is active, owned by jharp
 Description is cpu recovered
 Event firing causes trap to community private, last fired 02:40:29
Event 3 is active, owned by jharp
 Description is cpu hit50%
 Event firing causes trap to community private, last fired 00:00:00
Event 4 is active, owned by jharp
 Description is cpu recovered
 Event firing causes trap to community private, last fired 00:00:00
Event 5 is active, owned by jharp
 Description is cpu hit 45%
 Event firing causes trap to community private, last fired 00:00:00
Event 6 is active, owned by jharp
 Description is cpu recovered
 Event firing causes trap to community private, last fired 02:45:47
```

You may want to try both of these methods to see which method best suits your environment. You may even find that a combination of methods works well. In any case, testing should be done in a lab environment to ensure that everything works correctly. After testing in the lab, a limited deployment on a small group of routers will allow you to test the process of sending alerts to your Operations Center.

In this case, you will have to lower the thresholds to test the process: Trying to artificially raise the CPU on a production router is not recommended. You should also ensure that when the alerts come into the NMS stations at the Operations Center, there is an escalation policy to make sure that you are informed when devices exceed thresholds. These configurations have been tested in a lab with Cisco IOS Version 12.1(7). If you encounter any issues, you should check with Cisco Engineering or Systems Engineers to see if you have a bug in your IOS version.

Step 7: Implement Threshold Monitoring using SNMP or RMON

Once you have thoroughly tested threshold monitoring in the lab, and in a limited deployment, you are ready to implement thresholds across the core network. You can now systematically go through this baseline process for other important MIB variables on your network, such as buffers, free memory, cyclic redundancy check (CRC) errors, AMT cell loss, and so on.

If you use RMON alarm and event configurations, you can now stop polling from your NMS station. This will reduce the load on your NMS server and will reduce the amount of polling data on the network. By systematically going through this process for important network health indicators, you could easily come to the point that the network equipment are monitoring themselves using RMON Alarm and Event.

Additional MIBs

After you have learned this process, you may want to investigate other MIBs to baseline and monitor. The following subsections present a brief list of some OIDs and descriptions that you may find useful.

Router MIBs

Memory characteristics are very helpful in determining the health of a router. A healthy router should almost always have available buffer space with which to work. If the router begins to run out of buffer space, the

CPU will have to work harder to create new buffers and to try to find buffers for incoming and outgoing packets. An in−depth discussion of buffers is beyond the scope of this document. However, as a general rule, a healthy router should have very few, if any, buffer misses and should not have any buffer failures, or a zero free memory condition.

Object	Description	OID
ciscoMemoryPoolFree	The number of bytes from the memory pool that are currently unused on the managed device	1.3.6.1.4.1.9.9.48.1.1.1.6
ciscoMemoryPoolLargestFree	The largest number of contiguous bytes from the memory pool that are currently unused	1.3.6.1.4.1.9.9.48.1.1.1.7
bufferElMiss	The number of buffer element misses	1.3.6.1.4.1.9.2.1.12
bufferFail	The number of buffer allocation failures	1.3.6.1.4.1.9.2.1.46
bufferNoMem	The number of buffer create failures due to no free memory	1.3.6.1.4.1.9.2.1.47

Catalyst Switch MIBs

Object	Description	OID
cpmCPUTotal5min	Overall CPU busy percentage in the last five−minute period. This object deprecates the avgBusy5 object from the OLD−CISCO−SYSTEM−MIB	1.3.6.1.4.1.9.9.109.1.1.1.5
cpmCPUTotal5sec	Overall CPU busy percentage in the last five−second period. This object obsolesces the busyPer object from the OLD−CISCO−SYSTEM−MIB	1.3.6.1.4.1.9.9.109.1.1.1.3

sysTraffic	The percentage of bandwidth utilization for the previous polling interval	1.3.6.1.4.1.9.5.1.1.8
sysTrafficPeak	The peak traffic meter value since the last time the port counters were cleared or the system started	1.3.6.1.4.1.9.5.1.1.19
sysTrafficPeaktime	The time (in hundredths of a second) since the peak traffic meter value occurred	1.3.6.1.4.1.9.5.1.1.20
portTopNUtilization	Utilization of the port in the system	1.3.6.1.4.1.9.5.1.20.2.1.4
portTopNBufferOverFlow	The number of buffer overflows of the port in the system	1.3.6.1.4.1.9.5.1.20.2.1.10

Serial Link MIBs

Object	Description	OID
locIfInputQueueDrops	The number of packets dropped because the input queue was full	1.3.6.1.4.1.9.2.2.1.1.26
locIfOutputQueueDrops	The number of packets dropped because the output queue was full	1.3.6.1.4.1.9.2.2.1.1.27
locIfInCRC	The number of input packets that had cyclic redundancy checksum errors	1.3.6.1.4.1.9.2.2.1.1.12

RMON Alarm and Event Configuration Commands

Alarms

RMON alarms can be configured with the following syntax:

```
rmon alarm number variable interval {delta | absolute} rising-threshold
value
            [event-number] falling-threshold value [event-number]
            [owner string]
```

Element	Description
number	The alarm number, which is identical to the alarmIndex in the alarmTable in the RMON MIB.
variable	The MIB object to monitor, which translates into the alarmVariable used in the alarmTable of the RMON MIB.
interval	

	The time, in seconds, the alarm monitors the MIB variable, which is identical to the alarmInterval used in the alarmTable of the RMON MIB.
delta	Tests the change between MIB variables, which affects the alarmSampleType in the alarmTable of the RMON MIB.
absolute	Tests each MIB variable directly, which affects the alarmSampleType in the alarmTable of the RMON MIB.
rising−threshold value	The value at which the alarm is triggered.
event−number	(Optional) The event number to trigger when the rising or falling threshold exceeds its limit. This value is identical to the alarmRisingEventIndex or the alarmFallingEventIndex in the alarmTable of the RMON MIB.
falling−threshold value	The value at which the alarm is reset.
owner string	(Optional) Specifies an owner for the alarm, which is identical to the alarmOwner in the alarmTable of the RMON MIB.

Events

RMON events can be configured with the following syntax:

```
rmon event number [log] [trap community] [description string]
           [owner string]
```

Element	Description
number	Assigned event number, which is identical to the eventIndex in the eventTable in the RMON MIB.
log	(Optional) Generates an RMON log entry when the event is triggered and sets the eventType in the RMON MIB to log or log−and−trap.
trap community	(Optional) SNMP community string used for this trap. Configures the setting of the eventType in the RMON MIB for this row as either snmp−trap or log−and−trap. This value is identical to the eventCommunityValue in the eventTable in the RMON MIB.
description string	(Optional) Specifies a description of the event, which is identical to the event description in the eventTable of the RMON MIB.
owner string	(Optional) Owner of this event, which is identical to the eventOwner in the eventTable of the RMON MIB.

RMON Alarm and Event Implementation

For detailed information about RMON alarm and event implementation, please read the RMON Alarm and Event Implementation section of the *Network Management Systems Best Practices* white paper.

Related Information

- **Best Practices White Papers**
- **Network Management General Tips**

Updated: May 30, 2002 Document ID: 15112