CHAPTER 21

Using SNMP

Revised: March 20, 2009, OL-17222-03

This chapter provides information about Cisco Access Registrar (CAR) support for SNMP.

Overview

CAR provides SNMP MIB and trap support for users of network management systems. The supported MIBs enable the network management station to collect state and statistic information from an CAR server. The traps enable CAR to notify interested network management stations of failure or impending failure conditions.

CAR supports the MIBs defined in the following RFCs:

- RADIUS Authentication Client MIB, RFC 2618

  RADIUS Authentication Server MIB, RFC 2619

  RADIUS Accounting Client MIB, RFC 2620

  RADIUS Accounting Server MIB, RFC 2621

CAR MIB support enables a standard SNMP management station to check the current state of the server as well as the statistics on each client or each proxied remote server.

CAR Trap support enables a standard SNMP management station to receive trap messages from an CAR server. These messages contain information indicating that either the server was brought up or down, or that the proxied remote server is down or has come back online.

Supported MIBs

RADIUS-AUTH-CLIENT-MIB

The RADIUS-AUTH-CLIENT-MIB describes the client side of the RADIUS authentication protocol. The information contained in this MIB is useful when an CAR server is used as a proxy server.

RADIUS-AUTH-SERVER-MIB

The RADIUS-AUTH-SERVER-MIB describes the server side of the RADIUS authentication protocol. The information contained in this MIB describes managed objects used for managing a RADIUS authentication server.

The RADIUS-ACC-CLIENT-MIB describes the client side of the RADIUS accounting protocol. The information contained in this MIB is useful when an CAR server is used for accounting.

The RADIUS-ACC-CLIENT-MIB describes the server side of the RADIUS accounting protocol. The information contained in this MIB is useful when an CAR server is used for accounting.

SNMP Traps

When a trap is generated, a single copy of the trap is transmitted as a trap PDU to each destination contained within a list of trap recipients.

The list of trap recipients is shared by all events and is determined at server initialization time along with other trap configuration information. The list of trap recipients dictates where CAR traps are directed.

The configuration of any other SNMP agent on the host is ignored. By default, all traps are enabled but no trap recipients are defined. By default, no trap is sent until trap recipients are defined.

Traps are configured using the command line interface (CLI). After configuring traps, the configuration information is re initialized when a server reload or restart occurs.

When you configure traps, you must provide the following information:

List of trap recipients (community string for each)

Suppressing traps for any type of message

Frequency of traps for any type of message

Supported Traps

carServerStart

carServerStart

carNotifStartType, *firstStart*
the server process' first start. *reload*
reload

carServerStop

carServerStop

carInputQueueFull

carInputQueueFull

carNotifInputQueueHighThreshold—

carNotifInputQueueLowThreshold—

carNotifInputQueueHighThreshold is set to 90% and *carNotifInputQueueLowThreshold* is set to 60%.

Note The values for these objects cannot be changed at this time. You will be able to modify them in a future release of CAR.

After this notification has been sent, another notification of this type will not be sent again until the percentage usage of the input queue goes below the low threshold.

If the percentage usage reaches 100%, successive requests might be dropped, and the server might stop responding to client requests until the queue drops down again.

carInputQueueNotVeryFull

carInputQueueNotVeryFull

carOtherAuthServerNotResponding

carOtherAuthServerNotResponding

radiusAuthServerAddress—

radiusAuthClientServerPortNumber—

carAuthServerType—indicates the type of the concerned server

The index of these three objects identifies the entry in and *carAccServerExtTable*

One should not rely solely on for server state. Several conditions, including a restart of the CAR server, could result in either multiple *carOtherAuthServerNotResponding* *carOtherAuthServerResponding* *not* *carAuthServerRunningState* *carAuthServerExtTable*

down

radiusAuthServerAddress—

radiusAuthClientServerPortNumber—

carAuthServerType

radiusAuthServerTable *carAccServerExtTable*

carOtherAuthServerNotResponding *carOtherAuthServerResponding* *not* *carAuthServerRunningState* *carAuthServerExtTable*

radiusAccServerAddress—

radiusAccClientServerPortNumber—

carAcchServerType

radiusAuthServerTable *arAccServerExtTable*

carOtherAccServerNotResponding *carOtherAccServerResponding* *not* *carAccServerRunningState* *carAccServerExtTable*

not *responding*

radiusAccServerAddress—

radiusAccClientServerPortNumber—

carAccServerType

radiusAuthServerTable *arAccServerExtTable*

carOtherAccServerNotResponding
not *carAccServerRunningState* *carAccServerExtTable*

carAccountingLoggingFailure

carAccountingLoggingFailure

carLicenseUsage

carLicenseUsage

Configuring Traps

Directories Searched

1. /usr/local/share/snmp/snmp.conf

   This directory contains common configuration for the agent and the application. See man page **snmp.conf(5)**

snmp.conf(5)

.conf

("public" by default) is allowed read-only access and the second ("private" by default) is allowed write access, as well. The third to fifth community strings are also read-only.

Additionally, the above default search path can be over-ridden by setting the environmental variable SNMPCONFPATH to a colon-separated list of directories to search.

Finally, applications that store persistent data will also look for configuration files in the **/var/snmp** directory.

Configuration File Types

snmp.conf

snmp.conf

with each configuration file type. Also, most of the applications support a '-H' switch on the command line that will list the configuration files it will look for and the directives in each one that it understands.

The configuration file is intended to be a application suite-wide configuration file that supports directives that are useful for controlling the fundamental nature of all of the SNMP applications, such as how they all manipulate and parse the textual SNMP MIB files.

Switching Configuration Files in Mid-File

snmpwalk **snmp.conf**

```
dumpPacket true
```

this line. You do this by putting a special type specification token inside a square bracket ([ ]) set. In other words, inside your file you could put the above directive by adding a line like the following:

```
[snmp] dumpPacket true
```

```
# make this file handle snmp.conf tokens:
[snmp]
dumpPacket true
logTimestamp true
# return to our original snmpd.conf tokens:
[snmpd]
rocommunity mypublic
```

Community String