Using SNMP

This chapter provides the following information about Cisco Prime Access Registrar (Prime Access Registrar) support for SNMP:

- Overview, page 15-1
- Supported MIBs, page 15-1
- SNMP Traps, page 15-3
- SNMP Version 3 Support, page 15-13

Overview

Prime Access Registrar 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 Prime Access Registrar server. The traps enable Prime Access Registrar to notify interested network management stations of failure or impending failure conditions.

Prime Access Registrar supports the MIBs defined in the following RFCs:

- RADIUS Authentication Client MIB for IPv6, RFC 4668
- RADIUS Authentication Server MIB for IPv6, RFC 4669
- RADIUS Accounting Client MIB for IPv6, RFC 4670
- RADIUS Accounting Server MIB for IPv6, RFC 4671
- CISCO Diameter Base Protocol MIB

Prime Access Registrar 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.

Prime Access Registrar Trap support enables a standard SNMP management station to receive trap messages from an Prime Access Registrar 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

The MIBs supported by Prime Access Registrar enable a standard SNMP management station to check the current state of the server and statistics for each client or proxied remote server.

This section contains the following topics:

- RADIUS-AUTH-CLIENT-MIB
- RADIUS-AUTH-SERVER-MIB
- RADIUS-ACC-CLIENT-MIB
- RADIUS-ACC-SERVER-MIB
- CISCO-DIAMETER-BASE-PROTOCOL-MIB
- Diameter SNMP and Statistics Support
- TACACS+ SNMP and Statistics Support

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 Prime Access Registrar 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.

RADIUS-ACC-CLIENT-MIB

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

RADIUS-ACC-SERVER-MIB

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

CISCO-DIAMETER-BASE-PROTOCOL-MIB

Prime Access Registrar uses the CISCO-DIAMETER-BASE-PROTOCOL-MIB as an interface to query the Diameter statistics, though configuring the Diameter through SNMP is not possible.
Prime Access Registrar supports LocalStatistics and PeerStatiscics only. The LocalStats provides statistical information about the local diameter server and the PeerStats provides statistical information about the peers and the messages to/from the peers.

Diameter SNMP and Statistics Support

Prime Access Registrar also supports Diameter SNMP MIB (CISCO-DIAMETER-BASE-PROTOCOL-MIB) to describe the Diameter Base Protocol statistics.

Prime Access Registrar supports statistic of Diameter messages to include the additional counters. This is supported through the CLI/GUI and SNMP. The diameter statistics includes peer statistics and global summary statistics details.

TACACS+ SNMP and Statistics Support

Prime Access Registrar supports the CISCO-AAA-SERVER-MIB to describe the statistics of TACACS+ protocol.TACACS+ protocol is used to authenticate an user via various services such as login services This is supported through the CLI/GUI and SNMP.

SNMP Traps

The traps supported by Prime Access Registrar enable a standard SNMP management station to receive trap messages from an Prime Access Registrar server. These messages contain information indicating whether a server was brought up or down, or that the proxied remote server is down or has come back online.

A trap is a network message of a specific format issued by an SNMP entity on behalf of a network management agent application. A trap is used to provide the management station with an asynchronous notification of an event.

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 Prime Access Registrar 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.

Note SNMP queries and traps communication can be performed over IPv6.

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

This topic contains the following sections:

- Supported Traps, page 15-4
- Configuring Traps, page 15-12

- Community String, page 15-12

Supported Traps

The traps supported by Prime Access Registrar enable the Prime Access Registrar server to notify interested management stations of events, failure, or impending failure conditions. Traps are a network message of a specific format issued by an SNMP entity on behalf of a network management agent application. Traps are used to provide the management station with an asynchronous notification of an event.

This section contains the following topics:

- carServerStart
- carServerStop
- carInputQueueFull
- carInputQueueNotVeryFull
- carDiaInputQueueFull
- carDiaInputQueueNotFull
- carOtherAuthServerNotResponding
- carOtherAuthServerResponding
- carOtherAccServerNotResponding
- carOtherAccServerResponding
- carAccountingLoggingFailure
- carLicenseUsage
- carSigtranLicenseUsage
- carDiameterPeerDown
- carDiameterPeerUp
- carTPSCapacityFull
- carTPSCapacityNotFull
- carSigtranTPSCapacityFull
- carSigtranTPSCapacityNotFull
- carSessionCapacityFull
- carSessionCapacityNotFull
- carSigtranSessionCapacityFull
- carSigtranSessionCapacityNotFull
- carLicenseUsageReset
- carSigtranLicenseUsageReset
- carReplicationSyncFailure
- carReplicationSuccess
- TLSClientConnectionUpTrap
- TLSClientConnectionClosedTrap

carServerStart

carServerStart signifies that the server has started on the host from which this notification was sent. This trap has one object, *carNotifStartType,* which indicates the start type. A *firstStart* indicates this is the server process' first start. *reload* indicates this server process has an internal reload. This typically occurs after rereading some configuration changes, but *reload* indicates this server process did not quit during the reload process.

carServerStop

carServerStop signifies that the server has stopped normally on the host from which this notification was sent.

carInputQueueFull

carInputQueueFull indicates that the percentage of use of the packet input queue has reached its high threshold. This trap has two objects:

- *carNotifInputQueueHighThreshold*—indicates the high limit percentage of input queue usage
- *carNotifInputQueueLowThreshold*—indicates the low limit percentage of input queue usage

By default, *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 Prime Access Registrar.

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 indicates that the percentage usage of the packet input queue has dropped below the low threshold defined in *carNotifInputQueueLowThreshold*. This trap has two objects:

- *carNotifInputQueueHighThreshold*—indicates the high limit percentage of input queue usage
- *carNotifInputQueueLowThreshold*—indicates the low limit percentage of input queue usage

After this type of notification has been sent, it will not be sent again until the percentage usage goes back up above the high threshold defined in *carNotifInputQueueHighThreshold*.

carDiaInputQueueFull

carDiaInputQueueFull signifies that the percentage of use of the Diameter packet input queue has reached its high threshold. This trap has two objects:

- *carNotifDiaInputQueueHighThreshold*—indicates the high limit percentage of Diameter packet input queue usage.

- *carNotifDiaInputQueueLowThreshold*—indicates the low limit percentage of the Diameter packet input queue usage.

If the percentage usage reaches 100%, successive request is dropped, and the server stops responding to client requests until the queue drops down again. After this notification is sent, this type of notification will not be sent again until the percentage usage of the input queue goes back down below the low threshold.

carDiaInputQueueNotFull

carDiaInputQueueNotFull signifies that the percentage of use of Diameter packet input queue has dropped below the low threshold defined in carNotifDiaInputQueueLowThreshold. This trap has two objects:

- *carNotifDiaInputQueueHighThreshold*—indicates the high limit percentage of Diameter packet input queue usage.
- *carNotifDiaInputQueueLowThreshold*—indicates the low limit percentage of the Diameter packet input queue usage.

After this type of notification has been sent, it will not be sent again until the percentage usage goes back up above the high threshold defined in *carNotifDiaInputQueueHighThreshold*.

carOtherAuthServerNotResponding

carOtherAuthServerNotResponding indicates that an authentication server is not responding to a request sent from this server. This trap has three objects:

- *radiusAuthServerAddress*—indicates the identity of the concerned server
- *radiusAuthClientServerPortNumber*—indicates the port number of the concerned server
- *carAuthServerType*—indicates the type of the concerned server

The index of these three objects identifies the entry in *radiusAuthServerTable* and *carAccServerExtTable* which maintains the characteristics of the concerned server.

Note One should not rely solely on **carOtherAuthServerNotResponding** for server state. Several conditions, including a restart of the Prime Access Registrar server, could result in either multiple *carOtherAuthServerNotResponding* notifications being sent or in a *carOtherAuthServerResponding* notification *not* being sent. NMS can query the *carAuthServerRunningState* in *carAuthServerExtTable* for the current running state of this server.

carOtherAuthServerResponding

carOtherAuthServerResponding signifies that an authentication server which had formerly been in a *down* state is now responding to requests from the Prime Access Registrar server. This trap has three objects:

- *radiusAuthServerAddress*—indicates the identity of the concerned server
- *radiusAuthClientServerPortNumber*—indicates the port number of the concerned server
- *carAuthServerType*—indicates the type of the concerned server

The index of these three objects identifies the entry in *radiusAuthServerTable* and *carAccServerExtTable* which maintains the characteristics of the concerned server.

One should not rely on receiving this notification as an indication that all is well with the network. Several conditions, including a restart of the Prime Access Registrar server, could result in either multiple *carOtherAuthServerNotResponding* notifications being sent or in a *carOtherAuthServerResponding* notification *not* being sent. The NMS can query the *carAuthServerRunningState* in *carAuthServerExtTable* for the current running state of this server.

carOtherAccServerNotResponding

carOtherAuthServerNotResponding signifies that an accounting server is not responding to the requests sent from this server. This trap has three objects:

- *radiusAccServerAddress*—indicates the identity of the concerned server
- *radiusAccClientServerPortNumber*—indicates the port number of the concerned server
- *carAcchServerType*—indicates the type of the concerned server

The index of these three objects identifies the entry in *radiusAuthServerTable* and *arAccServerExtTable* which maintains the characteristics of the concerned server.

One should not solely rely on this for server state. Several conditions, including the restart of the Prime Access Registrar server, could result in either multiple *carOtherAccServerNotResponding* notifications being sent or in a *carOtherAccServerResponding* notification *not* being sent. The NMS can query the *carAccServerRunningState* in *carAccServerExtTable* for current running state of this server.

carOtherAccServerResponding

carOtherAccServerResponding signifies that an accounting server that had previously sent a *not responding* message is now responding to requests from the Prime Access Registrar server. This trap has three objects:

- *radiusAccServerAddress*—indicates the identity of the concerned server
- *radiusAccClientServerPortNumber*—indicates the port number of the concerned server
- *carAccServerType*—indicates the type of the concerned server

The index of these three objects identifies the entry in *radiusAuthServerTable* and *arAccServerExtTable* which maintains the characteristics of the concerned server.

One should not rely on the reception of this notification as an indication that all is well with the network. Several conditions, including the restart of the Prime Access Registrar server, could result in either multiple *carOtherAccServerNotResponding* notifications being sent or in a **carOtherAccServerResponding** notification *not* being sent. The NMS can query the *carAccServerRunningState* in *carAccServerExtTable* for the current running state of this server.

carAccountingLoggingFailure

carAccountingLoggingFailure signifies that this Prime Access Registrar server cannot record accounting packets locally. This trap has two objects:

- carNotifAcctLogErrorReason—indicates the reason packets cannot be recorded locally
- *carNotifAcctLogErrorInterval*—indicates how long to wait until another notification of this type might be sent. A value of 0 (zero) indicates no time interval checking, meaning that no new notification can be sent until the error condition is corrected.

carLicenseUsage

carLicenseUsage signifies the percentage of transactions per second (TPS) usage or session usage from the available license values.

TPS

The TPS trap is generated when the Prime Access Registrar server reaches license usage slabs namely 80%, 90%, 100%, and 110%. These traps are generated only once for every slab during the increasing steady state. Increasing steady state is a state when Prime Access Registrars' incoming request rate shows 80% of the license usage over a period of 20 minutes. These traps will be regenerated only if a increasing steady state is observed after a decreasing steady state.

Concurrent Session

The concurrent session trap is generated when the Prime Access Registrar server reaches 80%. The incoming traffic slabs defined for trap generation are 80%, 90%, 100%, and 110% of the licensed Concurrent Sessions. These traps are generated once for every slab during the increasing steady state.

carSigtranLicenseUsage

carSigtranLicenseUsageTrap signifies the percentage of SIGTRAN TPS usage or SIGTRAN session usage from the available license values.

carDiameterPeerDown

carDiameterPeerDown signifies that a Diameter peer is down. The identity of the peer is given by cdbpPeerIpAddress.

carDiameterPeerUp

carDiameterPeerUp signifies that a Diameter peer is up. The identity of the peer is given by cdbpPeerIpAddress.

carTPSCapacityFull

carTPSCapacityFull signifies that the TPS of the Prime Access Registrar server has reached the configured high threshold capacity. This trap has the following objects:

- *carNotifTPSHighThreshold*—indicates the maximum limit of the TPS of the Prime Access Registrar server.
- *carNotifTPSLowThreshold*—indicates the minimum limit of the TPS of the Prime Access Registrar server.
- *carServerTPSUsage*—indicates the current TPS usage of the Prime Access Registrar server.

After this notification is sent, this type of notification will not be sent again until the TPS of Prime Access Registrar server reduces below the configured *carNotifTPSLowThreshold* value.

carTPSCapacityNotFull

carTPSCapacityNotFull signifies that the TPS of the Prime Access Registrar server has dropped below the configured low threshold capacity. This trap has the following objects:

- *carNotifTPSHighThreshold*—indicates the maximum limit of the TPS of the Prime Access Registrar server.
- *carNotifTPSLowThreshold*—indicates that the minimum limit of the TPS of the Prime Access Registrar server.
- *carServerTPSUsage*—indicates the current TPS usage of the Prime Access Registrar server.

After this notification is sent, this type of notification will not be sent again until the TPS of Prime Access Registrar server increases beyond the configured *carNotifTPSHighThreshold* value.

carSigtranTPSCapacityFull

carSigtranTPSCapacityFull signifies that the SIGTRAN TPS of the Prime Access Registrar server has reached the configured high threshold capacity. This trap has the following objects:

- *carNotifSigtranTPSHighThreshold*—indicates the maximum limit of the SIGTRAN TPS of the Prime Access Registrar server.
- *carNotifSigtranTPSLowThreshold*—indicates the minimum limit of the SIGTRAN TPS of the Prime Access Registrar server.
- *carServerSigtranTPSUsage*—indicates the current SIGTRAN TPS usage of the Prime Access Registrar server.

After this notification is sent, this type of notification will not be sent again until the SIGTRAN TPS of Prime Access Registrar server reduces below the configured *carNotifSigtranTPSLowThreshold* value.

carSigtranTPSCapacityNotFull

carSigtranTPSCapacityNotFull signifies that the SIGTRAN TPS of the Prime Access Registrar server has reached the configured low threshold capacity. This trap has the following objects:

- *carNotifSigtranTPSHighThreshold*—indicates the maximum limit of the SIGTRAN TPS of the Prime Access Registrar server.
- *carNotifSigtranTPSLowThreshold*—indicates the minimum limit of the SIGTRAN TPS of the Prime Access Registrar server.
- *carServerSigtranTPSUsage*—indicates the current SIGTRAN TPS usage of the Prime Access Registrar server.

After this notification is sent, this type of notification will not be sent again until the SIGTRAN TPS of Prime Access Registrar server increases beyond the configured *carNotifSigtranTPSHighThreshold* value.

carSessionCapacityFull

carSessionCapacityFull signifies that the session TPS of the Prime Access Registrar server has reached the configured high threshold capacity. This trap has the following objects:

- *carNotifSessionHighThreshold*—indicates the maximum limit of the session TPS of the Prime Access Registrar server.

- *carNotifSessionLowThreshold*—indicates the minimum limit of the session TPS of the Prime Access Registrar server.
- *carServerSessionUsage*—indicates the current session TPS usage of the Prime Access Registrar server.

After this notification is sent, this type of notification will not be sent again until the session TPS of Prime Access Registrar server reduces below the configured *carNotifSessionLowThreshold* value.

carSessionCapacityNotFull

carSessionCapacityNotFull signifies that the session TPS of the Prime Access Registrar server has reached the configured low threshold capacity. This trap has the following objects:

- *carNotifSessionHighThreshold*—indicates the maximum limit of the session TPS of the Prime Access Registrar server.
- *carNotifSessionLowThreshold*—indicates the minimum limit of the session TPS of the Prime Access Registrar server.
- *carServerSessionUsage*—indicates the current session TPS usage of the Prime Access Registrar server.

After this notification is sent, this type of notification will not be sent again until the session TPS of Prime Access Registrar server increases beyond the configured *carNotifSessionHighThreshold* value.

carSigtranSessionCapacityFull

carSigtranSessionCapacityFull signifies that the SIGTRAN session TPS of the Prime Access Registrar server has reached the configured high threshold capacity. This trap has the following objects:

- *carNotifSigtranSessionHighThreshold*—indicates the maximum limit of the SIGTRAN session TPS of the Prime Access Registrar server.
- *carNotifSigtranSessionLowThreshold*—indicates the minimum limit of the SIGTRAN session TPS of the Prime Access Registrar server.
- *carServerSessionUsage*—indicates the current SIGTRAN session TPS usage of the Prime Access Registrar server.

After this notification is sent, this type of notification will not be sent again until the SIGTRAN session TPS of Prime Access Registrar server reduces below the configured *carNotifSigtranSessionLowThreshold* value.

carSigtranSessionCapacityNotFull

carSigtranSessionCapacityNotFull signifies that the SIGTRAN session TPS of the Prime Access Registrar server has reached the configured low threshold capacity. This trap has the following objects:

- *carNotifSigtranSessionHighThreshold*—indicates the maximum limit of the SIGTRAN session TPS of the Prime Access Registrar server.
- *carNotifSigtranSessionLowThreshold*—indicates the minimum limit of the SIGTRAN session TPS of the Prime Access Registrar server.
- *carServerSessionUsage*—indicates the current SIGTRAN session TPS usage of the Prime Access Registrar server.

After this notification is sent, this type of notification will not be sent again until the SIGTRAN session TPS of Prime Access Registrar server increases beyond the configured *carNotifSigtranSessionHighThreshold* value.

carLicenseUsageReset

carLicenseUsageReset signifies that the server usage is nominal after exceeding the license thresholds. This notification carries the percentage of License Usage.

carSigtranLicenseUsageReset

carSigtranLicenseUsageReset signifies that server SIGTRAN usage is nominal after exceeding the license thresholds. This notification carries the percentage of SIGTRAN License Usage.

carReplicationSyncFailure

carReplicationSyncFailure notifies that there is a synchronization failure in Prime Access Registrar replication. This notification is triggered when there is a failure in sync message exchanges or upon out of sync configuration detection. This trap has four objects:

- *carNotifReplicationMasterInetAddrType*—indicates the type of Internet address of the Master, which could be IPv4 address, IPv6 address, or DNS domain name.
- *carNotifReplicationMasterIPAddress*—indicates the IP address of the Master referred to using the version-neutral IP address.
- carNotifReplicationMemberIPAddress—indicates the type of Internet address of the Member, which could be IPv4 address, IPv6 address, or DNS domain name.
- carNotifReplicationMemberInetAddrType—indicates the IP address of the Member referred to using the version-neutral IP address.

TLSClientConnectionUpTrap

TLSClientConnectionUpTrap is sent from Prime Access Registrar when first connection is established by a RADIUS-TLS client.

TLSClientConnectionClosedTrap

TLSClientConnectionClosedTrap is sent from Prime Access Registrar when all the connections are closed by the RADIUS-TLS client.

carReplicationSuccess

carReplicationSuccess notifies that replication synchronization, which had formerly been in a down state is now resolved. This trap has four objects:

- *carNotifReplicationMasterInetAddrType*—indicates the type of Internet address of the Master, which could be IPv4 address, IPv6 address, or DNS domain name.
- *carNotifReplicationMasterIPAddress*—indicates the IP address of the Master referred to using the version-neutral IP address.

- carNotifReplicationMemberIPAddress—indicates the type of Internet address of the Member, which could be IPv4 address, IPv6 address, or DNS domain name.
- carNotifReplicationMemberInetAddrType—indicates the IP address of the Member referred to using the version-neutral IP address.

Configuring Traps

The Prime Access Registrar SNMP implementation uses various configuration files to configure its applications.

This section contains the following topics:

- SNMP Configuration
- Configuring Trap Recipient

SNMP Configuration

A sample configuration file is available in **/cisco-ar/ucd-snmp/share/snmp/snmpd.conf**. This configuration file is used to configure SNMP query permissions and trap recipients.

Configuring Trap Recipient

The following example shows the default configuration that sets up trap recipients for SNMP versions v1 and v2c.

Note Most sites use a single NMS, not two as shown below.

```
# --------------------------------------------------------------------------
trapcommunity trapcom
trapsink zubat trapcom 162
trap2sink ponyta trapcom 162
##############################################################################
```

Note **trapsink** is used in SNMP version 1; **trap2sink** is used in SNMP version 2.

trapcommunity defines the default community string to be used when sending traps. This command must appear prior to **trapsink** or **trap2sink** which use this community string.

trapsink and **trap2sink** are defined as follows:

trapsink hostname community port

trap2sink hostname community port

Community String

A community string is used to authenticate the trap message sender (SNMP agent) to the trap recipient (SNMP management station). A community string is required in the list of trap receivers.

SNMP Version 3 Support

The SNMP Version 3 (SNMPv3) feature provides secure access to devices by authenticating and encrypting data packets over the network. SNMPv3 is an inter-operable, standards-based protocol. SNMPv3 is a security model in which an authentication strategy is set up for a user and the group in which the user resides. Security level is the permitted level of security within a security model. A combination of a security model and a security level determines which security mechanism is used when handling an SNMP packet.

The security features provided in SNMPv3 are:

- Message integrity—Ensures that a packet has not been tampered with during transit.
- Authentication—Determines that the message is from a valid source.
- Encryption—Scrambles the content of a packet to prevent it from being learned by an unauthorized source.

Table 15-1 lists the security levels supported by SNMPv3.

Table 15-1 Security Levels Supported by SNMPv3

Security Level	Authentication Mechanism	Encryption Mechanism	Description
noAuthNoPriv	With Username	None	Uses a username match for authentication.
authNoPriv	Message Digest Algorithm 5 (MD5) or Secure Hash Algorithm (SHA)	None	Provides authentication based on the Hashed Message Authentication Code (HMAC)-MD5 or HMAC-SHA algorithms.
authPriv	MD5 or SHA	Data Encryption Standard (DES) or Advanced Encryption Standard (AES)	Provides authentication based on the HMAC-MD5 or HMAC-SHA algorithms. In addition to authentication, provides encryption based on the DES/AES standards.

Configuring SNMPv3 in Prime Access Registrar

This topic contains the following sections:

- Prerequisites, page 15-13
- Creating Secure User for SNMP Query, page 15-14
- Configuring SNMPv3 Traps, page 15-14

Prerequisites

1. You must enable SNMP agent capability in Prime Access Registrar. To do so:

Log into the CLI. In SNMP object defined as /radius/advanced/snmp, set Enabled to true as shown below:

```
--> cd /radius/advanced/snmp/
[//localhost/Radius/Advanced/SNMP ]
Enabled = TRUE
TracingEnabled = FALSE
InputQueueHighThreshold = 90
InputQueueLowThreshold = 60
```

```
MasterAgentEnabled = TRUE
```

2. Make required changes to the snmpd.conf file located in /cisco-ar/ucd-snmp/share/snmp. After any change to snmpd.conf, restart the Prime Access Registrar server for the changes to take effect.

Creating Secure User for SNMP Query

To use SNMPv3, you must define users with the prescribed security level and encryption methods in the following directory:

/cisco-ar/ucd-snmp/share/snmp/snmpd.conf

Table 15-2 provides sample commands for user creation.

Table 15-2 User Creation in SNMPv3

Command Syntax	Sample Command	Description
createUser *<username> <authentication mechanism> <authentication password> <encryption mechanism> <encryption password>* **rouser** <username> **Note** We can use a combination of Authentication Algorithms (SHA and MD5) and Encryption algorithms (DES and AES) for creating user.	**createUser** securev3user SHA "snmpv3authPass" DES "snmpv3encPass" **rouser** securev3user	Creates a user ***securev3user*** using SHA authentication mechanism and DES encryption protocol. This command is applicable for creating users with authPriv and authNoPriv security levels.
	createUser newuser **rouser** newuser noauth	Creates a user ***newuser*** with ***noauth*** privilege level. This command is applicable for creating users with noAuthNoPriv security level.

After modifying snmpd.conf file, ensure that you restart the Prime Access Registrar server for the changes to take effect.

Configuring SNMPv3 Traps

When using SNMPv3, you must define the trap session command in the snmpd.conf file. Using this command, the required security definitions can be achieved as explained in Table 15-3.

Note For receiving query responses and traps on the NMS, the NMS server must be configured corresponding to the definitions and configurations in snmpd.conf file of Prime Access Registrar.

Table 15-3 Trap Session Commands in SNMPv3

Security Level	Description	Sample Trap Session Command	Command Action
noAuthNoPriv	Authentication using username and no encryption	`trapsess -r 0 -v 3 -u snmpv3user -n "" -l noAuthNoPriv 10.10.10.11 162`	Instructs SNMP agent to send traps to snmpv3 user to the defined NMS address

authNoPriv	Authentication using MD5 or SHA and no encryption	`trapsess -r 0 -v 3 -u snmpv3user -n "" -l authNoPriv -a SHA -A snmpv3authPass -x AES -X snmpv3encPass 10.10.10.11 162`
		
		`(or)`
		
		`trapsess -r 0 -v 3 -u snmpv3user -n "" -l authNoPriv -a SHA -A snmpv3authPass  127.0.0.1 162`
			Instructs SNMP agent to send traps to snmpv3 user using SHA to the defined NMS address

authPriv	Authentication using MD5 or SHA and encryption using DES/AES	`trapsess -r 0 -v 3 -u snmpv3user -n "" -l authPriv -a SHA -A snmpv3authPass -x AES -X snmpv3encPass 10.10.10.11 162`	Instructs SNMP agent to send traps to snmpv3 user using SHA and AES algorithms to the defined NMS address

After modifying snmpd.conf file, ensure that you restart the Prime Access Registrar server for the changes to take effect.