CHAPTER 5

Managing the System

This chapter describes the basic tasks that you can do to manage the general system (or nonprotocol-specific) features. Our system management features are supported via the Simple Network Management Protocol (SNMP). This chapter describes the tasks needed to configure SNMP support on the communication server. A part of SNMP is the Management Information Base (MIB). MIBs provide variables that can be set or read to change parameters or provide information on network devices and interfaces. Cisco supports several MIBs, including the Internet standard MIB II, and also provides its own Cisco MIB. For information on the Cisco MIB, see the *Cisco Management Information Base (MIB) User Quick Reference* publication.

For a list of recommended books on network management, refer to the appendix “References and Recommended Reading” in the *Access and Communication Servers Command Reference* publication.

For a complete description of the commands mentioned in this chapter, refer to the “System Management Commands” chapter of the *Access and Communication Servers Command Reference* publication.

Understanding System Management

This chapter describes how to manage the communication server and its performance on the network. In general, system or network management falls into the categories described in the following sections:

- Configuration Management

  The configuration of network devices determines the network’s behavior. To manage device configurations, you need to list and compare configuration files on running devices, store configuration files on network servers for shared access, and perform software installations and upgrades. These configuration management tasks are described in the “Loading System Images and Configuration Files” chapter.

  Other configuration management tasks include naming the communication server, setting communication server time services, and configuring SNMP support. These tasks are described in this chapter.

- Security Management

  To manage security on the network, you need to restrict access to the system. You can do so on several different levels:

  — Assign passwords (and encrypt them) to restrict access to communication server terminal lines, login connections, or privileged EXEC mode.

    - Establish one of three versions of Terminal Access Controller Access Control System (TACACS) protection for network servers that have shared access: TACACS, XTACACS, or TACACS+, which is coupled with the Authentication, Authorization, and Accounting (AAA) model.
    - Restrict login connections to specific users with a username authentication system.
    - Control access on serial interfaces with Challenge Handshake Authentication Protocol (CHAP) and Password Authentication Protocol (PAP).
    - Create access lists to filter traffic to and from specific destinations. Subsequent chapters that describe the routing protocols in detail define access lists. This section provides general guidelines for creating access lists.
    - Create security labels for Internet Protocol (IP) datagrams using the Internet Protocol Security Option (IPSO), as described in the “Configuring IP” chapter.

- Fault Management

  To manage network faults, you need to discover, isolate, and fix the problems. You can discover problems with the system’s monitoring commands, isolate problems with the system’s test commands, and resolve problems with other commands, including **debug**.

  This section introduces basic fault management commands. For detailed troubleshooting procedures and a variety of scenarios, see the *Troubleshooting Internetworking Systems* publication. For complete details on all **debug** commands, see the *Debug Command Reference* publication.

- System Performance Management

  To manage system performance, you need to monitor and determine response time, error rates, and availability. Once these factors are determined, you can perform load balancing and modify system parameters to enhance performance. For example, priority queuing allows you to prioritize traffic order. You can configure fast and autonomous switching to improve network throughput, as described in the “Configuring Interfaces” chapter of this manual.

  See the *Internetwork Design Guide* for additional information.

- Accounting Management

  Accounting management allows you to track both individual and group usage of network resources. You can then reallocate resources as needed. For example, you can change the system timers and configure TCP keepalives. See also the IP accounting feature in the “Configuring IP” chapter of this manual. Additionally, the AAA/TACACS+ **aaa accounting** command allows you to set start/stop accounting for any or all of the listed functions for this command.

See “System Management Examples” at the end of this chapter.

Configuration Management

You can complete any of the tasks in the following sections to perform configuration management functions:

- Customize the Communication Server Prompt
- Set the Communication Server Name
- Create and Monitor Command Aliases
- Set the Interval for Load Data
- Set the Communication Server Time Services

- Monitor Time Services
- Configure Synchronization of Logging Messages
- Configure SNMP Support
- Configure the Cisco Discovery Protocol
- Generate a Downward-compatible Configuration

Other configuration management tasks are described in the chapter entitled “Loading System Images and Configuration Files.”

Customize the Communication Server Prompt

By default, the communication server prompt consists of the communication server name followed by an angle bracket (>) for EXEC mode or a pound sign (#) for privileged EXEC mode. To customize your communication server prompt, perform the following task in global configuration mode:

Task	Command
Customize the communication server prompt.	**prompt** *string*

Set the Communication Server Name

One of the first basic tasks is to name your communication server. The name of the communication server is considered the host name and is the name that is displayed by the system prompt. If no name is configured, the system default communication server name is *cs*. You can name the communication server while in global configuration mode as follows:

Task	Command
Set the host name.	**hostname** *name*

Create and Monitor Command Aliases

You can create aliases for commonly used or complex commands. Use word substitutions or abbreviations to tailor command syntax for you and your user community.

To create and display command aliases, perform the tasks in the following sections:

- Create a Command Alias
- Display Command Aliases

Create a Command Alias

To create a command alias, perform the following task in global configuration mode:

Task	Command
Configure a command alias.	**alias** *mode alias-name alias-command-line*

Display Command Aliases

To display alias names and the original command syntax, perform the following task in EXEC mode:

Task	Command
Show all command aliases and original command syntax, or specify the aliases in a particular command mode.	**show aliases** [*mode*]

Set the Interval for Load Data

You can change the length of time that a set of data is used for computing load statistics. By decreasing the load interval, dial backup and other decisions are based on an average that is computed over a shorter length of time and is more responsive to bursts of traffic.

To change this period, perform the following task in interface configuration mode:

Task	Command
Set the length of time that data is used for load calculations.	**load-interval** *seconds*

Set the Communication Server Time Services

All of our communication server products provide an array of time-of-day services. These services allow the products to accurately keep track of the current time and date, to synchronize multiple products to the same time, and to provide time services to other systems.

The heart of the time service is the system clock. This clock runs from the moment the system starts up and keeps track of the current date and time. The system clock can be set from a number of sources, and in turn can be used to distribute the current time through various mechanisms to other systems. When the system is initialized, the system clock is set to midnight on March 1, 1993. The system clock can then be set from the following sources:

- Network Time Protocol (NTP)
- Manual configuration

The system clock can provide time to the following services:

- User **show** commands
- Logging and debugging messages
- NTP

The system clock internally keeps track of time based on Coordinated Universal Time (UTC), also known as Greenwich Mean Time (GMT). You can configure information about the local time zone and summer time (daylight savings time) so that the time is displayed correctly relative to the local time zone.

The system clock keeps track of whether the time is “authoritative” or not (that is, whether it has been set by a time source considered to be authoritative). If it is not authoritative, the time will be available only for display purposes and will not be redistributed.

Configure NTP

The Network Time Protocol (NTP) is a protocol designed to time-synchronize a network of machines. NTP runs over UDP, which in turn runs over IP. NTP is documented in RFC 1305.

An NTP network usually gets its time from an authoritative time source, such as a radio clock or an atomic clock attached to a time server. NTP then distributes this time across the network. NTP is extremely efficient; no more than one packet per minute is necessary to synchronize two machines to within a millisecond of one another.

NTP uses the concept of a “stratum” to describe how many NTP “hops” away a machine is from an authoritative time source. A “stratum 1” time server has a radio or atomic clock directly attached, a “stratum 2” time server receives its time via NTP from a “stratum 1” time server, and so on. A machine running NTP will automatically choose as its time source the machine with the lowest stratum number that it is configured to communicate with via NTP. This strategy effectively builds a self-organizing tree of NTP speakers.

NTP is careful to avoid synchronizing to a machine whose time might not be accurate. It avoids doing so in two ways. First of all, NTP will never synchronize to a machine that is not in turn synchronized itself. Secondly, NTP will compare the time reported by several machines, and will not synchronize to a machine whose time is significantly different than the others, even if its stratum is lower.

The communications between machines running NTP (known as “associations”) are usually statically configured; each machine is given the IP address of all machines with which it should form associations. Accurate timekeeping is made possible by exchanging NTP messages between each pair of machines with an association. However, in a local-area network (LAN) environment, NTP can be configured to use IP broadcast messages instead. This alternative reduces configuration complexity because each machine can simply be configured to send or receive broadcast messages. However, the accuracy of timekeeping is marginally reduced because the information flow is one-way only.

The time kept on a machine is a critical resource, so we strongly recommend that you use the security features of NTP to avoid the accidental or malicious setting of incorrect time. Two mechanisms are available: an access list-based restriction scheme and an encrypted authentication mechanism.

Our implementation of NTP does not support stratum 1 service; in other words, it is not possible to connect a radio or atomic clock to this communication server. It is recommended that time service for your network be derived from the public NTP servers available in the Internet. If the network is isolated from the Internet, our implementation of NTP allows a machine to be configured so that it acts as though it is synchronized via NTP, when in fact it has determined the time using other means. Other machines will then synchronize to that machine via NTP.

When multiple sources of time are available, NTP is always considered to be more authoritative. NTP time will override the time set by any other method.

A number of manufacturers include NTP software for their host systems, and a publicly available version for systems running UNIX and its various derivatives is also available. This software allows host systems to be time-synchronized as well.

NTP services are enabled on all interfaces by default. You can perform the following tasks:

- Configure NTP authentication.
- Configure NTP associations.
- Configure NTP broadcast services.
- Configure NTP access restrictions.
- Configure the source IP address for NTP packets.
- Configure the system as an authoritative NTP server.

Configure NTP Authentication

You can authenticate the associations with other systems for security purposes. You must enable the NTP authentication feature, then define each of the authentication keys. Each key has a key number, a type, and a value. Currently the only key type supported is **md5**. Finally, define a list of trusted authentication keys. If a key is trusted, then this system will be willing to synchronize to a system that uses this key in its NTP packets.

To configure NTP authentication, perform the following tasks in global configuration mode:

Task		Command
Step 1	Enable the NTP authentication feature.	**ntp authenticate**
Step 2	Define the authentication keys.	**ntp authentication-key** *number* **md5** *value*
Step 3	Define trusted authentication keys.	**ntp trusted-key** *key-number*

Configure NTP Associations

An NTP association can be a peer association (meaning that this system is willing to either synchronize to the other system or to allow the other system to synchronize to it), or it can be a server association (meaning that this system will only synchronize to the other system, and not the other way around). If you want to form an NTP association with another system, perform one of the following tasks in global configuration mode:

Task	Command
Form a peer association with another system.	**ntp peer** *ip-address* [**version** *number*] [**key** *keyid*] [**source** *interface*] [**prefer**]
Form a server association with another system.	**ntp server** *ip-address* [**version** *number*] [**key** *keyid*] [**source** *interface*] [**prefer**]

Note that only one end of an association needs to be configured; the other system will automatically establish the association.

Configure NTP Broadcast Service

The system can either send broadcast packets or listen to them on an interface-by-interface basis. The estimated round-trip delay for broadcast packets can also be configured.

Perform the following task in global configuration mode to adjust the NTP broadcast delay period:

Task	Command
Adjust estimated delay.	**ntp broadcastdelay** *microseconds*

To configure your system to either send or receive NTP broadcast packets, perform the following tasks in interface configuration mode:

Task	Command
Send NTP broadcast packets.	**ntp broadcast** [**version** *number*]
Receive NTP broadcast packets.	**ntp broadcast client**

Configure NTP Access Restrictions

You can control NTP access on two levels by completing the following steps:

Step 1 Create an access group and assign a basic IP access list to it.

Step 2 Disable NTP services on a special interface.

To control access to NTP services, you can create an NTP access group and apply a basic IP access list to it. To do so, perform the following task in global configuration mode:

Task	Command
Create an access group and apply a basic IP access list to it.	**ntp access-group** {**query-only** | **serve-only** | **serve** | **peer**} *access-list-number*

The access group options are scanned in the following order from least restrictive to most restrictive:

1. Peer—Allows time requests and NTP control queries and allows the system to synchronize itself to a system whose address passes the access list criteria.
2. Serve—Allows time requests and NTP control queries, but does not allow the system to synchronize itself to a system whose address passes the access list criteria.
3. Serve-only—Allows only time requests from a system whose address passes the access list criteria.
4. Query-only—Allows only NTP control queries from a system whose address passes the access list criteria.

If the source IP address matches the access lists for more than one access type, the first type is granted. If no access groups are specified, all access types are granted to all systems. If any access groups are specified, only the specified access types will be granted.

For details on NTP control queries, see RFC 1305 (NTP version 3).

NTP services are enabled on all interfaces by default. You can disable NTP packets from being received through an interface by performing the following task in interface configuration mode:

Task	Command
Disable NTP services on a specific interface.	**ntp disable**

Configure the Source IP Address for NTP Packets

When the system sends an NTP packet, the source IP address is normally set to the address of the interface through which the NTP packet is sent. Perform the following task in global configuration mode if you want to configure a specific interface from which the IP source address will be taken:

Task	Command
Configure an interface from which the IP source address will be taken.	**ntp source** *interface*

The interface will be used for the source address for all packets sent to all destinations. If a source address is to be used for a specific association, use the **source** keyword with the **ntp peer** or **ntp server** command shown earlier in this chapter.

Configure the System as an Authoritative NTP Server

Perform the following task in global configuration mode if you want the system to be an authoritative NTP server, even if the system is not synchronized to an outside time source:

Task	Command
Make the system an authoritative NTP server.	**ntp master** [*stratum*]

Caution Use this command with extreme caution. It is very easy to override valid time sources using this command, especially if a low stratum number is configured. Configuring multiple machines in the same network with the **ntp master** command can cause instability in timekeeping if the machines do not agree on the time.

Configure Time and Date Manually

If no other source of time is available, you can manually configure the current time and date after the system is restarted. The time will remain accurate until the next system restart. We recommend that you use manual configuration only as a last resort.

To set up time services, complete the following tasks. If you have an outside source to which the communication server can synchronize, you do not need to manually set the system clock.

- Configure the time zone.
- Configure summer time (daylight saving time) if applicable.
- Set the system clock (if no other time source is available).

Configure the Time Zone

Complete the following task in global configuration mode to manually configure the time zone used by the communication server:

Task	Command
Set the communication server time zone.	**clock timezone** *zone hours* [*minutes*]

Configure Summer Time

To configure summer time (daylight saving time) in areas where it starts and ends on a particular day of the week each year, perform the following task in global configuration mode:

Task	Command
Configure summer time.	**clock summer-time** *zone* **recurring** [*week day month hh:mm week day month hh:mm* [*offset*]]

If summer time in your area does not follow this pattern, you can configure the exact date and time of the next summer time events by performing one of the following tasks in global configuration mode:

Task	Command
Configure summer time.	**clock summer-time** *zone* **date** *month date year hh:mm month date year hh:mm* [*offset*] or **clock summer-time** *zone* **date** *date month year hh:mm date month year hh:mm* [*offset*]

Set the System Clock

If you have an outside source on the network that provides time services (such as an NTP server or VINES time service), you do not need to manually set the system clock.

However, if you have do not have any time service source, complete one of the following tasks in EXEC mode to set the system clock:

Task	Command
Set the system clock.	**clock set** *hh:mm:ss day month year* or **clock set** *hh:mm:ss month day year*

Monitor Time Services

You can monitor clock and NTP EXEC services by completing the following tasks in EXEC mode:

Task	Command
Display the current system clock time.	**show clock** [**detail**]
Show the status of NTP associations.	**show ntp associations [detail]**
Show the status of NTP.	**show ntp status**

Configure Synchronization of Logging Messages

You can configure the system to synchronize unsolicited messages and debug output with solicited communication server output and prompts for a specific line. You can identify the types of messages to be output asynchronously based on the level of severity. You can also determine the maximum number of buffers for storing asynchronous messages for the terminal after which messages are dropped.

When synchronous logging of unsolicited messages and debug output is turned on, unsolicited output is displayed on the console or printed after solicited output is displayed or printed. Unsolicited messages and debug output is displayed on the console after the prompt for user input is returned. This is to keep unsolicited messages and debug output from being interspersed with solicited output and prompts. After the unsolicited messages are displayed, the console displays the user prompt again.

To configure for synchronous logging of unsolicited messages and debug output with solicited output and prompts, perform the following task in line configuration mode:

Task	Command
Enable synchronous logging of messages.	**logging synchronous** [**level** *severity-level-number* | **all**] [**limit** *maximum-number-of-buffers*]

Configure SNMP Support

The Simple Network Management Protocol (SNMP) system consists of three parts: an SNMP manager, an SNMP agent, and a Management Information Base (MIB). SNMP is an application-layer protocol that allows SNMP manager and agent stations to communicate. SNMP provides a message format for sending information between an SNMP manager and an SNMP agent. The SNMP manager can be part of a Network Management System (NMS), such as CiscoWorks.

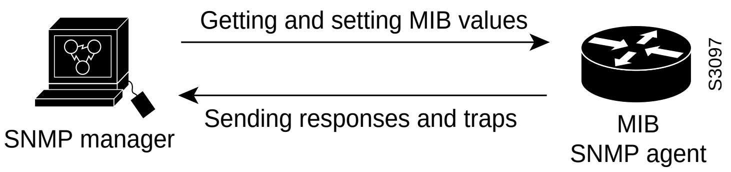
The agent and MIB reside on the communication server. In configuring SNMP on the communication server, you define the relationship between the manager and the agent.

The SNMP agent contains MIB variables whose values the SNMP manager can request or change. A manager can get a value from an agent or store a value into that agent. The agent gathers data from the MIB, the repository for information about device parameters and network data. The agent can also respond to a manager's requests to get or set data.

An agent can also send unsolicited traps to the manager. Traps are messages alerting the SNMP manager to a condition on the network. Traps can indicate improper user authentication, restarts, link status (up or down), closing of a TCP connection, or loss of connection to a neighbor communication server.

Figure 5-1 illustrates the communications relationship between the SNMP manager and agent. It shows that a manager can send the agent requests to get and set MIB values. The agent can respond to these requests. Independent of this interaction, the agent can send unsolicited traps to the manager notifying the manager of network conditions.

Figure 5-1 Communication between an SNMP Agent and Manager



Cisco supports the SNMP Version 1 protocol, referred to as SNMPv1, and the SNMP Version 2 protocol, referred to as SNMPv2. Our implementation of SNMP supports all MIB II variables (as described in RFC 1213) and SNMP traps (as described in RFC 1215). Cisco also supports the definition of management information described in RFCs 1155, 1157, and 1213, and supports some or all variables in the MIBs described in the following RFCs: 1156, 1212, 1231, 1243, 1253, 1285, 1286, 1315, 1381, 1382, 1398, 1447, 1450, 1512, 1516, and 1285 (FDDI).

RFC 1447, “SNMP v.2 Party MIB” (April 1993) describes the managed objects that correspond to the properties associated with SNMP v.2 parties, SNMP v.2 contexts, and access control policies, as defined by the SNMP v.2 Administrative Model. RFC 1450, “SNMPv 2 MIB,” (April 1993) describes the managed objects that instrument the behavior of an SNMP v.2 implementation. Cisco supports the MIB variables as required by the conformance clauses specified in these MIBs.

Cisco also provides its own MIB with every system. The Cisco MIB provides a new chassis MIB variable that enables the SNMP manager to gather data on system card descriptions, serial numbers, hardware and software revision levels, and slot locations.

See the *Cisco Management Information Base (MIB) User Quick Reference* for a detailed description of each Cisco MIB variable and SNMP trap.

Although SNMP v.2 offers more robust support than SNMP v.1, Cisco continues to support SNMP v.1. This is because not all management stations have migrated to SNMP v.2 and you must configure the relationship between the agent and the manager to use the version of SNMP supported by the management station.

SNMP v.1 offers a community-based form of security defined through an IP address access control list and password. SNMP v.2 offers richer security configured through an access policy that defines the relationship between a single manager and agent. SNMP v.2 security includes message authentication support using the Message Digest (MD5) algorithm, but because of the Data Encryption Standard (DES) export restrictions, it does not include encryption support through DES. SNMP v.2 security provides data origin authentication, ensures data integrity, and protects against message stream modification.

In addition to enhanced security, SNMP v.2 support includes a bulk retrieval mechanism and more detailed error message reporting to management stations. The bulk retrieval mechanism supports the retrieval of tables and large quantities of information, minimizing the number of round trips required.

The SNMP v.2 improved error handling support includes expanded error codes that distinguish different kinds of error conditions; these conditions are reported through a single error code in SNMP v.1. Error return codes now report the error type. Three kinds of exceptions are also reported: no such object exceptions, no such instance exceptions, and end of MIB view exceptions.

There is no specific command that you use to enable SNMP. The first SNMP-server command that you enter enables both versions of SNMP.

To configure SNMP support, perform the tasks in one of the following sections:

- Configure for Both SNMP v.2 and SNMP v.1
- Configure SNMP v.2 Support
- Configure SNMP v.1 Support

To configure the relationship between the agent and the manager on the communication server, you need to know the version of the SNMP protocol that the management station supports. An agent can communicate with multiple managers; for this reason, you can configure the communication server to support communications with one management station using the SNMP v.1 protocol and another using the SNMP v.2 protocol.

Configure for Both SNMP v.2 and SNMP v.1

You can perform the following tasks to configure support for both SNMP v.2 and SNMP v.1 on the communication server:

- Enable the SNMP Agent Shutdown Mechanism

- Establish the Contact, Location, and Serial Number of the SNMP Agent
- Define the Maximum SNMP Packet Size
- Monitor SNMP Status
- Disable the SNMP Agent

Enable the SNMP Agent Shutdown Mechanism

Using SNMP packets, a network management tool can send messages to users on virtual terminals and the console. This facility operates in a similar fashion to the EXEC **send** command; however, the SNMP request that causes the message to be issued to the users also specifies the action to be taken after the message is delivered. One possible action is a shutdown request. Because the ability to cause a reload from the network is a powerful—and potentially dangerous—feature, you can protect the system from network reloads by enabling the SNMP agent shutdown mechanism. Perform the following task in global configuration mode:

Task	Command
Use the SNMP message reload feature and request a system shutdown message.	**snmp-server system-shutdown**

To understand how to use this feature with SNMP requests, read the document *mib.txt* available by anonymous FTP from *ftp.cisco.com.*

Establish the Contact, Location, and Serial Number of the SNMP Agent

You can set the system contact, location, and serial number of the SNMP agent so that these descriptions can be accessed through the configuration file. To do so, perform one or more of the following tasks in global configuration mode:

Task	Command
Set the system contact string.	**snmp-server contact** *text*
Set the system location string.	**snmp-server location** *text*
Set the system serial number.	**snmp-server chassis-id** *text*

Define the Maximum SNMP Packet Size

You can set the maximum packet size permitted when the SNMP agent is receiving a request or generating a reply. To do so, perform the following task in global configuration mode:

Task	Command
Establish the maximum packet size.	**snmp-server packetsize** *byte-count*

Monitor SNMP Status

To monitor SNMP input and output statistics, including the number of illegal community string entries, errors, and requested variables, complete the following task in EXEC mode:

Task	Command
Monitor SNMP status.	**show snmp**

Disable the SNMP Agent

To disable both versions of SNMP (SNMP v.1 and SNMP v.2) concurrently, perform the following task in global configuration mode:

Task	Command
Disable SNMP agent operation.	**no snmp-server**

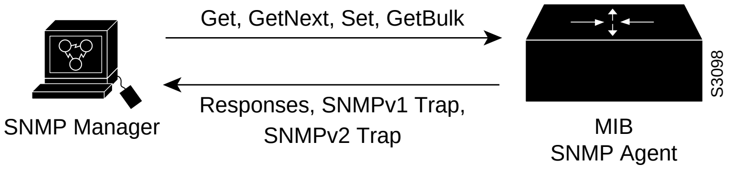
Configure SNMP v.2 Support

SNMP v.2 security requires that you create an access policy that defines the relationship between a manager and the agent. For each management station that the agent communicates with, you must create a separate access policy. Creating an access policy is a multiple-task process:

1. Define a view to identify the objects that can be seen, if you do not want to use one of the standard predefined views.
2. Define a context to identify the object resources that can be acted on.
3. Define a party for both the manager and the agent to identify them.
4. Using the definitions created in the previous steps, configure the access policy that characterizes the communications that can occur between the manager and the agent. The privileges that you define for the access policy depend on whether the agent is defined as the source or the destination. For example,
   - When the agent party is defined as the destination in an access policy, the access policy privileges define the management operations that the agent will accept from the manager and perform in relation to the object resources.
   - When the agent party is defined as the source in an access policy, the access policy privileges define the responses and traps that the agent can send to the manager.

Figure 5-2 shows the information exchanged between the manager and the agent. The top arrow, leading from the manager to the agent, shows the types of requests the manager can send to the agent. The bottom arrow, leading from the agent to the manager, shows the kind of information that the agent can send to the manager. Note that the agent sends trap messages to the manager in response to certain network conditions; trap messages are unsolicited and are not related to the request/response communication exchange between the manager and the agent that occurs in relation to MIB variables. For any given manager and agent relationship, the privileges defined in the access policy constrain communications to a specific set of operations.

Figure 5-2 Flow of Management Requests, Responses, and Traps



You must create access policies for each new agent that is installed. You also must create access policies on an agent when new management stations with which the agent will communicate are installed. Moreover, every time a network address changes on a management station, you must reconfigure the access policy to reflect the new information for the management station.

Because the process of creating an access policy is complex and must be performed many times, SNMP v.2 offers a single-step method that relies on an accepted convention called the *simplified security conventions*. You can configure security using this simplified method only if both the agent and the manager support it and consent to use it. The simplified method offers ease of use, but at the cost of forfeiting control over some values that can be configured if you create an access policy.

If you use the simplified security conventions method, the SNMP v.2 implementation assumes default values that it determines internally for required information that you cannot provide through the command interface. To use the simplified method, you enter one command supplying a user ID, and optionally, the name of a view, access rights, and a password. The SNMP v.2 implementation on the communication server derives most of the configuration information from other values.

This section describes each task that you must perform to configure an access policy. Then it addresses the alternative method and describes the task of configuring the user ID for the simplified security conventions method.

You can perform the following tasks to configure support for SNMP v.2 on the communication server:

- Configure or modify an SNMP view record.
- Configure or modify an SNMP context record.
- Configure or modify an SNMP party record.
- Configure or modify an SNMP access policy.
- Configure or modify an SNMP v.2 simplified security context record.
- Configure SNMP v.2 trap operations.

Configure an SNMP View Record

To create or update an SNMP view record, perform the following task in global configuration mode:

Task	Command
Create or modify a view record.	**snmp-server view** *view-name oid-tree* {**included** | **excluded**} [**volatile**]

To remove a view record, use the **no snmp-server view** command.

Configure an SNMP Context Record

To create or update an SNMP context record, perform the following task in global configuration mode:

Task	Command
Create or modify a context record.	**snmp-server context** *context-name context-oid view-name* [**volatile**]

To remove a context entry, use the **no snmp-server context** command. Specify only the name of the context. The name identifies the context to be deleted.

Configure an SNMP v.2 Party Record

To create or update an SNMP v.2 party record, perform the following task in global configuration mode:

Task	Command
Create or modify a party record.	**snmp-server party** *party-name party-oid* [*protocol-address*] [**packetsize** *size*] **[local** | **remote**] [**authentication** {**md5** *key* [**clock** *clock*] [**lifetime** *lifetime*]}] [**volatile**]

To remove a party record, use the **no snmp-server party** command.

Configure an SNMP v.2 Access Policy

To define or modify an SNMP v.2 access policy, perform the following task in global configuration mode:

Task	Command
Create an access policy.	**snmp-server access-policy** *destination-party source-party context privileges* [**volatile**]

To remove an SNMP v.2 access-policy, use the **no snmp-server access-policy** command.

Configure an SNMP v.2 Simplified Security Context Record

To create or update a simplified security context record, perform the following task in global configuration mode:

Task	Command
Create or modify a context record.	**snmp-server userid** *user-id* [**view** *view-name*] [**RO** | **RW**] [**password** *password*]

To remove a simplified security context record, use the **no snmp-server userid** command.

Note You may choose to use the same user ID and password across several machines. Because other values are derived internally from the agent’s IP address, these configurations are unique.

Define SNMP v.2 Trap Operations

A trap is an unsolicited message sent by an SNMP agent to an SNMP manager indicating that some event has occurred. The SNMP trap operations allow you to configure the communication server to send information to a network management application when a particular event occurs. You can specify the following features for SNMP v.2 agent trap operations:

- Source interface
- Recipient
- Trap message authentication
- Retransmission interval

- Message (packet) queue length for each trap host

To define the recipient of the trap message, you configure a party record for the manager, including the protocol address, and specify the party record as the destination party for the **snmp-server access policy** command. To define traps for the agent to send to the manager, perform one or more of the following tasks in global configuration mode:

Task	Command
Specify the source interface (and hence IP address) of the trap message.	**snmp-server trap-source** *interface*
Specify the access policy that defines the traps that the agent can send to the manager.	**snmp-server access-policy** *destination-party source-party context privileges* [**volatile**]
Establish trap message authentication.	**snmp-server trap-authentication** [**snmp1** | **snmp2**]
Define how often to resend trap messages on the retransmission queue.	**snmp-server trap-timeout** *seconds*
Establish the message queue length for each trap host.	**snmp-server queue-length** *length*

Configure SNMP v.1 Support

If the manager supports only the SNMP v.1 protocol, you must configure the relationship between the manager and the agent using SNMP v.1 support. You can use either of two methods to configure access to the agent. There are trade-offs involved in choosing one method over the other. The methods differ in the following ways:

- Using the **snmp-server community** command, you specify a string, and, optionally, an access list. The string is used as a password. The access list identifies the IP addresses of systems on which SNMP v.1 managers reside that may use the community string to gain access to the SNMP v.1 agent. You cannot restrict the MIB view using this method.
- Using an access policy, you can specify a password-like string and you can impose a restricted MIB view, but you cannot specify an access list to identify the IP addresses of managers that may access the agent. An SNMP v.1 access policy is similar to an SNMP v.2 access policy.

You can perform the following tasks to configure support for SNMP v.1 on the communication server:

- Configure access control for an SNMP v.1 community
- Configure a view to be used for a context record
- Configure a context to be used for a party record
- Configure an SNMP v.1 party record to be used for an access policy
- Configure an SNMP v.1 access policy
- Define SNMP v.1 trap operations

These tasks are described in the following sections

Configure Access Control for an SNMP v.1 Community

You can configure a community string, which acts like a password, to permit access to the agent on the communication server. Optionally, you can associate a list of IP addresses with that community string to permit only managers with these IP addresses to use the string.

To configure a community string, perform the following task in global configuration mode:

Task	Command
Define the community access string.	**snmp-server community** *string* [**RO** | **RW**] [*access-list number*]

You can configure one or more community strings. To remove a specific community string, use the **no snmp-server community** command.

Configure an SNMP View Record

To create or update an SNMP view record, perform the following task in global configuration mode:

Task	Command
Create or modify a view record.	**snmp-server view** *view-name oid-tree* {**included** | **excluded**} [**volatile**]

To remove a view record, use the **no snmp-server view** command.

Configure an SNMP Context Record

To create or update an SNMP context record, perform the following task in global configuration mode:

Task	Command
Create or modify a context record.	**snmp-server context** *context-name context-oid view-name* [**volatile**]

To remove a context entry, use the **no snmp-server context** command. Specify only the name of the context. The name identifies the context to be deleted.

Configure a Party Record

To create or update an SNMP v.1 party record to be used in an access policy, perform the following task in global configuration mode:

Task	Command
Create or modify a party record.	**snmp-server party** *party-name party-oid* [*protocol-address*] [**packetsize** *size*] [**local** | **remote**] [**authentication** {**snmpv1** *string*}] [**volatile**]

To remove a party record, use the **no snmp-server party** command.

Configure an SNMP Access Policy

To configure an access policy, you specify the SNMP v.1 proxy for which you configured the party record as both the destination party and the source party. To configure an access policy, perform the following task in global configuration mode:

Task	Command
Create an access policy.	**snmp-server access-policy** *destination-party source-party context privileges* [**volatile**]

To remove an SNMP access-policy, use the **no snmp-server access-policy** command.

Define SNMP Trap Operations for SNMP v.1

The SNMP trap operations allow a system administrator to configure the agent communication server to send information to a manager when a particular event occurs. You can specify the following features for SNMP server trap operations:

- Source interface
- Recipient
- Trap message authentication
- Retransmission interval
- Define the message (packet) queue length for each trap host

Perform the following tasks in global configuration mode to define traps for the agent to send to the specified manager:

Task	Command
Specify the source interface (and hence IP address) of the trap message.	**snmp-server trap-source** *interface*
Specify the recipient of the trap message.	**snmp-server host** *address community-string* [**snmp**] [**tty**]
Establish trap message authentication.	**snmp-server trap-authentication** [**snmp1** | **snmp2**]
Define how often to resend trap messages on the retransmission queue.	**snmp-server trap-timeout** *seconds*
Establish the message queue length for each trap host.	**snmp-server queue-length** *length*

Configure the Cisco Discovery Protocol

The Cisco Discovery Protocol (CDP) is a media- and protocol-independent protocol that runs on all Cisco-manufactured equipment including routers, bridges, communication servers, and switches. With CDP, network management applications can learn the device type and the SNMP- agent address of neighboring devices. This enables applications to send SNMP queries to neighboring devices.

CDP runs on all media that support SNAP, including LAN, Frame Relay, and ATM media. CDP runs over the data link layer only. Therefore, two systems that support different network layer protocols can learn about each other.

Each device configured for CDP sends periodic messages to a multicast address. Each device advertises at least one address at which it can receive SNMP messages. The advertisements also contain time-to-live, or holdtime, information, which indicates the length of time a receiving device should hold CDP information before discarding it.

There is a CDP MIB for the management of CDP on Cisco devices.

CDP Configuration Task List

CDP is enabled on your communication server by default, which means the communication server will send CDP packets. You can perform the tasks in the following sections to configure CDP:

- Enable CDP on an Interface
- Set CDP Transmission Timer and Hold Time
- Disable CDP for the Communication Server
- Monitor and Maintain CDP

Enable CDP on an Interface

Although CDP is enabled by default on the communication server, you must enable it on each interface in order to receive CDP information. To enable CDP on an interface, perform the following task in interface configuration mode:

Task	Command
Enable CDP on an interface.	**cdp enable**

Set CDP Transmission Timer and Hold Time

To set the frequency of CDP transmissions and the hold time for CDP packets, perform the following tasks in global configuration mode:

Task	Command
Specify frequency of transmission of CDP updates.	**cdp timer** *seconds*
Specify the amount of time a receiving device should hold the information sent by this device before discarding it.	**cdp holdtime** *seconds*

Disable CDP for the Communication Server

CDP is enabled on a communication server by default. To disable CDP on a communication server and reenable it, perform the following task in global configuration mode:

Task	Command
Disable CDP.	**no cdp run**
Enable CDP.	**cdp run**

Monitor and Maintain CDP

To monitor and maintain CDP on your device, perform the following tasks in privileged EXEC mode:

Task	Command
Reset the traffic counters to zero.	**clear cdp counters**
Delete the CDP table of information about neighbors.	**clear cdp table**
Display global information such as frequency of transmissions and the holdtime for packets being transmitted.	**show cdp**
Display information about a specific neighbor. Display can be limited to protocol or version information.	**show cdp entry** *entry-name* [**protocol** | **version**]
Display information about interfaces on which CDP is enabled.	**show cdp interface** [*type number*]
Display information about neighbors. The display can be limited to neighbors on a specific interface, and expanded to provide more detailed information.	**show cdp neighbors** [*interface-type interface-number*] [**detail**]
Display CDP counters, including the number of packets sent and received and checksum errors.	**show cdp traffic**
Display information about the types of debugging that are enabled for your router. See the *Debug Command Reference* for more information about CDP **debug** commands.	**show debugging**

Generate a Downward-compatible Configuration

In Cisco IOS Release 10.3, IP access lists changed format. If you decide to downgrade from a Release 11.0 to Release 10.2, you can configure the software to try to regenerate a configuration in the format of Release 10.2, thereby saving time and making your IP access lists compatible with the software.

To have the software try to regenerate a configuration in the format prior to Release 10.3, perform the following task in global configuration mode:

Task	Command
Generate a backward-compatible configuration.	**downward-compatible-config** *version*

Security Management

To set up security features, you need to identify sensitive information, find the network access points to that information, secure these access points, and maintain the secure access points.

The following sections describe the optional tasks you can use to control access to the system:

- Establish Password Protection
- Configure Multiple Privilege Levels
- Disable Password Protection
- Recover a Lost Enable Password

- Recover a Lost Line Password
- Create Access Lists
- Establish Terminal Access Control
- Enable CHAP
- Enable PAP
- Enable IP Accounting and Display IP Access Violations

Other chapters in this guide provide information on protocol-specific security features. The Configuring Interfaces" chapter provides additional information on the CHAP and PAP authentication features. Another example is the IP Security Option (IPSO) feature described in the chapter entitled "Configuring IP." Finally, see the separate protocol chapters for information about how to create access lists.

Establish Password Protection

Complete the tasks in the following sections to establish password protection:

- Protect Access to Terminal Lines
- Protect Passwords with Enable Secret
- Encrypt Passwords

Protect Access to Terminal Lines

You can provide access control on a terminal line by entering the password and establishing password checking. To do so, perform the following tasks in line configuration mode:

Task		Command[1]
Step 1	Assign a password to a terminal or other device on a line.	**password** *password*
Step 2	Enable password checking at login.	**login**

[1] These commands are documented in the "Terminal Line and Modem Support Commands" chapter of the *Access and Communication Servers Command Reference* publication.

The password checker is case sensitive and can include spaces. The password *Secret* is different than the password *secret*, for example, and the password *two words* is an acceptable password.

Protect Passwords with Enable Secret

To provide an additional layer of security, particularly for passwords that cross the network or are stored on a TFTP server, you can use either **enable password** or **enable secret**. Both commands accomplish the same thing; that is, they let you establish an encrypted password that users must type to enter enable mode (the default), or any privilege level you specify.

You should use **enable secret**, however, because it is the newer command and uses an improved encryption algorithm. You would use **enable password** only if you boot an older image of the IOS or you boot older boot roms that don't recognize enable secret. In that case **enable password** would be the only form of password protection available to you.

If you configure **enable secret**, it is used instead of enable password, not in addition to it .

To configure the router to require an enable password, perform one of the following tasks in global configuration mode:

Task	Command
Establish a password for the privileged command level.	**enable password** [**level** *level*] {*password}* **enable password** [**level** *level*] {*encryption-type encrypted-password*
Specify a secret password, saved using a non-reversible encryption method. When both the enable password and enable secrets are set, this is the password the user enters.	**enable secret** [**level**] {*password*} **enable secret** [**level** *level*] {*encryption-type encrypted-password*

Use this command with the **level** option to define a password for a specific privilege level. Once the level and the password are specified, give the password to the users you want to have access at this level. Use the **privilege level** configuration command to specify commands accessible at various level.

If you have **service password-encryption** set, the password you enter is encrypted. When the router displays it for you later, it will be displayed in the encrypted form.

If you specify an ecncyption type, you must provide an encrypted password—this would be an encrypted password you copy from another router configuration.

You cannot recover a lost encrypted password. You must clear NVRAM and set a new password. See the section “Recover a Lost Enable Password” or the section “Recover a Lost Line Password” if you have lost or forgotten your password.

Encrypt Passwords

Because protocol analyzers can examine packets, you can increase access security to your communication server by configuring it to encrypt passwords. Encryption prevents the password from being read in the configuration file with the **show configuration** EXEC command or with a protocol analyzer.

Configure the communication server to encrypt passwords by performing the following task in global configuration mode:

Task	Command
Encrypt a password.	**service password-encryption**

It is not possible to recover a lost encrypted password.

Configure Multiple Privilege Levels

By default, the communication server has two levels of password security, user level and privileged, or enabled, level. You can configure up to sixteen hierarchical levels of security on a communication server. By configuring multiple passwords, you can allow different sets of users to have access to specified commands.

For example, if you want the **configure** command to be available to a more restricted set of users than the **clear line** command, you can assign level 2 security to the **clear line** command and distribute the level 2 password fairly widely, and assign level 3 security to the **configure** command and distribute the password to level 3 commands to fewer users.

To configure additional levels of security, perform the tasks in the following sections:

- Change the Default Privilege Level for Lines
- Display Current Privilege Levels
- Protect Passwords over Networks with Enable Secret

Change the Default Privilege Level for Lines

To change the default privilege level for a given line or a group of lines, perform the following task in line configuration mode:

Task	Command
Specify a default privilege level for a line.	**privilege level** *level*

Display Current Privilege Levels

To display your level of privilege based on the password you used, perform the following task in EXEC mode:

Task	Command
Display your current privilege level.	**show privilege**

Protect Passwords over Networks with Enable Secret

To provide an additional level of password security, especially useful when a password crosses a network or is stored on a TFTP server, perform the following tasks in global configuration mode:

Task	Command
Establish a password for the privileged command level.	**enable password** *password*
Specify a secret password, saved using a non-reversible encryption method. When both the enable password and enable secrets are set, this is the password the user will be enter.	**enable secret** *password*

You cannot recover a lost encrypted password.

Disable Password Protection

You can disable line password verification by disabling password checking. To do so, perform the following task in line configuration mode:

Task	Command
Disable password checking or allow access to a line without password verification.	**no login**[1]

[1] This command is documented in the "Terminal Line and Modem Support Commands" chapter of the *Access and Communication Servers Command Reference* publication.

Recover a Lost Enable Password

On communication servers equipped with nonvolatile memory, you can accidentally lock yourself out if you forget the enable password. The process to recover a lost enable password depends upon on the type of communication server platform you have.

ASM-CS and Cisco 2500

To recover a lost enable password on an ASM-CS or a Cisco 2500, follow these steps:

Step 1 Turn the communication server power off, then on again.

Step 2 Press the Break key within the first 60 seconds after power cycle. This will put the communication server in the ROM monitor mode indicated by the prompt “>”.

Step 3 Enter the **o** command to display the configuration register value. Make a note of the value; you will need it in a later step of this procedure:

```
 > o
```

Step 4 From the ROM monitor prompt, type **o/r**. This will put the value 0x141 into the configuration register.

```
> o/r
```

Step 5 Initialize and boot the communication server:

```
 > i
```

Step 6 Answer **NO** to all the SETUP questions. You should see the following prompt:

```
cs(boot)>
```

Step 7 Enter privileged mode by entering the **enable** command. No password is required and the prompt will change to `cs(boot)#`.

```
cs(boot)> enable
cs(boot)#
```

Step 8 Use the **show configuration** command to see the enable password:

```
cs(boot)# show configuration
```

Step 9 Restore the configuration register to its original value (as recorded in step 3) using the configuration command:

```
 cs(boot)# config-register 0xoriginal values
```

Step 10 Reload and boot the communication server using the **reload** command:

```
cs(boot)# reload
```

500-CS

The DEFAULT button on the front of the system is a switch for resetting the system to the defaults. This button is helpful if you forget the enable password.

Caution The following procedure overwrites the configuration parameters stored in nonvolatile memory. You should maintain a written copy of the system configuration so that you can reconfigure the system.

To recover a lost enable password on a 500-CS, follow these steps:

Step 1 Turn off power to the 500-CS.

Step 2 Hold down the DEFAULT button and turn on the power.

Step 3 When the OK and LAN LEDs blink at the same time, release the DEFAULT button.

Note If you continue to press the DEFAULT button, the system software assumes the button is stuck and ignores it.

Recover a Lost Line Password

On communication servers equipped with nonvolatile memory, you can accidentally lock yourself out if you enable password checking on the console terminal line and then forget the line password.

To recover a lost line password, force the server into factory diagnostic mode and then follow these steps:

Step 1 Force the server into factory diagnostic mode.

See the hardware installation and maintenance publication for your product for specific information about configuring the processor configuration register for factory diagnostic mode. Table 5-1 summarizes the hardware or software settings required by various products to set factory diagnostic mode.

Step 2 You will be asked if you want to set the manufacturers' addresses. Respond by typing **Yes**. You will then see the following prompt:

```
TEST-SYSTEM>
```

Step 3 Enter the **enable** command to get the privileged prompt:

```
TEST-SYSTEM> enable
```

Step 4 Enter the **show configuration** command to review the system configuration and find the password. Do not change anything in the factory diagnostic mode.

```
TEST-SYSTEM# show configuration
```

Step 5 To resume normal operation, restart the server and/or reset the configuration register.

Step 6 Log into the server with the password that was shown in the configuration file.

Note All debugging capabilities are turned on during diagnostic mode.

Table 5-1 Factory Diagnostic Mode Settings for the Configuration Register

Platform	Setting
ASM-CS	Set jumper in bit 15 of the processor configuration register, then restart; remove jumper when finished.
Cisco 2500 series	Use the **config register** command to set the processor configuration register to 0x8000, then **initialize** and **boot** the system. Use the **reload** command to restart and set the processor configuration register to 0x2102 when finished.

Create Access Lists

This section summarizes the protocols that use access lists. The general guidelines for access lists vary from protocol to protocol. See the appropriate chapter in this guide for detailed task information on each protocol-specific access list. To control SNMP access, see “Security Management” earlier in this chapter. Also refer to the appropriate protocol-specific chapters of this publication.

Table 5-2 provides the protocols that have access lists specified by numbers and provides the corresponding numerical ranges.

Table 5-2 Protocols with Access Lists Specified by Numbers

Protocol	Range
IP	1–99
Extended IP	100–199
Ethernet type code	200–299
Ethernet address	700–799
IPX	800–899
Extended IPX	900–999
IPX SAP	1000–1099

Establish Terminal Access Control

You can configure the communication server to use one of three special TCP/IP protocols related to Terminal Access Controller Access Control System: regular TACACS, Extended TACACS (XTACACS), and AAA/TACACS+. All three versions provide additional control over servers that run on a timesharing system. TACACS services are provided by and maintained in a database on the TACACS server. Our basic TACACS support is modeled after the original Defense Data Network (DDN) application.

A comparative description of the supported versions follows. A comparison of the versions by commands is found in Table 5-3.

- TACACS—provides password checking, authentication, and notification of user actions for security and accounting purposes.
- XTACACS—provides information about protocol translator and communication server use. This information is used in UNIX auditing trails and accounting files.
- AAA/TACACS+—provides more detailed accounting information as well as more administrative control of authentication and authorization processes.

You can establish TACACS-style password protection on both user and privileged levels of the system EXEC.

Table 5-3 TACACS Command Comparison

Command	TACACS	XTACACS	TACACS+
aaa accounting			X
aaa authentication arap			X
aaa authentication enable default			X
aaa authentication login			X

Command	TACACS	XTACACS	TACACS+
aaa authentication override			X
aaa authentication ppp			X
aaa authorization			X
aaa new-model			X
arap authentication			X
arap use-tacacs	X	X	
enable last-resort	X	X	
enable use-tacacs	X	X	
login authentication			X
login tacacs	X	X	
ppp authentication	X	X	X
ppp use-tacacs	X	X	X
tacacs-server attempts	X	X	X
tacacs-server authenticate	X	X	
tacacs-server extended		X	
tacacs-server host	X	X	X
tacacs-server last-resort	X	X	
tacacs-server notify	X	X	
tacacs-server optional-passwords	X	X	
tacacs-server retransmit	X	X	X
tacacs-server timeout	X	X	X

Enable TACACS and XTACACS

The following sections describe the features available with TACACS and XTACACS:

Note Many original TACACS and XTACACS commands cannot be used once you have initialized AAA/TACACS+. To identify which commands can be used with the three versions, refer to Table 5-3.

- Set TACACS Password Protection at the User Level
- Disable Password Checking at the User Level
- Set Optional Password Verification
- Set TACACS Password Protection at the Privileged Level
- Disable Password Checking at the Privileged Level
- Set Notification of User Actions
- Set Authentication of User Actions
- Establish the TACACS Server Host and Response Times

- Set Limits on Login Attempts
- Enable the Extended TACACS Mode
- Enable TACACS for PPP and ARA Protocol Authentication

Note If you require additional protection using TCP/IP access lists, see the chapter “Configuring IP Routing Protocols” for more information.

Set TACACS Password Protection at the User Level

You can enable password checking at login by performing the following task in line configuration mode:

Task	Command
Set the TACACS-style user ID and password-checking mechanism.	**login tacacs**[1]

[1] This command is documented in the “Terminal Line and Modem Support Commands” chapter of the *Access and Communication Servers Command Reference* publication.

Disable Password Checking at the User Level

If a TACACS server does not respond to a login request, the communication server will deny the request by default. However, you can prevent that login failure in one of two ways. You can allow a user to access privileged EXEC mode if that user enters the password set by the **enable** command.

Alternatively, you can ensure a successful login by allowing the user to access the privileged EXEC mode without further question.

To specify one of these features, perform either of the following tasks in global configuration mode:

Task	Command
Allow a user to access privileged EXEC mode.	**tacacs-server last-resort password**
Set “last resort” options for logins.	**tacacs-server last-resort succeed**

Set Optional Password Verification

You can specify that the first TACACS request to a TACACS server is made without password verification. To do so, perform the following task in global configuration mode:

Task	Command
Set TACACS password as optional.	**tacacs-server optional-passwords**

When the user types in the login name, the login request is transmitted with the name and a zero-length password. If accepted, the login procedure is completed. If the TACACS server refuses this request, the terminal server prompts for a password and tries again when the user supplies a password. The TACACS server must support authentication for users without passwords to make use of this feature. This feature supports all TACACS requests such as login, SLIP, and enable.

Set TACACS Password Protection at the Privileged Level

You can set the TACACS protocol to determine whether a user can access the privileged EXEC level. To do so, perform the following task in global configuration mode:

Task	Command
Set the TACACS-style user ID and password-checking mechanism at the privileged EXEC level.	**enable use-tacacs**

When you set TACACS password protection at the privileged EXEC level, the EXEC **enable** command will ask for both a new username and a password. This information is then passed to the TACACS server for authentication. If you are using the extended TACACS, it will also pass any existing UNIX user identification code to the server.

Caution If you use the **enable use-tacacs** command, you must also specify **tacacs-server authenticate enable**; otherwise, you will be locked out of the communication server.

Note When used without extended TACACS, this task allows anyone with a valid username and password to access the privileged command level, creating a potential security problem. This is because the TACACS query resulting from entering the **enable** command is indistinguishable from an attempt to log in without extended TACACS.

Disable Password Checking at the Privileged Level

You can specify what happens if the TACACS servers used by the **enable** command do not respond. To invoke this "last resort" login feature, perform either of the following tasks in global configuration mode:

Task	Command
Allow user to enable by asking for the privileged EXEC-level password.	**enable last-resort password**
Allow user to enable without further questions.	**enable last-resort succeed**

Set Notification of User Actions

You can cause a message to be transmitted to the TACACS server when a user either makes a TCP connection, enters the **enable** command, or logs out. To do so, perform the following task in global configuration mode:

Task	Command
Set server notification of user actions.	**tacacs-server notify** {**connection** [**always**] | **enable** | **logout** [**always**] | **slip** [**always**]}

The retransmission of the message is performed by a background process for up to five minutes. The terminal user, however, receives an immediate response allowing access to the terminal.

The **tacacs-server notify** command is only available when you have set up an extended TACACS server using the latest Cisco extended TACACS server software, available using FTP (see the README file in the *ftp.cisco.com* directory).

Set Authentication of User Actions

For a TCP connection, you can specify that if a user tries to make a connection, the communication server requires a response from the network or communication server indicating whether the user can make the connection. You can also specify that the communication server should perform authentication even when a user is not logged in.

For a SLIP or PPP session, you can specify that if a user tries to start a session, the communication server requires a response from the network or communication server indicating whether the user can start the session. You can specify that the communication server should perform authentication even when a user is not logged in. You can also request that the communication server install access lists.

For use of the **enable** command, you can specify that if a user issues the enable command, the communication server must respond indicating whether the user can give the command.

To configure any of these scenarios, perform the following task in global configuration mode:

Task	Command
Set server authentication of user actions.	**tacacs-server authenticate** {**connection** [**always**] | **enable** | **slip** [**always**] [**access-lists**]}

The **tacacs-server authenticate** command is only available when you have set up an extended TACACS server using the latest Cisco extended TACACS server software, available using FTP (see the README file in the *ftp.cisco.com* directory).

Establish the TACACS Server Host and Response Times

You can specify the names of the IP host or hosts maintaining a TACACS server. The software searches for the hosts in the order specified, so this feature can be useful for setting up a list of preferred servers.

You can also modify the number of times the system software searches the list of TACACS servers (from the default of two times) and the interval it waits for a reply (from the default of 5 seconds).

Perform the following tasks as needed for your system configuration in global configuration mode:

Task	Command
Specify a TACACS host.	**tacacs-server host** *name*
Specify the number of times the server will search the list of TACACS server hosts before giving up.	**tacacs-server retransmit** *retries*
Set the interval the server waits for a TACACS server host to reply.	**tacacs-server timeout** *seconds*

Set Limits on Login Attempts

You can set controls on the number of login attempts that can be made on a line set up for TACACS by performing the following task in global configuration mode:

Task	Command
Control the number of login attempts that can be made on a line set for TACACS verification.	**tacacs-server attempts** *count*

Enable the Extended TACACS Mode

Extended TACACS mode provides information about the terminal requests to help set up UNIX auditing trails and accounting files for tracking use of protocol translators and communication servers. The information includes responses from these network devices and validation of user requests.

An unsupported, extended TACACS server is available using FTP for UNIX users who want to create the auditing programs (see the README file in the *ftp.cisco.com* directory).

Extended TACACS differs from standard TACACS in that standard TACACS provides only username and password information.

To enable extended TACACS mode, perform the following task in global configuration mode:

Task	Command
Enable an extended TACACS mode.	**tacacs-server extended**

Enable TACACS for PPP and ARA Protocol Authentication

You can use the TACACS protocol for authentication within PPP sessions. To do so, perform the following tasks in interface configuration mode:

Task		Command
Step 1	Enable CHAP or PAP.	**ppp authentication {chap | pap} [if-needed]**
Step 2	Enable TACACS under PPP.	**ppp use-tacacs** [**single-line**]

You can also use the TACACS protocol for authentication within AppleTalk Remote Access (ARA) protocol sessions. To do so, perform the following task in line configuration mode:

Task	Command
Enable TACACS under the ARA protocol.	**arap use-tacacs** [**single-line**][1]

[1] This command is documented in the "AppleTalk Remote Access Commands" chapter of the *Access and Communication Servers Command Reference* publication.

Use the **ppp use-tacacs** and **arap use-tacacs** commands only when you have set up an extended TACACS server using the latest Cisco extended TACACS server software, available using FTP (see the README file in the *ftp.cisco.com* directory).

For more information on PPP, refer to the "Configuring Interfaces" chapter later in this publication. For more information on the ARA protocol, refer to the "Configuring an AppleTalk Remote Access Server" chapter later in this publication. For examples of enabling TACACS for PPP and ARA protocol authentication, see the section "System Management Examples".

Configure AAA/TACACS+

AAA/TACACS+ combines original and enhanced functionality from previous versions of TACACS as well incorporating a new model of control called AAA (Authentication, Authorization, and Accounting). The resulting benefits are more accurate accounting information and improved remote access functionality.

AAA's subset of services are broken down into the following functional groups:

- Authentication—provides complete server control of authentication through login and password query, challenge/response, messaging support, and encryption in MD5.
- Authorization—provides remote access control, including one-time authorization, authorization for each service, per-user account list and profile, user group support, support of IP, IPX, ARA, and Telnet. Additionally, you can create access or command permissions and restrictions.
- Accounting—collects and sends the TACACS server information used for billing, auditing, and reporting. These functions are discussed in the "System Management Examples" section later in this chapter.

AAA also provides an API (Advanced Programming Interface) that allows the protocol to be integrated into existing standard databases.

You will need a server running TACACS software to use the AAA/TACACS+ functionality on your communication server. You can obtain this software free of charge from Cisco, or can purchase software from a third-party vendor.

Note Many original TACACS and XTACACS commands cannot be used once you initialize AAA/TACACS+. To identify the commands that can be used with the three versions of TACACS, refer to Table 5-3.

The following sections describe the features available in AAA/TACACS+:

- Enable AAA/TACACS+ and Set Authentication Key
- Enable Authentication for ARA
- Enable TACACS+ Password Protection at the Privileged Level
- Enable Authentication for Login
- Enable an Authentication Override
- Enable Authentication for PPP
- Restrict Network Access

Additionally, the following functionality from the previous versions of TACACS are also available in TACACS+. Refer to the section "Enable TACACS and XTACACS."

- Establish the TACACS Server Host and Response Times
- Set Limits on Login Attempts

Enable AAA/TACACS+ and Set Authentication Key

To enable AAA/TACACS+, perform the following tasks in global configuration mode:

Task	Command
Enable AAA/TACACS+.	**aaa new-model**
Set the authentication and encryption key to the same key used on the TACACS+ daemon.	**tacacs-server key** *key*

Enable Authentication for ARA

With the **aaa authentication arap** command, you create one or more lists of authentication methods that will be tried when ARA users log into the communication server. These lists are used with the **arap authentication** line command.

The *list-name* is any character string used to name the list you are creating. The *method* refers to the actual list of methods the authentication algorithm will try, in the sequence entered. You can enter up to four methods that this list should try in sequence.

To create a default list that will be used if no list is specified in the **arap authentication** command, use the **default** argument followed by the methods you wish to be used in default situations.

The additional methods of authentication will only be used if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all methods return an error, specify **none** as the final method in the command line.

Perform the following task in global configuration mode:

Task	Command
Enable authentication for ARA users.	**aaa authentication arap** {**default** | *list-name*} *method1* [...[*method4*]]

Enable TACACS+ Password Protection at the Privileged Level

Use the **aaa authentication enable default** command to create a series of authentication methods that are used to determine if a user can access privileged EXEC command level. You can specify up to four authentication methods. The additional methods of authentication are only used if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all methods return an error, specify **none** as the final method in the command line.

Perform the following task in global configuration mode:

Task	Command
Enable TACACS+ user ID and password checking for users requesting privileged EXEC level.	**aaa authentication enable default** *method1* [...[*method4*]]

Enable Authentication for Login

With the **aaa authentication login** command, you create one or more lists of authentication methods that are tried at login. These lists are used with the **login authentication** line command.

The keyword *list-name* is any character string used to name the list you are creating. The *method* keyword refers to the actual list of methods the authentication algorithm tries, in the sequence entered. You can enter up to four methods that this list should try in sequence.

To create a default list that will be used if no list is specified in the **login authentication** command, use the **default** argument followed by the methods you want used in default situations.

The additional methods of authentication are only used if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all methods return an error, specify **none** as the final method in the command line.

Perform the following task in global configuration mode:

Task	Command
Enable AAA authentication at login.	**aaa authentication login** {**default** | *list-name*} *method1* [...[*method4*]]

Enable an Authentication Override

To have the communication server check the local user database for authentication before attempting another form of authentication, use the **aaa authentication local-override** command. This command is useful when you want to configure an override to the normal authentication process for certain personnel such as system administrators.

Perform the following task in global configuration mode:

Task	Command
Create an override for authentication.	**aaa authentication local-override**

Enable Authentication for PPP

With the **aaa authentication ppp** command, you create one or more lists of authentication methods that are tried during PPP sessions. These lists are used with the **ppp authentication** line command.

The keyword *list-name* is any character string used to name the list you are creating. The *method* keyword refers to the actual list of methods the authentication algorithm tries, in the sequence entered. You can enter up to four methods that this list tries in sequence.

To create a default list that will be used if no list is specified in the **ppp authentication** command, use the **default** argument followed by the methods you want used in default situations.

The additional methods of authentication are only used if the previous method returns an error, not if it fails. To specify that the authentication should succeed even if all methods return an error, specify **none** as the final method in the command line.

Perform the following task in global configuration mode:

Task	Command
Enable AAA authentication for PPP.	**aaa authentication ppp** {**default** | *list-name*} *method1* [...[*method4*]]

Restrict Network Access

Using the **aaa authorization** command you create a list of one and up to four authorization methods that are used when a user accesses the specified function.

The additional methods of authorization are only used if the previous method returns an error, not if it fails. To specify that the authorization should succeed even if all methods return an error, specify **none** as the final method in the command line.

Perform the following task in global configuration mode:

Task	Command
Restrict network access using AAA.	**aaa authorization** {**network** | **connection** | **exec command** *level*} *methods*

Enable CHAP

Access control using Challenge Handshake Authentication Protocol (CHAP) is available on all serial interfaces that use PPP encapsulation. The authentication feature reduces the risk of security violations on your communication server.

When CHAP is enabled, a remote device (a PC, workstation, communication server, or communication server) attempting to connect to the local communication server is requested, or "challenged," to respond.

The challenge consists of an ID, a random number, and either the host name of the local communication server or the name of the user on the remote device. This challenge is transmitted to the remote device.

The required response consists of two parts:

- An encrypted version of the ID, a secret password (or secret), and the random number
- Either the host name of the remote device or the name of the user on the remote device

When the local communication server receives the challenge response, it verifies the secret by looking up the name given in the response and performing the same encryption operation. The secret passwords must be identical on the remote device and the local communication server.

By transmitting this response, the secret is never transmitted, thus preventing other devices from stealing it and gaining illegal access to the system. Without the proper response, the remote device cannot connect to the local communication server.

CHAP transactions occur only when a link is established. The local communication server does not request a password during the rest of the call. (The local communication server can, however, respond to such requests from other devices during a call.)

To use CHAP, perform the following task in interface configuration mode:

Task	Command
Enable CHAP on the interface.	**ppp authentication chap** [**if-needed**] or **ppp authentication chap** [*list-name*]

The optional argument **if-needed** can only be used with TACACS or XTACACS. The optional keyword *list-name* can only be used with AAA/TACACS+. CHAP is specified in RFC 1334. It is an additional authentication phase of the PPP Link Control Protocol.

Once you have enabled CHAP, the local communication server requires a response from the remote devices. If the remote device does not support CHAP, no traffic is passed to that device.

Enable PAP

Access control using the Password Authentication Protocol (PAP) is available on all serial interfaces that use PPP encapsulation. The authentication feature reduces the risk of security violations on your communication server.

The optional argument **if-needed** can only be used with TACACS or XTACACS. The optional keyword *list-name* can only be used with AAA/TACACS+. To use PAP, perform the following task in interface configuration mode:

Task	Command
Enable PAP on the interface.	**ppp authentication pap** [**if-needed**] or **ppp authentication pap** [*list-name*]

Enable IP Accounting and Display IP Access Violations

In addition to providing basic IP accounting functions, our IP accounting support provides information identifying IP traffic that fails IP access lists. Identifying IP source addresses that violate IP access lists alerts you to possible attempts to breach security. The data also indicates that you should verify IP access list configurations.

To make this feature available to users, you must first enable IP accounting. Users can then display the number of bytes and packets from a single source that attempted to breach security against the access list for the source destination pair.

The IP accounting access violations output displays the number of the access list failed by the last packet for the source and destination pair. The number of packets reveals how aggressive the attack is upon a specific destination.

By default, IP accounting displays the number of packets that have passed access lists and were routed. To enable IP accounting, perform the following task for each interface in interface configuration mode:

Task	Command
Enable IP accounting.	**ip accounting**[1]

[1] This command is documented in the "IP Commands" chapter in the *Access and Communication Servers Command Reference* publication.

To display IP access violations for a specific IP accounting database, perform the following task in EXEC mode:

Task	Command
Display IP access-violation information.	**show ip accounting access-violations**

Fault Management

To perform general fault management, complete the tasks in the following sections:

- Display System Information
- Test Network Connectivity
- Limit TCP Transactions
- Convert Line Numbers from Octal to Decimal
- Log System Error Messages
- Enable Debug Operations

Most chapters in this guide include fault management tasks in a monitoring and maintaining section. For example, the chapter “Configuring Interfaces” provides a section on interface loopback testing. Another example is the information on Internet Control Messages Protocol (ICMP) support described in the chapter “Configuring IP.”

Display System Information

To provide information about system processes, the software includes an extensive list of EXEC commands that begin with the word **show**, which, when executed, display detailed tables of system information. Perform the following tasks in EXEC mode to display the information described:

Task	Command
Display information about all active processes.	**show processes** [**cpu**]
Display the configured protocols.	**show protocols**

Look for specific **show** commands in the tables of configuration tasks found throughout the chapters in this guide. See the *Access and Communication Servers Command Reference* publication for detailed descriptions of the commands.

Test Network Connectivity

Complete the tasks in the following sections to test basic network connectivity:

- Set up the TCP Keepalive Packet Service
- Test Connections with the Ping Command
- Trace Packet Routes

Set up the TCP Keepalive Packet Service

The TCP keepalive capability allows a communication server to detect when the host with which it is communicating experiences a system failure, even if data stops being transmitted (in either direction). This is most useful on incoming connections. For example, if a host failure occurs while talking to a printer, the communication server might never notice, since the printer does not generate any traffic in the opposite direction. If keepalives are enabled, they are sent once every minute on otherwise idle connections. If five minutes pass and no keepalives are detected, the connection is closed. The connection will also be closed if the host replies to a keepalive packet with a reset packet. This will happen if the host crashes and comes back up again.

To set up the TCP keepalive packet service, perform the following task in global configuration mode:

Task	Command
Generate TCP keepalive packets on idle network connections, either incoming connections initiated by a remote host, or outgoing connections initiated by a user.	**service tcp-keepalives {in | out}**

Test Connections with the Ping Command

As an aid to diagnosing basic network connectivity, many network protocols support an echo protocol. The protocol involves sending a special datagram to the destination host, then waiting for a reply datagram from that host. Results from this echo protocol can help in evaluating the path-to-host reliability, delays over the path, and whether the host can be reached or is functioning.

To use the echo protocol, perform the following task in EXEC mode:

Task	Command
Invoke a diagnostic tool for testing connectivity.	**ping** [*protocol*] {*host* | *address*}

Look for specific **ping** commands in the tables of configuration tasks found throughout the chapters in this guide. See the *Access and Communication Servers Command Reference* publication for detailed descriptions of the command.

Trace Packet Routes

You can discover the routes that packets will actually take when traveling to their destinations. To do so, perform the following task in EXEC mode:

Task	Command
Trace packet routes through the network.	**trace** [*protocol*] [*destination*]

Limit TCP Transactions

When using a standard TCP implementation to send keystrokes between machines, TCP tends to send one packet for each keystroke typed. On larger networks, many small packets use up bandwidth and contribute to congestion.

John Nagle’s algorithm (RFC-896) helps alleviate the small-packet problem in TCP. In general, it works this way: The first character typed after connection establishment is sent in a single packet, but TCP holds any additional characters typed until the receiver acknowledges the previous packet. Then the second, larger packet is sent, and additional typed characters are saved until the acknowledgement comes back. The effect is to accumulate characters into larger chunks, and pace them out to the network at a rate matching the round-trip time of the given connection. This method is usually a good for all TCP-based traffic. However, do not enable the Nagle slow packet avoidance algorithm if you have XRemote users on X Window sessions.

By default, the Nagle algorithm is not enabled. To invoke the Nagle algorithm and thereby reduce TCP transactions, perform the following task in global configuration mode:

Task	Command
Enable the Nagle slow-packet-avoidance algorithm.	**service nagle**

Convert Line Numbers from Octal to Decimal

You can convert the line numbers on the ASM-CS communication server from their default octal values to decimal values.

To convert ASM-CS octal line numbers to decimal numbers, perform the following task in global configuration mode:

Task	Command
Specify that line numbers be displayed and interpreted as decimal numbers rather than octal numbers.	**service decimal-tty**

Log System Error Messages

By default, the network servers send the output from **debug** EXEC command and system error messages to the console terminal. You can redirect these messages, as well as output from asynchronous events such as interface transition, to other destinations. These destinations include virtual terminals, internal buffers, and UNIX hosts running a syslog server; the syslog format is compatible with 4.3 BSD UNIX.

Additionally, you can set the severity level of the messages to control the type of messages displayed. You can also have log messages timestamped to enhance real-time debugging and management.

There are three syslog messages at LOG_NOTICE syslog level that make it easier to check the status of how the system provides address resolution. An example follows:

```
%LINK-5-BOOTP: Ethernet0 address 131.108.160.24, resolved by 131.108.1.111
%LINK-5-RARP: Ethernet0 address 131.108.160.24, resolved by 131.108.1.111
%LINK-5-SLARP: Ethernet0 address 131.108.160.24, resolved by 131.108.1.111
```

The following startup messages help you identify NVRAM problems:

```
Warning: NVRAM device not found
Warning: NVRAM invalid, possibly due to write erase
```

Log Errors to a UNIX Syslog Daemon

To set up the syslog daemon on a 4.3 BSD UNIX system, include a line such as the following in the file */etc/syslog.conf*:

```
local7.debugging /usr/adm/logs/cisco.log
```

The **local7** keyword specifies the logging facility to be used; see Table 5-5 for a list of other keywords. The **debugging** keyword specifies the syslog level; see Table 5-4 for a list of other keywords.

The syslog daemon sends messages at this level or a more severe level to the file specified in the next field. The file must already exist, and the syslog daemon must have permission to write to it.

Enable Message Logging

To enable message logging, perform the following task in global configuration mode:

Task	Command
Enable message logging.	**logging on**

Set the Error Message Display Device

By default, error messages are directed to the system console. To direct messages to other devices, perform one of the following tasks in global configuration mode:

Task	Command
Log messages to an internal buffer.	**logging buffered**
Log messages to a UNIX syslog server host.	**logging** *host*
Redirect messages to the system console.	**no logging on**

The internal buffer is circular, so when you copy logging messages to an internal buffer instead of writing them to the console terminal, newer messages overwrite older messages. To display the messages that are logged in the buffer, use the **show logging** EXEC command. The first message displayed is the oldest message in the buffer.

The EXEC command **terminal monitor** locally accomplishes the task of displaying the system error messages to a nonconsole terminal.

When you identify more than one syslog server host, you build a list of syslog servers that receive logging messages. The **no logging** command deletes the syslog server with the specified address from the list of syslogs.

Define the Error Message Severity Level and Facilities

You can limit messages displayed to the selected device by specifying the severity level of the error message. To do so, perform one of the following tasks in global configuration mode:

Task	Command
Limit messages logged to the console.	**logging console** *level*
Limit messages logged to the terminal lines.	**logging monitor** *level*
Limit messages logged to the syslog servers.	**logging trap** *level*

The **logging console** command limits the logging messages displayed on the console terminal to messages with a level number at or below the specified severity level, which is specified by the *level* argument. Table 5-4 lists the error message *level* keywords and corresponding UNIX syslog definitions in order from the most severe level to the least severe level.

Table 5-4 Error Message Logging Keywords

Level Keyword	Level	Description	Syslog Definition
emergencies	0	System unusable	LOG_EMERG
alerts	1	Immediate action needed	LOG_ALERT
critical	2	Critical conditions	LOG_CRIT
errors	3	Error conditions	LOG_ERR
warnings	4	Warning conditions	LOG_WARNING
notifications	5	Normal but significant condition	LOG_NOTICE
informational	6	Informational messages only	LOG_INFO
debugging	7	Debugging messages	LOG_DEBUG

The **no logging console** command disables logging to the console terminal.

The default is to log messages to the console at the **debugging** level and those level numbers that are lower, which means all levels. The **logging monitor** command defaults to **debugging** also. The **logging trap** command defaults to **informational**.

To display logging messages on a terminal, use the **terminal monitor** EXEC command.

Current software generates four categories of error messages:

- Error messages about software or hardware malfunctions, displayed at levels **warnings** through **emergencies**
- Output from the **debug** commands, displayed at the **debugging** level
- Interface up/down transitions and system restart messages, displayed at the **notifications** level
- Reload requests and low-process stack messages, displayed at the **informational** level

Define the Syslog Facility

You can also configure the syslog facility in which error messages are sent by performing the following task in global configuration mode:

Task	Command
Configure system log facilities.	**logging facility** *facility-type*

Table 5-5 lists the logging facility types and their descriptions.

Table 5-5 Logging Facility Types

Facility Type	Description
auth	Indicates the authorization system.
cron	Indicates the cron facility.
daemon	Indicates the system daemon.
kern	Indicates the Kernel.
local0–7	Reserved for locally defined messages.
lpr	Indicates line printer system.
mail	Indicates mail system.
news	Indicates USENET news.
sys9	Indicates system use.
sys10	Indicates system use.
sys11	Indicates system use.
sys12	Indicates system use.
sys13	Indicates system use.
sys14	Indicates system use.
syslog	Indicates the system log.

user	Indicates user process.
uucp	Indicates UNIX-to-UNIX copy system.

Refer also to your syslog manual pages.

To display the addresses and levels associated with the current logging setup, as well as any other logging statistics, perform the following task in EXEC mode:

Task	Command
Display the state of syslog error and event logging, including host addresses and whether console logging is enabled.	**show logging**

Enable Timestamps on Log Messages

By default, log messages are not timestamped. You can enable timestamping of log messages by performing the following task in global configuration mode:

Task	Command
Enable log timestamps.	**service timestamps log uptime** or **service timestamps log datetime** [**msec**] [**localtime**] [**show-timezone**]

Enable Debug Operations

Your communication server includes hardware and software to aid in tracking down problems with the communication server or with other hosts on the network. The privileged **debug** EXEC commands start the console display of several classes of network events. The following tasks describe in general the system debug message feature. Refer to the *Debug Command Reference* publication for all information regarding **debug** commands. Also refer to the *Troubleshooting Internetworking Systems* publication.

Task	Command
Display the state of each debugging option.	**show debugging**
Display a list and brief description of all the **debug** command options.	**debug ?**
Begin message logging for the specified **debug** command.	**debug** *command*
Turn message logging off for the specified **debug** command.	**no debug** *command*

You can configure timestamping of system debug messages. Timestamping enhances real-time debugging by providing the relative timing of logged events. This information is especially useful when customers send debugging output to your technical support personnel for assistance. To enable timestamping of system debug messages, perform the following task in global configuration mode:

Task	Command
Enable timestamping of system debug messages.	**service timestamps debug uptime** or **service timestamps debug datetime** [**msec**] [**localtime**] [**show-timezone**]

Normally, the messages are displayed only on the console terminal. See the section “Set the Error Message Display Device” earlier in this chapter to change the output device.

Note The system gives high priority to debugging output. For this reason, debugging commands should be turned on only for troubleshooting specific problems or during troubleshooting sessions with technical support personnel. Excessive debugging output can render the system inoperable.

System Performance Management

The following sections describe how to manage general system performance:

- Configure Switching and Scheduling Priorities
- Establish Queuing Strategies
- Modify the System Buffer Size
- Delay EXEC Startup
- Handle Idle Telnet Connections

In addition, most chapters in this guide include performance tasks specific to the chapter content, and the *Internetworking Design Guide* includes detailed information on performance issues that arise when designing a network.

Configure Switching and Scheduling Priorities

The normal operation of the network server allows the switching operations to use as much of the central processor as is required. If the network is running unusually heavy loads that do not allow the processor the time to handle the routing protocols, you might need to give priority to the system process scheduler. To do so, perform the following task in global configuration mode:

Task	Command
Define the maximum amount of time that can elapse without running the lowest-priority system processes.	**scheduler-interval** *milliseconds*

By default, a Route Switch Processor (RSP) spends 4000 microseconds switching packets at interrupt level and then 200 microseconds running processes. To change these scheduling priorities, perform the following task in global configuration mode:

Task	Command
Change the amount of time the RSP spends switching packets and running processes.	**scheduler allocate** *network-microseconds process-microseconds*

Establish Queuing Strategies

We provide two types of queuing strategies for prioritizing network traffic:

- Priority Queuing
- Custom Queuing

You can configure both priority queuing and custom queuing, but you can only assign one or the other to an interface.

Priority Queuing

Priority output queuing is a mechanism that allows the administrator to set priorities on the type of traffic passing through the network. Packets are classified according to various criteria, including protocol and subprotocol type, and then queued on one of four output queues (high, medium, normal, and low).

When the server is ready to transmit a packet, it scans the priority queues in order, from highest to lowest, to find the highest-priority packet. After that packet is completely transmitted, the server scans the priority queues again. If a priority output queue fills up, packets will be dropped and, for IP, quench indications will be sent to the original transmitter.

Although you can enable priority output queuing for any interface, the intended application was for low-bandwidth, congested serial interfaces. Our priority output queuing mechanism allows traffic control based on protocol or interface type. You can also set the size of the queue and defaults for what happens to packets that are not defined by priority output queue rules.

The priority output queuing mechanism can be used to manage traffic from all networking protocols. Additional fine-tuning is available for IP and for setting boundaries on the packet size.

Note Priority queuing introduces extra overhead that is acceptable for slow interfaces, but might not be acceptable for higher-speed interfaces such as Ethernet.

Note Priority queuing does not operate over X.25.

The four priority queues—high, medium, normal, and low—are listed in order from highest to lowest priority. Keepalives sourced by the network server are always assigned to the high-priority queue; all other management traffic (such as IGRP updates) must be configured. Packets that are not classified by the priority list mechanism are assigned to the normal queue.

A priority list is a set of rules that describes how packets should be assigned to priority queues. A priority list might also describe a default priority or the queue size limits of the various priority queues.

Custom Queuing

Priority queuing introduces a fairness problem in that packets classified to lower-priority queues might not get serviced in a timely manner or at all, depending upon the bandwidth used by packets sent from the higher-priority output queues.

With custom queuing, a “weighted fair” queuing strategy is implemented for the processing of interface output queues. You can control the percentage of an interface’s available bandwidth that is used by a particular kind of traffic. When custom queuing is enabled on an interface, the system maintains 11 output queues for that interface that can be used to modify queuing behavior.

For queue numbers 1 through 10, the system cycles through the queues sequentially, delivering packets in the current queue before moving on to the next. Associated with each output queue is a configurable byte count, which specifies how many bytes of data the system should deliver from the current queue before it moves on to the next queue. When a particular queue is being processed, packets are sent until the number of bytes sent exceed the queue byte count or the queue is empty. Bandwidth used by a particular queue can only be indirectly specified in terms of byte count and queue length.

Queue number 0 is a system queue; it is emptied before any of the queues numbered 1 through 10 are processed. The system enqueues high-priority packets, such as keepalive packets, to this queue. Other traffic cannot be configured to use this queue.

Note With custom or priority queueing enabled, the system takes longer to switch packets because the packets are classified by the processor card.

Queuing Task List

You can perform any of the priority-setting tasks in the following sections, depending upon the needs of your network:

- Set Priority by Protocol Type
- Assign a Default Priority
- Set Priority by Interface Type
- Specify the Maximum Packets and Bytes in the Priority Queues
- Assign a Priority Group or a Custom Queue to an Interface
- Monitor the Priority and Custom Queuing Lists

Set Priority by Protocol Type

You can establish queuing priorities based upon the protocol type by performing the following task in global configuration mode. All Cisco-supported protocols are allowed.

Task	Command
Establish queuing priorities based upon the protocol type.	**priority-list** *list-number* **protocol** *protocol-name* {**high** | **medium** | **normal** | **low**} *queue-keyword keyword-value* or **queue-list** *list-number* **protocol** *protocol-name queue-number queue-keyword keyword-value*

Assign a Default Priority

You can assign a queue for those packets that did not match any other rule in the list. To do so, perform one of the following tasks in global configuration mode:

Task	Command
Assign a priority queue for those packets that do not match any other rule in the priority list.	**priority-list** *list-number* **default {high | medium | normal | low**}
Assign a queue number for those packets that do not match any other rule in the custom queue list.	**queue-list** *list-number* **default** *queue-number*

Set Priority by Interface Type

You can establish queuing priorities on packets entering from a specific interface by performing one of the following tasks in global configuration mode:

Task	Command
Establish queuing priorities on packets entering from a given interface.	**priority-list** *list-number* **interface** *interface-type interface-number* **{high | medium | normal | low}**
Establish custom queuing based on packets entering from a given interface.	**queue-list** *list-number* **interface** *interface-type interface-number queue-number*

Specify the Maximum Packets and Bytes in the Priority Queues

You can specify the maximum number of packets that might be waiting in each of the priority queues. To do so, perform one of the following tasks in global configuration mode:

Task	Command
Specify the maximum number of packets that can be waiting in each of the priority queues.	**priority-list** *list-number* **queue-limit** *high-limit medium-limit normal-limit low-limit* or **queue-list** *list-number* **queue** *queue-number* **limit** *limit-number*
Designate the byte size allowed per queue.	**queue-list** *list-number* **queue** *queue-number* **byte-count** *byte-count-number*

Assign a Priority Group or a Custom Queue to an Interface

You can assign a priority list number to an interface. Only one list can be assigned per interface. To assign a priority group or custom queue to an interface, perform one of the following tasks in interface configuration mode:

Task	Command
Assign a priority list number to the interface.	**priority-group** *list*
Assign a custom queue list number to the interface.	**custom-queue-list** *list*

Monitor the Priority and Custom Queuing Lists

You can display information about the input and output queues when priority queuing is enabled on an interface. To do so, perform one of the following tasks in EXEC mode:

Task	Command
Show the status of the priority queuing lists.	**show queueing priority**
Show the status of the custom queuing lists.	**show queueing custom**

If you enter the **show queueing** command without any keywords, the communication server displays status on both custom and priority queue lists.

Modify the System Buffer Size

You can adjust initial buffer pool settings and the limits at which temporary buffers are created and destroyed. To do so, perform the following tasks in global configuration mode:

Task	Command
Adjust the system buffer sizes.	**buffers** {**small** | **middle** | **big** | **large** | **verylarge** | **huge** | *type number*} {**permanent** | **max-free** | **min-free** | **initial**} *number*
Dynamically resize all huge buffers to the value that you supply.	**buffers huge size** *number*

Note Normally you need not adjust these parameters; do so only after consulting with technical support personnel. Improper settings can adversely impact system performance.

During normal system operation, there are two sets of buffer pools: public and interface.

- The public pools grow and shrink based upon demand. Some public pools are temporary and are created and destroyed as needed. Other public pools are permanently allocated and cannot be destroyed. The public buffer pools are small, middle, big, large, very large, and huge.
- Interface pools are static, that is, they are all permanent. There is one interface pool for each interface in the router. For example, on a Cisco 4000 1E 4T configuration, there is one Ethernet buffer pool and 4 serial buffer pools. In the **buffers** command, the *type* and *number* arguments control the interface pools.

See the section “Buffer Modification Examples” at the end of this chapter.

The server has one pool of queuing elements and six public pools of packet buffers of different sizes. For each pool, the server keeps count of the number of buffers outstanding, the number of buffers in the free list, and the maximum number of buffers allowed in the free list. To display statistics about the buffer pool on the system, perform the following task in EXEC mode:

Task	Command
Display all public pool information.	**show buffers**
Display all public and interface pool information.	**show buffers all**
Display a brief listing of all allocated buffers.	**show buffers alloc**
Dump all allocated buffers.	**show buffers alloc dump**
Display all interface pool information.	**show buffers interface**
If the specified interface has its own buffer pool, display information for that pool.	**show buffers interface** *type number*
Display a brief listing of buffers allocated for this interface.	**show buffers interface** *type number* **alloc**
Dump the buffers allocated to this interface.	**show buffers interface** *type number* **alloc dump**

Delay EXEC Startup

You can delay the startup of the EXEC on noisy lines until the line has been idle for 3 seconds. To do so, perform the following task in global configuration mode:

Task	Command
Delay startup of the EXEC.	**service exec-wait**

Delaying startup of the EXEC is useful on noisy modem lines or when a modem attached to the line is configured to ignore MNP or V.42 negotiations, and MNP or V.42 modems may be dialing in. In these cases, noise or MNP/V.42 packets might be interpreted as usernames and passwords, causing authentication failure before the user can type a username/password. Delaying startup is not useful on nonmodem lines or lines without some kind of login configured.

Handle Idle Telnet Connections

You can configure the communication server to set the TCP window to zero (0) when the Telnet connection is idle. To do so, perform the following task in global configuration mode:

Task	Command
Set the TCP window to zero when the Telnet connection is idle.	**service telnet-zero-idle**

Normally, data sent to noncurrent Telnet connections is accepted and discarded. When **service telnet-zero-idle** is enabled, if a session is suspended (that is, some other connection is made active or the EXEC is sitting in command mode), the TCP window is set to zero. This action prevents the remote host from sending any more data until the connection is resumed. Use this command when it is important that all messages sent by the host be seen by the users and the users are likely to use multiple sessions. Do not use this command if your host will eventually time out and log out a TCP user whose window is zero.

Accounting Management

Accounting management allows you to track individual and group usage of network resources. You can then reallocate resources as needed. The following sections describe accounting management tasks:

- Enable AAA/TACACS+ Accounting
- Display Stack Utilization
- Display Memory Utilization
- Enable IP Accounting for Access List Violations

Additional tasks for measuring system resources are covered in other chapters; for example, IP accounting tasks are described in the chapter, “Configuring IP.”

Enable AAA/TACACS+ Accounting

The **aaa accounting** command allows you to set start/stop accounting for any or all of the listed functions for this command. For minimal accounting control, issue the **stop-only** command, which sends a stop record accounting notice at the end of the requested user process. For additional accounting control, you can issue the **start-stop** command, where TACACS+ sends a start accounting notice at the beginning of the requested process and a stop accounting notice at the end of the process. You can further control access and accounting by issuing the **wait-start** command, which ensures that the start notice is received by the TACACS+ server before granting the user's process request. Accounting is only done to the TACACS+ server.

Before using the aaa accounting command, you must initialize AAA/TACACS+ as described in "Enable AAA/TACACS+ and Set Authentication Key" earlier in this chapter.

Perform the following task in global configuration mode:

Task	Command
Enable accounting.	**aaa accounting** {**system** | **network** | **connection** | **exec** | **command** *level*} {**start-stop** | **wait-start** | **stop-only**} **tacacs+**

Display Stack Utilization

You can display stack utilization of processes and interrupt routines, including the reason for the last system reboot. This feature is useful for analyzing system crashes. To display stack utilization, perform the following task in EXEC mode:

Task	Command
Display stack utilization of processes and interrupt routines.	**show stacks**

Display Memory Utilization

To display memory usage information, perform the following tasks in EXEC mode:

Task	Command
Display memory pool statistics including summary information about the activities of the system memory allocator and a block-by-block listing of memory use.	**show memory** [*type*] [**free**]
Display information about memory utilization.	**show processes memory**

Enable IP Accounting for Access List Violations

You can enable accounting for Internet Protocol (IP) access list violations and display information identifying IP traffic that fails IP access lists. For information on the IP accounting access-violations feature and commands, see the "Configuring IP" chapter of this publication and the "IP Commands" chapter of the *Access and Communication Server Command Reference* publication.

System Management Examples

The following sections provide system management examples:

- System Configuration File Example
- Allowing Users to Clear Lines Example
- Defining an Enable Password for System Operators Example
- Buffer Modification Examples
- TACACS Authentication Examples
- AAA/TACACS+ Authentication Examples
- Username Examples

System Configuration File Example

The following is an example of a typical system configuration file:

```
! Define line password
line 0 4
password secret
login
!
! Define privileged-level password
enable-password Secret Word
!
! Define a system hostname
hostname TIP
! Define host filenames
boot host host1-confg 131.108.1.111
boot host host2-confg 131.108.1.111
! Define system filenames
boot system sys1-system 131.108.13.111
boot system sys2-system 131.108.1.111
!
! Enable SNMP
snmp-server community
snmp-server trap-authentication
snmp-server host 131.108.1.27 public
snmp-server host 131.108.1.111 public
snmp-server host 131.108.2.63 public
!
! Define TACACS server hosts
tacacs-server host 131.108.1.27
tacacs-server host 131.108.13.33
tacacs-server host 131.108.1.33
!
! Define a message-of-the-day banner
banner motd ^C
The Information Place welcomes you

Please call 1-800-555-2222 for a login account, or enter
your password at the prompt.
^C
```

Allowing Users to Clear Lines Example

If you want to allow users to clear lines, you can do either of the following:

- Change the privilege level for the **clear** and **clear line** commands to 1 or “ordinary user level,” as follows. This will allow any user to clear lines.

  ```
  privilege exec level 1 clear line
  ```

- Change the privilege level for the **clear** and **clear line** commands to level 2. To do this, use the **privilege level** global configuration command to specify privilege level 2. Then define an enable password for privilege level 2 and tell all of the users who can be trusted with this feature what the password is.

  ```
  enable password level 2 pswd2
  privilege exec level 2 clear line
  ```

Defining an Enable Password for System Operators Example

In the following example, you define an enable password for privilege level 10 for system operators and make **clear** and **debug** commands available to anyone with that privilege level enabled.

```
enable password level 10 pswd10
privilege exec level 10 clear line
privilege exec level 10 debug ppp chap
privilege exec level 10 debug ppp error
privilege exec level 10 debug ppp negotiation
```

The following example lowers the privilege level of **write terminal** and most configuration commands to operator level so that the configuration could be viewed by an operator, but leaves the privilege level of the **configure** command at 15 so an operator cannot change the configuration. Unless the privilege level for the individual configuration commands has been lowered to level 10, they will not be displayed in the write terminal output. The user is only allowed to see what they can configure if they have configuration privileges.

```
enable password level 15 pswd15
privilege exec level 15 configure
enable password level 10 pswd10
privilege exec level 10 write terminal
```

Buffer Modification Examples

In the following example, the system will try to keep at least 50 small buffers free:

```
buffers small min-free 50
```

In the following example, the system will try to keep no more than 200 medium buffers free:

```
buffers middle max-free 200
```

In the following example, the system will try to create one large temporary extra buffer, just after a reload:

```
buffers large initial 1
```

In the following example, the system will try to create one permanent huge buffer:

```
buffers huge permanent 1
```

TACACS Authentication Examples

The following example shows TACACS enabled for PPP authentication:

```
int async 1
ppp authentication chap
```

```
ppp use-tacacs
```

The following example shows TACACS enabled for ARAP authentication:

```
line 3
arap use-tacacs
```

AAA/TACACS+ Authentication Examples

The following example creates a default AAA authentication algorithm used with ARAP:

```
aaa authentication arap default if-needed none
```

The following example creates the same authentication algorithm for ARAP but calls the list MIS-access:

```
aaa authentication arap MIS-users if-needed none
```

Username Examples

The following sample configuration sets up secret passwords on communication servers A, B, and C, thus enabling the three communication servers to connect to each other.

To authenticate connections between communication servers A and B, enter the following commands:

- On communication server A:

  ```
  username B password a-b_secret
  ```

- On communication server B:

  ```
  username A password a-b_secret
  ```

To authenticate connections between communication servers A and C, enter the following commands:

- On communication server A:

  ```
  username C password a-c_secret
  ```

- On communication server C:

  ```
  username A password a-c_secret
  ```

To authenticate connections between communication servers B and C, enter the following commands:

- On communication server B:

  ```
  username C password b-c_secret
  ```

- On communication server C:

  ```
  username B password b-c_secret
  ```

When you specify an encryption type of 0 to enter an unencrypted password, the system displays the encrypted version of the password. For example, suppose you enter the following command:

```
username bill password westward
```

The system would display this command:

```
username bill password 7 21398211
```

The encrypted version of the password is 21398211. The password was encrypted by the Cisco-defined encryption algorithm, as indicated by the “7.”

If you were to enter the following command, the system would assume that the password is already encrypted and would do no encryption. It would display the command exactly as you typed it:

```
username bill password 7 21398211
username bill password 7 21398211
```