Chapter 4
Configuring the System

4

This chapter describes how to configure the system. These tasks include:

- Setting global system characteristics such as the host name and console banner message
- Defining the size of the system buffers
- Changing the system boot file specifications
- Establishing system and line passwords and system security
- Defining network services such as the IP Finger protocol
- Enabling and directing the logging of system debugging messages
- Configuring the console and virtual terminal lines

This chapter concludes with alphabetical summaries of the commands described in this chapter.

Configuring the Global System Parameters

The following sections contain procedures and command descriptions for configuring the global system characteristics: host name, and passwords, and configuring system security and system management functions. The global configuration commands described in the following sections are entered in configuration mode. See the section "Entering Configuration Mode" in the chapter "First-Time Startup and Basic Configuration" for the procedures to enter into this mode.

Setting the Host Name

Use the **hostname** global configuration command to specify the host name for the network server, which is used in prompts and default configuration file names.

hostname *name*

The argument *name* is the new host name for the network server and is case sensitive. The default host name is *Gateway.*

Example:

This command changes the host name to *sandbox*.

```
hostname sandbox
```

Displaying Banner Messages

A banner is the message that the EXEC command interpreter displays whenever a user starts any EXEC process or activates a line. The general form of the **banner** command follows.

banner {**motd**|**exec**|**incoming**} *c text c*

The **motd**, **exec**, and **incoming** keywords control when the banner message is displayed. The use of these keywords is described in the following sections.

The argument *c* specifies a delimiting character of your choice. The argument *text* specifies the message to be shown on the screen whenever an interface line is activated.

Follow **banner** with one or more blank spaces and then type the delimiting character then one or more lines of text *text*, terminating the message with the second occurrence of the delimiting character. There is no limit to the amount of characters that can be used for the banner, with the exception of buffer limits and what is appropriate for a banner.

Example:

The following example uses the # character as a delimiting character:

```
banner motd #
Building power will be off from 7:00 AM until 9:00 AM this coming Tues-
day.
#
```

Note: You cannot use the delimiting character in the banner message.

Displaying a Message of the Day Banner

To specify a general-purpose message-of-the-day type banner, use the **banner motd** global configuration command.

banner motd *c text c*

This displays a message-of-the-day type banner whenever a line is activated, or when an incoming Telnet connection is created.

Note: The command **banner** is equivalent to the command **banner motd**, except that the banner is displayed on incoming connections.

Displaying a Banner with an EXEC Process

To be able to display a message when an EXEC process is created, use the **banner exec** global configuration command.

banner exec *c text c*

This command specifies a message to be displayed when an EXEC process is created (line activated, or incoming connection to VTY).

Displaying a Incoming Message Banner

To display an incoming message to a particular terminal line, use the **banner incoming** global configuration command.

banner incoming *c text c*

This specifies a message to be displayed on incoming connections to particular terminal lines, (for example, lines used for "milking machine" applications).

Note: Messages are never displayed on incoming stream type connections, since they might interfere with printer daemons.

The EXEC banner can be suppressed on certain lines using the **no exec-banner** line subcommand (described in the section "Suppressing Banner Messages" later in this chapter.) Lines so configured will *not* display the EXEC or MOTD banners when an EXEC is created.

Example:

This example illustrates how to display a message-of-the-day, and a message that will be displayed when an EXEC process is created. Use the **banner** global configuration commands and **no exec-banner** line subcommand to accomplish these settings.

```
! Both messages are inappropriate for the VTYs.
line vty 0 4
no exec-banner
!
banner exec /
This is Cisco Systems training group server.
Unauthorized access prohibited.
/
!
banner motd /
The server will go down at 6pm for a software upgrade
/
```

Setting the System Buffers

In normal system operation, there are several pools of different sized buffers. These pools grow and shrink based upon demand. Some buffers are temporary and are created and destroyed as warranted. Other buffers are permanently allocated and cannot be destroyed. The **buffers** command allows a network administrator to adjust initial buffer pool settings, as well as the limits at which temporary buffers are created and destroyed. It is normally not necessary to adjust these parameters; do so only after consulting with Cisco support personnel. Improper settings could adversely impact router performance. The full syntax of this command follows:

buffers {**small**|**middle**|**big**|**large**|**huge**} {**permanent**|**max-free**|**min-free**|**initial**} *number*
no buffers {**small**|**middle**|**big**|**large**|**huge**} {**permanent**|**max-free**| **min-free**|**initial**} *number*

First choose the keyword that describes the size of buffers in the pool—small, big, huge, etc. The default number of the buffers in a pool is determined by the hardware configuration, and can be displayed with the EXEC **show buffers** command.

The following keyword specifies the buffer management parameter to be changed, and can be one of the following arguments:

- **permanent**—The number of permanent buffers that the system tries to allocate. Permanent buffers are normally not deallocated by the system.
- **max-free**—The maximum number of free or unallocated buffers in a buffer pool.
- **min-free**—The minimum number of free or unallocated buffers in a buffer pool.
- **initial**—The number of additional temporary buffers which should be allocated when the system is reloaded. This can be used to insure that the router has necessary buffers immediately after reloading in a high traffic environment.

The argument *number* specifies the number of buffers to be allocated.

The **no buffers** command with appropriate keywords and argument restores the default buffer values.

Examples:

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

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

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

```
buffers medium max-free 200
```

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

```
buffers large initial 1
```

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

```
buffers huge permanent 1
```

To display statistics about the buffer pool on the network server, use the command **show buffers**. For more information, refer to the section “Monitoring System Processes” in the chapter “Managing the System.”

Setting Configuration File Specifications

This section describes the **boot** global configuration commands used to configure boot files. The **boot** command can be used to perform these tasks:

- Change default file names.
- Specify a server host for netbooting configuration files and boot image files.
- Specify the size buffer to configure for netbooting a host or network configuration file.

The commands to load files over the network take effect the next time the software is reloaded, provided they have been written into nonvolatile memory.

Changing the Network Configuration File

The network configuration file contains commands that apply to all network servers and terminal servers on a network. The default name of this file is *network-confg*. See the section “Entering Configuration Mode” in the chapter “First-Time Startup and Basic Configuration.” To change the name of this file use the **boot network** global configuration command. The full command syntax follows:

boot network *filename* [*address*]
no boot network [*filename address*]

The keyword **network** changes the network configuration file from *network-confg*. The argument *filename* is the new name for the network configuration file. If you omit the argument *address*, the network server uses the default broadcast address of *255.255.255.255*. If you use *address*, you can specify a specific network host or a subnet broadcast address.

Changing the Host Configuration File

The host configuration file contains commands that apply to one network server in particular. To change the host configuration file name, use the **boot host** global configuration command. The full command syntax follows:

boot host *filename* [*address*]
no boot host [*filename address*]

The keyword **host** changes the host configuration file name to a name you specify in the *filename* argument. The network server uses its name to form a host configuration file name. To form this name, the network server converts its name to all lowercase letters, removes all domain information, and appends “-confg.” By default, the host file name is *gateway-confg*.

Obtaining the Boot File Over the Network

New versions of the software can be downloaded over the network. Use the **boot system** global configuration command to do this. The full command syntax follows.

boot system *filename* [*address*]
no boot system [*filename address*]

The keyword **system** indicates that the file name and host addresses for booting operating software over the network are in the nonvolatile memory. In this case, the argument *filename* is the file name of the operating software to load, and the argument *address* is the address of the network host holding that file.

The **boot system** command overrides the processor configuration register setting unless the register specifies the use of default (ROM) operating software. Therefore, to permit netbooting, set the configuration register bits on the processor card to any pattern other than 0-0-0-0 or 0-0-0-1.

Note: The Cisco software boots images over a network by using one system image to load another system image. This means that there must be enough room in memory for two complete system images. Some versions of the software are so large that two copies of it will not fit in memory. Therefore, the CSC/2 netboot algorithm uses a secondary bootstrap system image to netboot the desired system image. (You need Software Release 8.0 or later EPROMs to use this secondary bootstrap.)

The secondary bootstrap is a very small system image which is netloaded and invoked to netboot the desired system image. The secondary bootstrap for CSC/2 processors is named *boot-csc2*. A secondary bootstrap is *not* required for the CSC/3 processor, since it has enough memory to net boot any CSC/3 image.

If bit 9 of the configuration register is set (the factory default), use of the secondary bootstrap is enabled. Contact Cisco Customer Service for copies of the secondary bootstrap software.

Refer to the Cisco publications *Modular Products Hardware Installation and Reference* or the *IGS Hardware Installation and Reference* for more information about the processor configuration registers.

Note: The IGS requires four megabytes of RAM to netboot.

Example:

To use the nonvolatile memory option to specify netbooting, place a **boot system** command in the nonvolatile memory. You use this command to specify both the file name of the operating software to load, and the Internet address of the server host holding that file:

```
boot system /usr/local/tftpdir/cisco.ts2 192.7.31.19
```

Manually Booting from ROM

Use the **b** command at the ROM monitor prompt (>) to boot the system manually from the ROM software. The syntax is as follows:

b

The following is an example of using the **b** command to manually boot from ROM:

```
>b
F3:
{ROM Monitor copyrights}
```

Manually Netbooting

Use the **b** command at the ROM monitor prompt (>) to netboot the system manually. Check the appropriate hardware manual for the correct jumper or configuration register setting. The syntax for TFTP netbooting is as follows:

b *filename* [*address*]

The *filename* argument specifies the filename of the image you want loaded. It is case sensitive. The *address* argument is optional and defines the IP address of the host you want to boot from. The following is an example of the **b** command for manually netbooting:

```
>b testme4.test 131.108.15.112
F3:
{ROM Monitor copyrights}
```

Specifying a Boot File Buffer Size

To specify the size of the buffer to be used for netbooting a host or a network configuration file, use the **boot buffersize** global configuration command. The full command syntax follows:

boot buffersize *bytes*
no boot buffersize *bytes*

The argument *bytes* specifies the size of the buffer to be used. By default it is the size of your non-volatile memory, or 32 kilobytes if you do not have nonvolatile memory. There is no minimum or maximum size that may be specified.

The EXEC commands **write terminal** and **write network** use the information specified by the **buffersize** keyword when performing their functions (see the section “Entering Configuration Mode” in the chapter “First-Time Startup and Basic Configuration” for more information about these EXEC commands).

Configuring Multiple Instances of the Boot Commands

You can configure multiple instances of the **boot** commands. When issued, each command is executed in order and so can be used to begin a systematic search or to build a specific list. For example, you can issue multiple **boot** commands to build an ordered list of configuration-file-name-and-host-address pairs. The network server scans this list until it successfully loads the appropriate network or host configuration file or system boot image. In this example, the network server looks first for *fred-confg* on network 192.31.7.24 and, if it cannot load that file, then for *wilma-confg* on network 192.31.7.19:

```
boot host /usr/local/tftpdir/fred-confg 192.31.7.24
boot host /usr/local/tftpdir/wilma-confg 192.31.7.19
```

Note: This example uses fictitious file names; the syntax of these file names depends on the TFTP server you are loading the files from.

If the network server cannot find either file, a background process tries at ten-minute intervals (default) to load one or the other of the files.

You may issue multiple instances of all variations of the **boot** command, including the **no boot** forms. This feature can be useful for removing configuration files. To remove a configuration file-name and host-address pair from the list, use the **no boot** command syntax.

Troubleshooting Information when Netbooting

Cisco routers support netbooting over both TFTP and MOP across all supported media types, such as Ethernet, FDDI, serial, Token Ring, and HSSI. During a netbooting session, routers behave like hosts: they route via proxy ARP or a default gateway. However, when netbooting, a router ignores routing information, static IP routes, and bridging information. As a result, intermediate routers are responsible for handling ARP and TFTP requests correctly. For serial and HSSI media, ARP is not used.

If you need to netboot from a server, it is recommended that you “ping” the server from the ROM software. If you are unable to ping the server, there is a problem with the server configuration or hardware. Contact your technical support representative for assistance. See “Useful Information to Provide Technical Support” later in this section for details.

The sections that follow contain solutions to common problems that occur when netbooting. Note that these solutions apply only if you were able to ping the server successfully.

Client ARP Request Times Out

When netbooting, the client you netboot from sends an ARP request to the server over every available appropriate network interface (such as an Ethernet port or a Token Ring port). The client expects the server or a router to return an ARP response. If the client does not receive an ARP response from the server or a router, a message similar to the following is displayed on the client console:

```
Booting gs3-bfx..................[timed out]
```

One possible cause of no ARP response is that intermediate routers are not performing proxy ARP. Look for **no ip proxy-arp** in the configuration of the intermediate router. Another possible cause is that the client is using a broadcast address and the intermediate router does not have an IP helper address defined that points to the TFTP server.

Timeouts and Out-of-Order Packets

When netbooting, it is not unusual for the client to send additional requests before receiving a response to the initial ARP request. This can result in timeouts, out-of-order packets, and multiple responses. Timeouts (shown as periods on a netbooting display) and out-of-order packets (shown as 0s) do not necessarily prevent a successful boot. It is acceptable to have timeouts and out-of-order packets. The following examples show successful boots even though a timeout and out-of-order packets have occurred:

```
Booting gs3-bfx from 131.108.1.123: !.!!!!!!!!!!!!!!!!!!!!!!

Booting gs3-bfx from 131.108.1.123: !0.0!!!!!!!!!!!!!!!!!!!!!
```

Note that intermittent timeouts and out-of-order packets may occur throughout a netbooting session without being cause for concern. Excessive timeouts and out-of-order packets can be caused by bad routing paths on the intermediate routers, an extremely slow server, multiple path problems or noise on the line. If your netbooting session appears to have excessive timeouts and out-of-order packets, contact your technical support representative and report the problem. Before calling technical support, you need to gather some information. See the following section, “Useful Information to Provide Technical Support,” for details.

Useful Information to Provide Technical Support

Collect the following information for the technical support representative:

- ROM images
- NVRAM configurations for client and adjacent routers
- Debugging output from the adjacent router using the following commands:
    - **debug arp**
    - **debug ip-udp**
    - **debug tftp**

Establishing Passwords and System Security

This section describes how to configure password protection and terminal access security.

You may set passwords to control access to the privileged command level and to individual lines. The Terminal Access-Controller Access System (TACACS) protocol controls terminal use by means of a user-ID-and-password pair. The Defense Data Network developed TACACS to control access to its TAC terminal servers; Cisco patterned its TACACS support after the DDN application.

These system security measures may not provide the level of protection needed for some environments; individual routing protocols and bridging support may have additional security procedures. You may also need to use access lists for additional protection. For procedures for configuring access lists, refer to the sections describing configuration of a particular routing protocol or bridging support.

Establishing the Privileged-Level Password

To assign a password for the privileged command level, use the **enable password** global configuration command:

enable password *password*

The argument *password* is case sensitive and specifies the password prompted for in response to the EXEC command **enable.** The *password* argument may contain any alphanumeric characters, including spaces, up to 80 characters. The password checking is also case sensitive. The password *Secret* is different than the password *secret*, for example, and the password *two words* is an acceptable password.

Note: On systems with software Release 8.2 and earlier, the command syntax was **enable-password**.

To enter the privileged command level, type the following EXEC command and then press Return:

enable

Next, type the password for the privileged command level at the Password: prompt.

When you use the **enable** command at the console terminal, the EXEC does not prompt you for a password if the privileged mode password is not set. Additionally, if the **enable** password is not set and the line 0 (console line) password is not set, then it is only possible to enter privileged mode on the console terminal. This feature allows you to use physical security rather than passwords to protect privileged mode if that is what you choose to do.

If neither the **enable** password nor the line 0 (console) password is set, it is possible to enter privileged command mode either without entering a password at the console terminal or by entering the console line password when prompted while using any other line.

Example:

The following example sets the password *secretword* for the privileged command level on all lines, including the console:

```
enable password secretword
```

Specifying a Password

When an EXEC is started on a line with password protection, the EXEC prompts for the password. If you enter the correct password, the EXEC prints its normal nonprivileged prompt. You may try three times to enter a password before the EXEC exits and returns the terminal to the idle state.

To specify a password, use the **password** line subcommand. The full command syntax follows:

password *text*
no password

The *text* argument may contain any alphanumeric character, including spaces, up to 80 characters. The password checking is also case sensitive. The password *Secret* is different than the password *secret*, for example, and the password *two words* is an acceptable password.

To enable checking for the password specified by the **password** command, use the line subcommand **login**:

login

Alternatively, to use the TACACS user ID and password-checking mechanism instead, use the following subcommand**:**

login tacacs

To disable all password checking, use the command:

no login

The server prints the message-of-the-day banner before prompting for a password, so you immediately see messages such as no trespassing notifications. By default, virtual terminals require a password. If you do not set a password for a virtual terminal, it will respond to attempted connections by displaying an error message and closing the connection. Use the **no login** subcommand to disable this behavior and allow connections without a password.

Example:

The following example sets the password *letmein* on line 5:

```
line 5
password letmein
login
```

Recovering from a Lost Password

If your network server has the nonvolatile memory option, you can lock yourself out if you enable password checking on the console terminal line and then forget the line password.

To recover from this, force the network server into factory diagnostic mode by turning off the network server, inserting a jumper in bit 15 of the processor configuration register, (or bit 7 of the processor configuration register in the IGS or CRM), and turning on the network server. Follow these steps.

Step 1: You will be asked if you want to set the manufacturers' addresses. Respond by typing "Yes." You then see the following prompt:

```
TEST-SYSTEM>
```

Step 2: Type the **enable** command to get the privileged prompt:

```
TEST-SYSTEM#enable
```

Step 3: Type the **show configuration** command to review the system configuration and find the password.

Step 4: To resume normal operation, turn off the network server, remove the jumper from bit 15 (or bit 7) of the configuration register, and turn on the network server again.

Step 5: Log in to the network server with the password that was shown in the configuration file.

The processor configuration registers are described in Appendix A, "The CPU Bootstrap Program" in the *Modular Products Hardware Installation and Reference* publication, or Appendix A, "IGS Configuration Register" in the *IGS Hardware Installation and Reference* publication.

When the network server restarts in factory diagnostic mode, it does not read the nonvolatile memory, thus avoiding the command to set a password for the console terminal. Do not change anything in the factory diagnostic mode.

Note: All debugging capabilities are turned on during diagnostic mode.

Establishing Terminal Access Control

Cisco Systems provides unsupported versions of both a standard and an extended TACACS server. The servers run on most UNIX systems available from Cisco using FTP on the *ftp.cisco.com* directory. You may use the servers to create UNIX accounting applications that monitor use of a system and user logins.

The configuration commands in the following sections tailor the behavior of the standard TACACS server.

Setting the Server Host Name

The **tacacs-server host** global configuration command specifies a TACACS host. The full syntax of this command follows.

> **tacacs-server host** *name*
> **no tacacs-server host** *name*

The argument *name* is the name or Internet address of the host. You can use multiple **tacacs-server host** subcommands to specify multiple hosts. The server will search for the hosts in the order you specify them. The **no tacacs-server host** global configuration command deletes the specified name or address.

Limiting Login Attempts

The **tacacs-server attempts** global configuration command controls the number of login attempts that may be made on a line set up for TACACS verification.

> **tacacs-server attempts** *count*
> **no tacacs-server attempts**

The argument *count* is the number of attempts. The default is three attempts.

The **no tacacs-server attempts** global configuration command restores the default.

Example:

This command changes the login attempt to just one try:

```
tacacs-server attempts 1
```

Controlling Retries

The **tacacs-server retransmit** global configuration command specifies the number of times the server will search the list of TACACS server hosts before giving up. The server will try all servers, allowing each one to time-out before increasing the retransmit count.

> **tacacs-server retransmit** *retries*
> **no tacacs-server retransmit**

The argument *retries* is the retransmit count. The default is two retries.

The **no tacacs-server retransmit** global configuration command restores the default.

Example:

This command specifies a retransmit counter value of five times:

```
tacacs-server retransmit 5
```

Setting the Timeout Intervals

The **tacacs-server timeout** global configuration command sets the interval the server waits for a server host to reply.

tacacs-server timeout *seconds*
no tacacs-server timeout

The argument *seconds* specifies the number of seconds. The default interval is five seconds. The **no tacacs-server timeout** global configuration command restores the default.

Example:

This command changes the interval timer to ten seconds:

```
tacacs-server timeout 10
```

Setting the Last Resort Login Feature

If, when running the TACACS server, the TACACS server does not respond, the default action is to deny the request. Use the **tacacs-server last-resort** global configuration command to change that default.

tacacs-server last-resort {**password**|**succeed**}
no tacacs-server last-resort {**password**|**succeed**}

The command causes the network server to request the privileged password as verification, or forces successful login without further input from the user, depending upon the keyword specified, as follows:

- **password**—Allows the user to access the privileged-level command mode by entering the password set by the **enable** command.
- **succeed**—Allows the user to access the privileged-level command mode without further question.

Note: The last resort login feature can be useful when it is important to be able to ensure that login can occur. An example of such a condition is when a systems administrator needs to login in order to troubleshoot TACACS servers which are down.

The **no tacacs-server last-resort** global configuration command restores the system to the default behavior.

Establishing Privileged-Level TACACS

The following variations of the **enable** command may be used to configure privileged-level command access using the TACACS protocol.

Enabling the Privileged Mode

The **enable use-tacacs** global configuration command is used for setting the TACACS protocol for determining whether a user can access the privileged command level.

enable use-tacacs
no enable use-tacacs

If you use this command, the EXEC **enable** command will ask for both a new user name and password. This 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.

Note: When used without extended TACACS, this command allows anyone with a valid user name 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.

Enabling the Privileged Mode Last Resort Login Feature

The **enable last-resort** global configuration command allows you to specify what happens if the TACACS servers used by the **enable** command do not respond.

enable last-resort {**password**|**succeed**}
no enable last-resort {**password**|**succeed**}

The default action is to fail. Use of the keyword changes the action, as follows:

- **password**—Allows you to enable by entering the privileged command level password.
- **succeed**—Allows you to enable without further question.

The **no enable last-resort** global configuration command restores the default.

Configuring TACACS Accounting

What follows are the configuration commands that tailor the behavior of the extended TACACS client.

Enabling Extended TACACS Mode

The **tacacs-server extended** global configuration command enables an extended TACACS mode.

tacacs-server extended
no tacacs-server extended

This mode provides information about the terminal requests for use in setting up host auditing trails and accounting files for tracking use of terminal servers and routers. Information includes responses from terminal servers and routers, and validation of user requests. An unsupported, extended TACACS server is available from Cisco Systems via anonymous FTP for UNIX users who want to create the auditing programs.

The **no tacacs-server extended** command disables this mode.

Login Notification

The **tacacs-server notify** global configuration command causes a message to be transmitted to the TACACS server, with retransmission being performed by a background process for up to five minutes. The terminal user, however, receives an immediate response allowing access to the terminal. The full syntax of this command follows.

tacacs-server notify {**connect**|**slip**|**enable**|**logout**}
no tacacs-server notify {**connect**|**slip**|**enable**|**logout**}

The keywords specify notification of the TACACS server whenever a user does one of the following:

- **connect**—User makes TCP connections.
- **slip**—User turns SLIP on or off (Terminal Server only).
- **enable**–User enters the **enable** command.
- **logout**—User logs out.

The **no tacacs-server notify** command with the appropriate keyword disables notification.

Note: When used with extended TACACS, the command **tacacs-server notify enable** allows anyone with a valid user name and password to access the privileged-level command mode.

Login Authentication

The **tacacs-server authenticate** command requires a response from the network or communications server to indicate whether the user may perform the indicated action.

tacacs-server authenticate {**connect**|**slip**|**enable**}
no tacacs-server authenticate {**connect**|**slip**|**enable**}

Actions that require a response include the following, specified as optional keywords:

- **connect**—User TCP connections
- **slip**—SLIP connections (Terminal Server only)
- **enable**—Use of **enable** command

The **no tacacs-server authenticate** command with the appropriate keyword disables the action.

Username Authentication

Networks that cannot support a TACACS service may still wish to use a username-based authentication system. In addition, it may be useful to define special usernames that get special treatment (for example, an "info" username that does not require a password, but connects the user to a general purpose information service).

The network server software supports these needs by implementing a local **username** configuration command. The format for the command follows:

> **username** *name* **[nopassword|password** *encryptiontype* **password]**
> **username** *name* **[accesslist** *number***]**
> **username** *name* **[autocommand** *command***]**
> **username** *name* **[noescape] [nohangup]**

Multiple **username** commands can be used to specify options for a single user.

The **nopassword** keyword means that no password is required for this user to log in. This is usually most useful in combination with the **autocommand** keyword.

The **password** keyword specifies a possibly encrypted password for this username.

The *encryptiontype* argument is a single-digit number. Currently defined encryption types are 0, which means no encryption, and 7, which specifies a Cisco-specified encryption algorithm. Passwords entered unencrypted are written out with the Cisco encryption. A password, which can contain imbedded spaces, must be the last option specified in the **username** command.

The **accesslist** keyword specifies an outgoing access list that overrides the access list specified in the **access class** line configuration subcommand. It is used for the duration of the user's session. The access list number is specified by the *number* argument.

The **autocommand** keyword causes the command specified by the *command* argument to be issued automatically after the user logs in. When the command is complete, the session is terminated. Since the command can be any length and contain imbedded spaces, commands using the **autocommand** keyword must be the last option on the line.

The **nohangup** keyword prevents the network server from disconnecting the user after an automatic command (set up with the **autocommand** keyword) has completed. Instead, the user gets another login prompt.

The **noescape** keyword prevents a user from using an escape character on the host to which the user is connected.

Examples:

To implement a service similar to the UNIX **who** command, which can be given at the login prompt and lists the current users of the network server, the command takes the following form:

```
username who nopassword nohangup autocommand show users
```

To implement an ID that will work even if all the TACACS servers break, the command is as follows:

```
username superuser password superpassword
```

Configuring the Simple Network Management Protocol (SNMP)

The Simple Network Management Protocol (SNMP) provides a way to access and set configuration and run time parameters for the network server. Cisco System's implementation of SNMP is compatible with RFCs 1155, 1156, and 1157. The Cisco Management Information Base (MIB) supports RFC 1213 and provides Cisco-specific variables.

A separate document, available in RFC 1212-type (MIB II) format, describes all the Cisco-specific SNMP variables in the Cisco portion of the MIB. It also describes what is required to establish minimum configuration. Contact Cisco Systems to obtain a copy of this document, which includes instructions for accessing the variables using SNMP.

Enabling and Disabling the SNMP Server

To be able to configure the SNMP server, you need to be in the configuration command collection mode. You enter this mode using the EXEC command **configure** at the EXEC prompt. See the section "Entering Configuration Mode" in the chapter "First-Time Startup and Basic Configuration" for a description of the procedure.

You begin SNMP operation by entering the configuration commands that define the desired operation. To disable SNMP server operations on the network server after it has been started, use the **no snmp-server** global configuration command:

no snmp-server

Defining the SNMP Server Access List

To set up an access list that determines which hosts can send requests to the network server, use the **snmp-server access-list** global configuration command. The full command syntax follows.

snmp-server access-list *list*
no snmp-server access-list *list*

This command sends all traps to the host. The network server ignores packets from hosts that the access list denies.

The argument *list* is an integer from 1 through 99 that specifies an IP access list number. The access list applies only to the global read-only SNMP agent configured with the command **snmp-server community**.

The **no snmp-server access-list** global configuration command removes the specified access list.

Example:
This command sends traps to all hosts defined by access list 21:

```
snmp-server access-list 21
```

Setting the System Contact String

To set the system contact string (syscontact), use the **snmp-server contact** command. The command syntax follows:

snmp-server contact *text*

The *text* argument is a string that specifies the system contact information.

Setting the System Location String

To set the system location string, use the **snmp-server location** command. The command syntax follows:

snmp-server location *text*

The *text* argument is a string that specifies the system location information.

Setting the Community String

To set up the community access string, use the **snmp-server community** global configuration command. The full command syntax follows:

snmp-server community [*string* [**RO**|**RW**] [*list*]]
no snmp-server [**community** [*string*]]

This command enables SNMP server operation on the network server. The argument *string* specifies a community string that acts like a password and permits access to the SNMP protocol.

By default, an SNMP community string permits read-only access (keyword **RO**); use the keyword **RW** to allow read-write access. The optional argument *list* is an integer from 1 through 99 that specifies an access list of Internet addresses that may use the community string.

The **no snmp-server community** global configuration command removes the specified community string or access list.

Examples:

This command assigns the string *comaccess* to the SNMP server, allows read-only access, and specifies that addresses that match the criteria in access list 4 may use the community string. (Notice that the string is entered *without* quotes or any other parsing characters.)

```
snmp-server community comaccess RO 4
```

In this example, *any* host using the community string *braves* has SNMP read-only access to the router:

```
snmp-server access-list 1
snmp-server community bluejay rw
snmp-server community braves ro
!
access-list 1 permit 142.111.131.1
access-list 1 deny 0.0.0.0 255.255.255.255
```

This example configuration restricts SNMP read-only access to only the host at 142.111.131.1:

```
snmp-server community bluejay rw
snmp-server community braves ro 1
!
access-list 1 permit 142.111.131.1
access-list 1 deny 0.0.0.0 255.255.255.255
```

In comparing the two examples, consider that the second example restricts read-only access to *only* the host at 142.111.131.1. The first example allows *any* host SNMP read-only access if it has the community string, and disallows SNMP read-only access to all hosts with the community string *braves* except the host 142.111.131.1.

Establishing the Message Queue Length

To establish the message queue length for each TRAP host, use the **snmp-server queue-length** global configuration command:

snmp-server queue-length *length*
no snmp-server queue-length

This command defines the length of the message queue for each TRAP host.

The argument *length* is the number of TRAP events that can be held before the queue must be emptied; the default is 10. Once a TRAP message is successfully transmitted, software will continue to empty the queue, but never faster than at a rate of four TRAP messages per second.

The **no snmp-server queue-length** command resets the queue length to its default value of 10.

Example:

This command establishes a message queue that traps four events before it must be emptied:

```
snmp-server queue-length 4
```

Establishing Packet Filtering

To establish the packet filtering size, use the **snmp-server packetsize** global configuration command. The full command syntax follows:

snmp-server packetsize *bytes*
no snmp-server packetsize

This command allows control over the largest SNMP packet size permitted when the SNMP server is receiving a request or generating a reply.

The argument *bytes* is a byte count from 484 through 8,192. The default is 484. The **no snmp-server packetsize** command resets this default.

Example:
This command establishes a packet filtering maximum size of 1024 bytes:

```
snmp-server packetsize 1024
```

Establishing the Trap Message Recipient

To specify the recipients of trap messages, use the **snmp-server host** global configuration command. The full syntax follows.

snmp-server host *address community-string* [**snmp**] [**tty**]
no snmp-server host *address*

This command specifies which host or hosts should receive trap messages. You need to issue the **snmp-server host** command once for each host acting as a trap recipient.

The argument *address* is the name or Internet address of the host. The argument *community-string* is the password-like community string set with the **snmp-server community** command.

The following optional keywords define which traps are sent:

- **snmp**—Enables the SNMP traps described in RFC 1157.
- **tty**—Enables the Cisco enterprise-specific trap when a TCP connection closes.

If you do not specify any optional keywords, the sending of all trap types is enabled.

If you specify multiple **snmp-server host** commands for a given host or address, the community string used is the one on the last command line you entered, and the traps sent are a combination of all the optional keywords you specified.

The **no snmp-server host** command removes the specified host.

Note: Previous versions of the software enabled traps by default. Traps are now disabled by default.

Examples

This command sends all SNMP traps to 131.108.2.160:

```
snmp-server host 131.108.2.160
```

To turn these trap messages off, use the **no snmp-server host** command:

```
no snmp-server host 131.108.2.160
```

The following example causes all the SNMP traps to be sent to the host specified by the name *cisco.com*. The community string is defined to be *comaccess*.

```
snmp-server host cisco.com comaccess snmp
```

Examples: Specifying Multiple snmp-server host Commands

Suppose the initial configuration is as follows:

```
snmp-server host 131.108.2.3 public snmp
```

You then enter the following configuration command:

```
snmp-server host 131.108.2.3 private
```

This results in the following configuration, which uses the community string you specified last and the trap type **snmp**:

```
snmp-server host 131.108.2.3 private snmp
```

Starting again with the initial configuration, suppose you enter the following command:

```
snmp-server host 131.108.2.3 notpublic tty
```

This results in the following configuration, which uses the community string you specified last and the trap types **snmp** and **tty**:

```
snmp-server host 131.108.2.3 notpublic snmp tty
```

To modify the initial configuration so that only **tty** traps are sent, enter the following commands:

```
no snmp-server host 131.108.2.3
snmp-server host 131.108.2.3 public tty
```

Establishing the TARP Message Timeout

To define how often to try resending trap messages on the retransmission queue, use these global configuration commands:

snmp-server trap-timeout *seconds*
no snmp-server trap-timeout

The argument *seconds* sets the interval for resending the messages. The default is set to 30 seconds. The **no snmp-server trap-timeout** command restores this default.

Example:

This command sets an interval of 20 seconds to try resending TRAP messages on the retransmission queue:

```
snmp-server trap-timeout 20
```

Enabling SNMP System Shutdown Feature

Using SNMP packets, a network management tool can send messages to users on virtual terminals and the network server's console. This facility operates in a similar fashion to the SNMP **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.

Requesting a *shutdown-after-message* is similar to issuing a **send** command followed by a **reload** command. Because the ability to cause a reload from the network is a powerful feature, it is protected by this configuration command. To use this SNMP message reload feature the device configuration must include the **snmp-server system-shutdown** global configuration command. The full command syntax follows:

snmp-server system-shutdown
no snmp-server system-shutdown

The **no snmp-server system-shutdown** option prevents a SNMP system-shutdown request (from an SNMP manager) from resetting the Cisco agent.

To understand how to use this feature with SNMP requests, read the document *mib.txt* available by anonymous FTP from ftp.cisco.com. This document is available in RFC 1213-type format. It describes all the Cisco-specific SNMP variables in the Cisco portion of the MIB. It also describes what is required to establish minimum configuration. Contact Cisco Systems to obtain a copy of this document, which includes instructions for accessing the variables using SNMP.

Configuring the Trivial File Transfer Protocol (TFTP) Server

You can configure the network server to act as a limited Trivial File Transfer Protocol (TFTP) server from which other Cisco servers can boot their software. As a TFTP server host, the network server responds to TFTP read request messages by sending a copy of its ROM software to the requesting host. The TFTP read request message must use the file name that you specified in the network server configuration.

To specify TFTP server operation for a communications server, use the **tftp-server system** global configuration command. The full syntax follows.

tftp-server system *filename list*
no tftp-server system *filename list*

This command has two arguments: *filename* and *list*. The argument *filename* is the name you give the communications server ROM file, and the argument *list* is an IP access-list number.

The system sends a copy of the ROM software to any host which issues a TFTP read request with this file name. To learn how to specify an access list, see the "Configuring IP Access Lists" section in the chapter "Routing IP."

You can specify multiple file names by repeating the **tftp-server system** command. To remove a previously defined file name, use the **no tftp-server system** command and append the appropriate file name and an access-list number.

Images that run from ROM, including IGS images, cannot be loaded over the network. Therefore, it does not make sense to use TFTP to offer the ROMs on these images.

Example:

This command causes the router to send, via TFTP, a copy of the ROM software when it receives a TFTP read request for the file *configfile*. The requesting host is checked against access list 22.

```
tftp-server system configfile 22
```

Tailoring Use of Network Services

The **service** global configuration command tailors use by the network server of network-based services. Some **service** commands also configure system defaults; see **decimal-tty** for an example. The full command syntax follows:

service *keyword*
no service *keyword*

The argument *keyword* is one of the following:

- **config**—Specifies TFTP autoloading of configuration files; disabled by default on system with nonvolatile memory.
- **decimal-tty**—Specifies that line numbers be displayed and interpreted as decimal numbers rather than octal numbers; disabled by default.
- **finger**—Allows Finger protocol requests (defined in RFC 742) to be made of the network server; enabled by default. This service is equivalent to issuing a remote **show users** command.
- **tcp-keepalives-**{**in**|**out**}—Generates keepalive packets on idle network connections. The **in** keyword generates them on incoming connections (initiated by remote host); the **out** keyword generates them on outgoing connections (initiated by a user). There is a column in the EXEC **show tcp** display showing the keepalive statistics. The `wakeups` row shows how many keepalives have been transmitted without receiving any response (this is reset

to 0 when a response is received).

The **no service** command disables the specified service or function.

Example:

The following command enables TFTP autoloading of configuration files:

```
service config
```

Redirecting System Error Messages

By default, the network server sends the output from the EXEC command **debug** and system error messages to the console terminal.

To redirect these messages, as well as output from asynchronous events such as interface transition, to other destinations, use the **logging** configuration command options.

These destinations include the console terminal, virtual terminals, and UNIX hosts running a syslog server; the syslog format is compatible with 4.3 BSD UNIX.

To configure the logging of messages, you need to be in the configuration command collection mode. To enter this mode, use the EXEC command **configure** at the EXEC prompt (see the section "Entering Configuration Mode" in the chapter "First-Time Startup and Basic Configuration" for the procedure).

The following sections describe how to implement these redirection options.

Enabling Message Logging

To enable or disable message logging, use the following global configuration commands:

logging on
no logging on

The **logging on** command enables message logging to all supported destinations other than the console. This behavior is the default.

The **no logging on** command enables logging to the console terminal only.

Logging Messages to an Internal Buffer

The default logging device is the console; all messages are displayed on the console unless otherwise specified.

To log messages to an internal buffer, use the logging buffered global configuration command. The full command syntax follows.

logging buffered
no logging buffered

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

The **no logging buffered** command cancels the use of the buffer and writes messages to the console terminal, which is the default.

Logging Messages to the Console

To limit how many messages are logged to the console, use the **logging console** global configuration command. The full syntax of this command follows:

logging console *level*
no logging console

The **logging console** command limits the logging messages displayed on the console terminal to messages with a level at or above the specified severity, which is specified by the *level* argument.

The argument *level* can be one of the following keywords, listed here in order from the most severe to the least severe level.

- **emergencies**—System unusable
- **alerts**—Immediate action needed
- **critical**—Critical conditions
- **errors**—Error conditions
- **warnings**—Warning conditions (output from **debug** commands are logged at this level)
- **notifications**—Normal but significant conditions
- **informational**—Informational messages only
- **debug**—Debugging messages

The default is to log messages at the **warnings** level to the console.

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

Example:
This command sets console logging of messages at the debug level:

```
logging console debug
```

Logging Messages to Another Monitor

To limit the level of messages to log to the terminal lines (monitors), use logging monitor command. The full syntax of this command follows.

logging monitor *level*
no logging monitor

The **logging monitor** command limits the logging messages displayed on terminal lines other than the console line to messages with a level at or above *level*. The argument *level* is one of the keywords described for the **logging console** command in the previous section,

"Logging Messages to the Console." To display logging messages on a terminal, use the privileged EXEC command **terminal monitor.**

The **no logging monitor** command disables logging to terminal lines other than the console line.

Example:

This command sets the level of messages displayed on monitors other than the console to notifications:

```
logging monitor notifications
```

Logging Messages to a UNIX Syslog Server

To log messages to the syslog server host, use the **logging** global configuration command. The full syntax is as follows:

logging *internet-address*
no logging *internet-address*

The **logging** command identifies a syslog server host to receive logging messages. The argument *internet-address* is the Internet address of the host. By issuing this command more than once, 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.

Limiting Messages to a Syslog Server

To limit how many messages are sent to the syslog servers, use the **logging trap** global configuration command. Its full syntax follows:

logging trap *level*
no logging trap

The **logging trap** command limits the logging messages sent to syslog servers to messages with a level at or above *level*. The argument *level* is one of the keywords described for the **logging console** command in the earlier section, "Logging Messages to the Console."

To send logging messages to a syslog server, specify its host address with the **logging** command.

The **no logging trap** command disables logging to syslog servers.

The current software generates four categories of the syslog messages:

1. Error messages about software or hardware malfunctions, displayed at the errors level.
2. Output from the **debug** commands, displayed at the warnings level.
3. Interface up/down transitions and system restart messages, displayed at the notifications level.
4. Reload requests and low-process stack messages, displayed at the informational level.

The EXEC command **show logging** displays the addresses and levels associated with the current logging setup. The command output also includes ancillary statistics.

Example:

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.debug /usr/adm/logs/tiplog
```

The `local7` keyword specifies the logging facility to be used.

The `debug` argument specifies the syslog level. See the previous *level* arguments list for other arguments that can be listed.

The UNIX system sends messages at or below this 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.

Configuring Console and Virtual Terminal Lines

To configure your console and virtual terminal lines you need to be in the configuration command collection mode. To enter this mode, use the EXEC command **configure** at the EXEC prompt (see the section “Entering Configuration Mode” in the chapter “First-Time Startup and Basic Configuration” for the procedure).

Starting Line Configuration

To start configuring a terminal line, use the **line** command. This command identifies a specific line for configuration and starts line configuration command collection.

The **line** command has the following syntax:

line [*type-keyword*] *first-line* [*last-line*]

This command can take up to three arguments: a keyword, a line number, or a range of lines numbers.

The optional argument *type-keyword* specifies the type of line to be configured; it is one of the following keywords:

- **console**—Console terminal line
- **aux**—Auxiliary line, (described in the following section)
- **vty**—Virtual terminal for remote console access

When the line type is specified, the argument *first-line* is the relative number of the terminal line (or the first line in a contiguous group) you want to configure. Numbering begins with zero.

The optional argument *last-line* then is the relative number of the last line in a contiguous group you want to configure.

If you omit *type*, then *first-line* and *last-line* are absolute rather than relative line numbers. To display absolute line numbers, use the EXEC command **show users all.**

The network server displays an error message if you do not specify a line number.

Note: Line numbers, by default, are octal on the network servers.

The **line** command enables you to easily configure a large group of lines all at once. After you set the defaults for the group, you can use additional **line** commands and subcommands to set special characteristics, such as location, for individual terminal lines.

Example:

The following command starts configuration for the first five virtual terminal lines:

```
line vty 0 4
```

Configuring the CPU Auxiliary Port

The **line** command keyword **aux** allows use of an auxiliary RS-232 DTE port available on all processor cards. Use this port to attach to an RS-232 port of a CSU/DSU, protocol analyzer, or modem. You can monitor that port remotely by connecting to the TCP port whose number is 2000 decimal plus the line number of the auxiliary port. For example, if the auxiliary port was line 1 (obtained from the EXEC command **show users all**), then the TCP port would be 2001. You must order a special cable from Cisco Systems for use with this auxiliary port.

Note: You cannot use the auxiliary port as a second console port, nor can you initiate connections from this port. Its purpose is to *receive* connections from remote systems.

To configure the auxiliary port, use this variation of the **line** command:

line aux *port-address*

When configuring the auxiliary port, address it as line 0, as in this sample:

```
line aux 0
```

The auxiliary ports assert DTR only when a Telnet connection is established. The console port does not use RTS/CTS handshaking for flow control.

By default, the auxiliary port supports an EXEC process. This default can be re-enabled using the **exec** line subcommand.

Example:

These commands configure the auxiliary port with a line speed of 2400 baud, and enable the EXEC.

```
line aux 0
exec
speed 2400
```

No modem control signals are supported on this line. If an auto-answer modem is configured on the line, you must dial up, log in, then hang up. The DTR signal will be active whenever an EXEC is configured on the auxiliary port.

Note: The EXEC is still present and may be used by the next person that dials into the number. This could cause security problems.

Establishing Line Passwords

When you start an EXEC on a line with password protection, the EXEC prompts for the password. If you enters the correct password, the EXEC prints its normal prompt. You may try three times to enter a password before the EXEC exits and returns the terminal to the idle state.

To specify a password, use the **password** line subcommand. Its full syntax follows.

password *text*
no password

The *text* argument may contain any alphanumeric characters, including spaces, up to 80 characters. The password checking is case sensitive. The password *Secret* is different than the password *secret*, for example, and the password *two words* is an acceptable password.

Example:

This command sets the words "Big Easy" as the password on line 1:

```
line 1
password Big Easy
```

Establishing Connection Restrictions

To establish connection restrictions on the lines to some Internet addresses, use the **access-class** line subcommand. The full command syntax follows.

access-class *list* {**in** | **out**}
no access-class *list* {**in** | **out**}

The **access-class** subcommand restricts connections on a line or group of lines to certain Internet addresses. The argument *list* is an integer from 1 through 99 that identifies a specific access list of Internet addresses. The keyword **in** applies to incoming connections, such as virtual terminals. The keyword **out** applies to outgoing Telnet connections. The **no access-class** command removes access restrictions on the line for the specified connections.

Example:

This example subcommand sets restrictions on access list 1 for outgoing Telnet connections:

```
access-class 1 out
```

See the section "Configuring IP Access Lists" in the "Routing IP" chapter for information about configuring access lists.

Suppressing Banner Messages

By default, messages defined by the **banner motd** and **banner exec** commands are always displayed. This condition is defined by the **exec-banner** line subcommand. Its full syntax follows:

exec-banner
no exec-banner

To suppress display of a banner, enter the **no exec-banner** command.

Example:

These commands suppresses the banner on virtual terminal lines 0 through 4:

```
line vty 0 4
no exec-banner
```

Turning On/Off the Vacant Banner

The router will display a message on the console when there is no active EXEC. This message, called the vacant message, is different from the banner message displayed when an EXEC process is activated.

To turn the vacant message banner on or off, use the **vacant-message** line configuration subcommands. The **vacant-message** command enables the banner to be displayed on the screen of an idle terminal. The full syntax of this command follows.

vacant-message
vacant-message *c message c*
no vacant-message

The **vacant-message** subcommand without any arguments causes the default message to be displayed. If you desire a banner, follow **vacant-message** with one or more blank spaces and a delimiting character (*c*) you choose. Then type one or more lines of text (*message*), terminating the text with the second occurrence of the delimiting character.

The **no vacant-message** line configuration subcommand suppresses a banner message.

Example:
This example will turn on the system banner and display a message.

```
line 0
vacant-message #
                Welcome to Cisco Systems, Inc.
        This is the console terminal of the router Dross.
#
```

Note: You cannot use the delimiting character in the banner message.

Setting the Escape Character

The **escape-character** line subcommand defines the escape character. The following illustrates the full syntax of this command:

escape-character *decimal-number*
no escape-character

The argument *decimal-number* is the ASCII decimal representation of the desired escape character or an escape character (Ctrl-P, for example). Typing the escape character followed by the X key returns you to the EXEC when you are connected to another computer. The default escape character is Ctrl-^. (See the appendix “ASCII Character Set” for a list of ASCII characters.)

The operating software interprets Break on the console as an attempt to halt the system.

Note: Depending upon the configuration register setting, console breaks will either be ignored or cause the server to shut down. The Break key *cannot* be used as the escape character on the Cisco router.

The **no escape-character** line configuration subcommand reinstates the default escape character.

Example:

This command changes the escape characters to Ctrl-P (ASCII character 17):

```
line 5
escape-character 17
```

Setting the Terminal Location

To set the location of the terminal, use the **location** line subcommand. The full syntax of this command follows:

location *text*
no location

This subcommand is for informational purposes only; it is not used by any aspects of the system software. The argument *text* is the desired description. The description appears in the output of the EXEC command **systat.** A maximum of 80 characters can be entered.

The **no location** subcommand removes the information.

Example:

This command describes the location of the terminal on line 2 as being Andrea's terminal:

```
line 2
location Andrea's terminal
```

Setting the EXEC Timeout Intervals

The EXEC command interpreter waits for a specified interval of time until the user starts input. If no input is detected, the EXEC resumes the current connection. If no connections exist, the EXEC returns the terminal to the idle state and disconnects the incoming session.

To set this interval, use the **exec-timeout** line configuration subcommand. The full syntax of the command follows.

exec-timeout *minutes* [*seconds*]
no exec-timeout

The argument *minutes* is the number of minutes, and the optional argument *seconds* specifies additional interval time in seconds. The default interval is ten minutes; an interval of zero specifies no time-outs.

The **no exec-timeout** subcommand removes the time-out definition. It is the same as entering **exec timeout 0**.

Example:

This command sets an interval of 2 minutes, 30 seconds.

```
exec-timeout 2 30
```

This command sets an interval of 10 seconds:

```
exec-timeout 0 10
```

Setting the Screen Length

To set the terminal screen length, use the **length** line configuration subcommand. The full syntax of the command follows.

length *screen-length*
no length

The argument *screen-length* is the number of lines on the screen. The network server uses this value to determine when to pause during multiple-screen output. The default length is 24 lines. A value of zero disables pausing between screens of output.

The **no length** command is the same as entering the command **length 0**.

Note: Not all commands pay attention to the configured screen length. For example, the **show terminal** command assumes a screen length of 24 lines or more.

Setting Notification

To enable the terminal to notify the user about pending output, use the **notify** line subcommand. The full syntax of the command follows.

notify
no notify

The **notify** subcommand sets a line to inform a user who has multiple, concurrent Telnet connections when output is pending on a connection other than the current connection.

The **no notify** line configuration subcommand ends notification and is the default.

Setting Character Padding

To set the padding on characters, use the **padding** line configuration subcommand. The full syntax of the command follows.

padding *decimal-number count*

no padding *decimal-number*

The **padding** subcommand sets padding for a specified output character. The argument *decimal-number* is the ASCII decimal representation of the character, and the argument *count* is the number of NUL bytes sent after that character.

The **no padding** line configuration subcommand removes padding for the specified output character.

Example:

The following command pads Return (ASCII character 13) with 25 NUL bytes:

```
padding 13 25
```

Global System Configuration Command Summary

This section lists all of the global system configuration commands in alphabetical order.

banner {**motd**|**exec**|**incoming**} *c text c*

Displays the message that the EXEC command interpreter displays whenever a user starts any EXEC process or activates a line. The **motd**, **exec**, and **incoming** keywords control when the banner message is displayed. The argument *c* specifies a delimiting character of your choice. The argument *text* specifies the message to be shown on the screen whenever an interface line is activated.

[**no**] **boot buffersize** *bytes*

Specifies the size of the buffer to be used for netbooting a host or a network configuration file. The argument *bytes* by default is the size of your nonvolatile memory, or 32 kilobytes if you do not have nonvolatile memory. There is no minimum or maximum size that may be specified.

[**no**] **boot host** *filename* [*address*]

Specifies the host configuration file name. The argument *filename* is the new name for the host configuration file. If you omit the argument *address*, the network server uses the default broadcast address of *255.255.255.255*. The optional argument *address* allows you to specify a specific network host or a subnet broadcast address.

[**no**] **boot network** *filename* [*address*]

Specifies the network configuration file. The argument *filename* is the new name for the network configuration file. If you omit the optional argument *address*, the network server uses the default broadcast address of *255.255.255.255*. The optional argument *address* allows you to specify a specific network host or a subnet broadcast address.

[**no**] **boot system** *filename* [*address*]

Specifies a second operating software image from a file that is not in nonvolatile memory. The argument *filename* is the file name of the operating software to load, and the optional argument *address* is the address of the network host holding that file.

[**no**] **buffers** {**small**|**middle**|**big**|**large**|**huge**} {**permanent**|**max-free**|**min-free**|**initial**} *number*

Allows a network administrator to adjust initial buffer pool settings and set limits at which temporary buffers are created and destroyed.The first keyword denotes the size of buffers in the pool; the default number of the buffers in a pool is determined by the hardware configuration. The second keyword specifies the buffer management parameter to be changed, as follows:

- **permanent**—The number of permanent buffers that the system tries to allocate.
- **max-free**—The maximum number of free or unallocated buffers in a buffer pool.
- **min-free**—The minimum number of free or unallocated buffers in a buffer pool.
- **initial**—The number of additional temporary buffers which should be allocated when the system is reloaded.

The argument *number* specifies the number of buffers to be allocated.

The **no buffers** command with appropriate keywords and arguments restores the default buffer values.

enable password *password*

Assigns a password for the privileged command level. The argument password is case sensitive and specifies the password prompted for in response to the EXEC command **enable**.

[**no**] **enable last-resort** {**succeed**|**password**}

Allows you to specify what happens if the TACACS servers used by the **enable** command do not respond. The default action is to fail. The keywords change this default action:

- **succeed**—Allows you to enable without further question.
- **password**—Allows you to enable by entering the privileged command level.

[**no**] **enable use-tacacs**

Enables or disables use of TACACS to check the user ID and password supplied to the

EXEC **enable** command.

hostname *name*

Specifies the name for the network server. The default is *Gateway.*

[**no**] **logging** *internet-address*

Identifies a syslog server host to receive logging messages. The argument *internet-address* is the Internet address of the host.

[**no**] **logging buffered**

Copies logging messages to an internal buffer instead of writing them to the console.

[**no**] **logging console** *level*

Limits the logging of messages displayed on the console terminal to messages with a level at or above the specified severity, which is specified by the *level* argument. The argument *level* can be one of the following keywords, listed here in order from the most severe to the least severe level.

- **emergencies**—System unusable
- **alerts**—Immediate action needed
- **critical**—Critical conditions
- **errors**—Error conditions
- **warnings**—Warning conditions (default)
- **notifications**—Normal but significant conditions
- **informational**—Informational messages only
- **debug**—Debugging messages

[**no**] **logging monitor** *level*

Limits the logging messages displayed on terminal lines other than the console line to messages with a level at or above *level.* The argument *level* is one of the keywords described for the **logging console** command.

[**no**] **logging on**

Enables or disables message logging to all supported destinations except the console. This behavior is the default.

[**no**] **logging trap** *level*

Limits the logging messages sent to syslog servers to messages with a level at or above *level*. The argument *level* is one of the keywords described for the **logging console** command.

[**no**] **service** *keyword*

Tailors use by the network server of network-based services. The argument *keyword* is one of the following:

- **config**—Specifies TFTP autoloading of configuration files; disabled by default on system with nonvolatile memory.
- **decimal-tty**—Specifies that line numbers be displayed and interpreted as decimal numbers rather than octal numbers; disabled by default.
- **finger**—Allows Finger protocol requests (defined in RFC 742) to be made of the network server; enabled by default.
- **tcp-keepalives-**{**in**|**out**}—Generates keepalive packets on idle network connections. The **in** keyword generates them on incoming connections; the **out** keyword generates them on outgoing connections.

no snmp-server

Disables the SNMP operations.

[**no**] **snmp-server access-list** *list*

Sets up an access list that determines which hosts can send requests to the network server. The argument *list* is an integer from 1 through 99 that specifies an IP access list.

[**no**] **snmp-server community** *string* [**RO**|**RW**][*list*]

Enables or disables SNMP server operation on the network server. The argument *string* specifies a community string that acts like a password and permits access to the SNMP protocol.

[**no**] **snmp-server host** *address community-string* [s**nmp**|**tty**|**x25**]

Specifies which host or hosts should receive TRAP messages. Issue the **snmp-server host** command once for each host acting as a TRAP recipient. The argument *address* is the name or Internet address of the host. The argument *community-string* is the password-like community string set with the **snmp-server community** command. These optional keywords direct the TRAP:

- **snmp**—Causes all SNMP-type TRAP messages to be sent and starts the Cisco-specific RELOAD TRAP message.

- **tty**—Causes TCP connection TRAP messages to be included.

[**no**] **snmp-server packetsize** *bytes*

Sets or removes control over the largest SNMP packet size permitted when the SNMP server is receiving a request or generating a reply. The argument *bytes* is a byte count from 484 through 8,192. The default is 484.

[**no**] **snmp-server queue-length** *length*

Defines the length of the message queue for each TRAP host. The argument *length* is the number of TRAP events that can be held before the queue must be emptied; the default is ten.

[**no**] **snmp-server system-shutdown**

Allows or restricts use of the SNMP message reload feature, and prevents an SNMP system-shutdown request from resetting the Cisco agent.

[**no**] **snmp-server trap-authentication**

Allows or restricts the network server from sending a TRAP message when it receives a packet with an incorrect community string.

[**no**] **snmp-server trap-timeout** *seconds*

Defines how often to try resending TRAP messages on the retransmission queue. The argument *seconds* sets the interval for resending the messages. The default is set to 30 seconds. The **no** form of the command restores the default.

[**no**] **tacacs-server attempts** *count*

Controls the number of login attempts that may be made on a line set up for TACACS verification. The argument *count* is the number of attempts. The default is three attempts.

[**no**] **tacacs-server authenticate** {**connect**|**slip**|**enable**}

Specifies that a response is required from the network or communications server to indicate whether the user may perform the indicated action. Actions which require a response include the following:

- **connect**—Makes TCP connections
- **slip**—SLIP connections (Terminal Server only)
- **enable**—Use of **enable** command

The **no** form of the command disables the action.

[**no**] **tacacs-server extended**

Enables or disables an extended TACACS mode. This mode provides information about the terminal requests for use in setting up UNIX auditing trails and accounting files for tracking use of terminal servers and routers.

[**no**] **tacacs-server host** *name*

Specifies a TACACS host. The argument *name* is the name or Internet address of the host. You can use multiple commands to specify multiple hosts.

[**no**] **tacacs-server last-resort** {**password**|**succeed**}

Causes the network server to request the privileged password as verification, or forces successful login without further input from the user, depending upon the keyword specified, as follows.

- **password**—Allows the user to access the privileged-level command mode by entering the password set by the **enable** command.
- **succeed**—Allows the user to access the privileged-level command mode without further question.

The **no** form of the command disables the action.

tacacs-server notify {**connect**|**slip**|**enable**|**logout**}

Causes a message to be transmitted to the TACACS server, with retransmission being performed by a background process for up to five minutes. The keywords specify notification of the TACACS server whenever a user does one of the following:

- **connect**—Makes TCP connections.
- **slip**—Turns SLIP on or off (Terminal Server only).
- **enable**–Enters the **enable** command.
- **logout**—Logs out.

The **no** form of the command disables the messages.

[**no**] **tacacs-server retransmit** *retries*

Specifies the number of times the server will search the list of TACACS server hosts before giving up. The argument *retries* is the retransmit count. The default is two retries. The **no** form of the command restores the default.

[**no**] **tacacs-server timeout** *seconds*

Sets the interval the server waits for a server host to reply. The argument *seconds* specifies the number of seconds. The default interval is five seconds. The **no** form of the command restores the default.

[**no**] **tftp-server system** *filename ip-access-list*

Specifies or removes TFTP server operation for a communications server. The argument *filename* is the name given to the network server ROM file, and the argument *access-list* is an IP access list number.

Line Configuration Subcommand Summary

This section contains a summary of all the line configuration subcommands in alphabetical order.

[**no**] **access-class** *list* {**in**|**out**}

Restricts or permits connections on a line or group of lines to certain Internet addresses. The argument *list* is an integer from 1 through 99 that identifies a specific access list of Internet addresses. The keyword **in** applies to incoming connections; the keyword **out** applies to outgoing Telnet connections.

[**no**] **escape-character** *decimal-number*

Sets or removes the escape character on the specified line. The argument *decimal-number* is either the ASCII decimal representation of the character or a control sequence (Ctrl-E, for example). The default escape character is Ctrl-^.

[**no**] **exec-banner**

Enables or disables a banner. By default, a banner is displayed on the console. To suppress display of a banner, enter the **no** variation of the command.

[**no**] **exec-timeout** *minutes* [*seconds*]

Sets the interval the EXEC waits for user input before resuming the current connection, or if no connections exist, before returning the terminal to the idle state and disconnecting the incoming session. The argument *minutes* is the number of minutes, and the optional argument *seconds* specifies additional interval time in seconds. The default interval is ten minutes; an interval of zero specifies no time-outs. The **no** form of the command restores the default.

[**no**] **length** *screen-length*

Sets the terminal screen length. The argument *screen-length* is the number of lines on the screen. The network server uses this value to determine when to pause during multiple-screen output. The default length is 24 lines. A value of 0 (zero) or the **no** form of the command disables pausing between screens of output.

line [*type-keyword*] *first-line* [*last-line*]

Identifies a specific line for configuration and starts line configuration command collection. The optional argument *type-keyword* specifies the type of line to be configured, it is **console**, **aux**, or **vty**. When the line type is specified, the argument *first-line* is the relative number of the terminal line (or the first line in a contiguous group) you want to configure. The optional argument *last-line* then is the relative number of the last line in a contiguous group you want to configure. If you omit *type*, then *first-line* and *last-line* are absolute rather than relative line numbers.

[**no**] **location** *text*

Enters or removes informational only information about the terminal location and/or status. The argument *text* is the desired description.

[**no**] **login**

Enables or disables password checking for the password specified by the **password** command.

[**no**] **login tacacs**

Causes the TACACS user ID and password checking mechanism to be used instead of the regular password checking. The **no** form of the command disables this mechanism.

[**no**] **notify**

Enables or disables line notification of when a user who has multiple, concurrent Telnet connections has output pending on a connection other than the current line.

[**no**] **padding** *decimal-number count*

Sets or unsets padding for a specified output character. The argument *decimal-number* is the ASCII decimal representation of the character, and the argument *count* is the number of NUL bytes sent after that character.

[**no**] **password** *text*

Specifies a password. The *text* argument specifies a password. It may contain any alphanumeric characters, including spaces, up to 80 characters.

vacant-message *c message c*
[no] vacant-message

Controls whether or not a banner is displayed on the screen of an idle terminal. The command without any arguments causes the default message to be displayed. The **no vacant-message** command suppresses a banner message. To display a banner, follow **vacant-message** with one or more blank spaces and a delimiting character (*c*) you choose, then type one or more lines of text (*message*), and terminate the text with the second occurrence of the delimiting character.