



Configuring CSS Remote Access Methods

This chapter describes how to configure the Secure Shell Daemon (SSH), Remote Authentication Dial-In User Service (RADIUS), and the Terminal Access Controller Access Control System (TACACS+) remote access method. Information in this chapter applies to all models of the CSS, except where noted.

This chapter contains the following major sections:

- [Configuring the Secure Shell Daemon Protocol](#)
- [Configuring the CSS as a Client of a RADIUS Server](#)
- [Configuring the CSS as a Client of a TACACS+ Server](#)
- [Controlling Remote Access to the CSS](#)
- [Controlling Access to the CSS](#)

Configuring the Secure Shell Daemon Protocol

The Secure Shell Daemon (SSHD) protocol provides secure encrypted communications between two hosts communicating over an insecure network. The CSS supports an implementation of OpenSSH to provide this secure communication. SSHD uses the standard CSS login sequence of entering the username and password at the CSS login prompts.

SSHD on the CSS supports both the SSH v1 and v2 protocols. For SSH v1, the software provides encrypted communication using ciphers such as 3DES or Blowfish. For SSH v2, the software provides 128-bit AES, Blowfish, 3DES, CAST128, Arcfour, 192-bit AES, or 256-bit AES.



Caution

When using SSHD, ensure that the CSS is not configured to perform a network boot from a network-mounted file system on a remote system (a diskless environment). If you require the CSS to use the network-mounted method of booting, be aware that the SSHD protocol is not supported.

If the CSS has been booted using a network boot from a network-mounted file system, the CSS logs the following error message by SSHD as the protocol attempts to initialize (and then exit from operation):

```
Unable to initialize sshd; failure to seed random number generator
```

This section includes the following topics:

- [Enabling SSH](#)
- [Configuring SSH Access](#)
- [Configuring SSHD in the CSS](#)
- [Configuring Telnet Access When Using SSHD](#)
- [Showing SSHD Configurations](#)

Enabling SSH

To enable SSH functionality in your CSS, you must purchase the Secure Management software option. If you purchased the Secure Management software option:

- During the initial CSS order placement, the software Claim Certificate is included in the accessory kit.
- After receiving the CSS, Cisco Systems sends the Claim Certificate to you by mail.

**Note**

If you cannot locate the Secure Management option Claim Certificate in the accessory kit, call the Cisco Technical Assistance Center (TAC) toll free, 24 hours a day, 7 days a week at 1-800-553-2447 or 1-408-526-7209. You can also e-mail TAC at tac@cisco.com.

Follow the instructions on the Claim Certificate to obtain the Secure Management software license key.

To enter the Secure Management license key and activate SSH:

1. Log in to the CSS and enter the **license** command.

```
# license
```

2. Enter the Secure Management license key.

```
Enter the Software License Key (q to quit): nnnnnnnnnnnn
```

The Secure Management license key is now properly installed and the SSH function activated.

Configuring SSH Access

SSH access to the CSS is enabled by default through the **no restrict ssh** command. You can verify the SSH access selection in the running-config file.

To enhance security when using SSHD, disable Telnet access (Telnet access is enabled by default). Use the **telnet-access disable** command as described in the “Controlling Access to the CSS” section.

To enable SSH access to the CSS, enter:

```
(config)# no restrict ssh
```

To disable SSH access, enter:

```
(config)# restrict ssh
```

Configuring SSHD in the CSS

The CSS provides the following commands for configuring SSHD:

- **sshd keepalive** - Enables TCP keepalive messages
- **sshd port** - Specifies the SSHD port
- **sshd server-keybits** - Sets the number of bits in the ephemeral protocol server key (SSH v1 only)

Ensure you enable SSHD access to the CSS for SSHD to accept connections from SSH clients. By default, SSH access is enabled through the **no restrict ssh** global command.

Configuring SSHD Keepalive

The CSS supports sending TCP keepalive messages to the client as a means for the server to determine whether the SSHD connection to the client is functioning (for example, if the network has gone down or the client has become unresponsive). If you disable sending SSHD keepalives to a client, sessions may hang indefinitely on the server, which consumes system resources.

Use the **sshd keepalive** command to enable SSHD keepalive. SSHD keepalive is enabled by default.

To enable sending SSHD keepalives to the client, enter:

```
(config)# sshd keepalive
```

To disable sending SSHD keepalives, enter:

```
(config)# no sshd keepalive
```

Configuring SSHD Port

Use the **sshd port** command to specify the port number to which the server listens for connections from clients. Enter a port number of 22 or from 512 to 65535. The default is 22 (the default port for SSH).

For example, to configure port number 65530 as the SSHD port, enter:

```
(config)# sshd port 65530
```

To reset the port number to the default of 22, enter:

```
(config)# no sshd port
```

Configuring SSHD Server-Keybits

Use the **sshd server keybits** command to specify the number of bits in the ephemeral protocol server key. The **sshd server keybits** command pertains only to SSH v1 connections. Enter the number of bits from 512 to 1024. The default is 768.

For example, to set the number of bits in the server key to 1024, enter:

```
(config)# sshd server-keybits 1024
```

To reset the number of bits to the default of 768, enter:

```
(config)# no sshd server-keybits
```

Configuring Telnet Access When Using SSHD

By default, Telnet access to the CSS is enabled. When you use SSHD, you can disable nonsecure Telnet access to the CSS. To enhance security when using SSHD, we recommend that you disable Telnet access. Use the global restrict telnet command to disable Telnet access to the CSS.

To disable Telnet access, enter:

```
(config)# restrict telnet
```

To reenable Telnet access to the CSS, enter:

```
(config)# no restrict telnet
```

Showing SSHD Configurations

Use the **show sshd** command to display SSHD configurations. This command provides the following options:

- **show sshd config** - Displays the SSHD configuration
- **show sshd sessions** - Displays a summary of the current active SSHD server sessions. The command displays data only if an SSH client is currently configured.
- **show sshd version** - Show the current version of the SSHield package running in the CSS.

To display the SSHD configuration, enter:

```
# show sshd config
```

[Table 11-1](#) describes the fields in the **show sshd config** command output.

Table 11-1 Field Descriptions for the show sshd config Command

Field	Description
Maximum Sessions Allowed	The maximum number of concurrent SSHD sessions (five maximum).
Active Sessions	The number of currently active SSHD sessions.
Log Level	The current log level.
Listen Socket Count	The number of sockets that SSHD is currently listening on (not currently configurable, default is 1).
Listen Port	The port number that SSHD uses to listen for client connections (set by the sshd port command). The default is 22 (the default port for SSH). The port number is 22 or from 512 to 65535.

Table 11-1 *Field Descriptions for the show sshd config Command (continued)*

Field	Description
Listen Address	The address that SSHD uses to listen for client connections (not currently configurable; default is 0.0.0.0).
Server Key Bits	The number of bits to use when generating the SSHv1 server key. The default is 768. The range is from 512 to 1024.
RSA Protocol (SSH1)	The status of SSHv1 access (not currently configurable; default is enabled).
Empty Passwords	Disabled. The username must always have an associated password.
Keepalive	The status of sending a TCP keepalive to the client: Enabled or Disabled. SSHD keepalive is enabled by default.
SSH2 Cipher List	A list of SSHv2 cipher suites supported for authentication, encryption, and data integrity between the client and the server.

To display the SSHD sessions, enter:

```
# show sshd sessions
```

[Table 11-2](#) describes the fields in the **show sshd sessions** command output.

Table 11-2 *Field Descriptions for the show sshd sessions Command*

Field	Description
Session_ID	The session ID.
Conn_TID	The connection task ID of the SSHD server handling the connection (tSshConn).
Login_TID	The login task ID handling the connection (tSshCli).

Table 11-2 *Field Descriptions for the show sshd sessions Command (continued)*

Field	Description
PTY_FD	<p>The file descriptor used by the login task to communicate with the CSS CLI.</p> <p>The PTY_FD file descriptor allows you to correlate the SSH client sessions with those sessions listed under the Line field in the show lines output. For example, the show sshd sessions output displays an SSH client session connected to PTY_FD32. If you enter the show lines command you see a line in the display listing sshc32 (for SSH client pty_fd32). This correlation allows you to view the login time, idle time, and the location of the client of the SSH sessions through the show lines command.</p>
Remote IP/ Remote Port	The remote IP and port number of the SSHD session.

To display the SSHD version, enter:

```
# show sshd version
SSHield version 1.5, SSH version OpenSSH_3.0.2p1
```

Configuring the CSS as a Client of a RADIUS Server

The Remote Authentication Dial-In User Service (RADIUS) protocol is a distributed client/server protocol that protects networks against unauthorized access. RADIUS uses the User Datagram Protocol (UDP) to exchange authentication and configuration information between the CSS authentication client and the active authentication server that contains all user authentication and network service access information. The RADIUS host is normally a multiuser system running RADIUS server software.

When a user remotely logs in to a CSS operating as a RADIUS client, the CSS sends an authentication request (including username, encrypted password, client IP address, and port ID) to the central RADIUS server. The RADIUS server is responsible for receiving user connection requests, authenticating users, and returning all configuration information necessary for the client to deliver services to the users. Transactions between the RADIUS client and the RADIUS server are authenticated through the use of a shared secret.

Once the RADIUS server receives the authentication request, it validates the sending client and consults a database of users to match the login request. If no response is returned by the RADIUS server within a period of time, the authentication request is retransmitted a predefined number of times. The RADIUS client can forward requests to an alternate secondary RADIUS server in the event that the primary server is down or is unreachable.

In a configuration where both a primary RADIUS server and a secondary RADIUS server are specified, and one or both of the RADIUS servers become unreachable, the CSS automatically transmits a keepalive authentication request to query the server(s). The CSS transmits the username “query” and the password “areyouup” to the RADIUS server (encrypted with the RADIUS server’s key) to determine the server’s state. The CSS continues to send this keepalive authentication request until the RADIUS server indicates it is available.

Use the **radius-server** command and its options to specify the RADIUS server host (primary RADIUS server, and, optionally, a secondary RADIUS server), communication time interval settings, and a shared secret text string. This command is available in global configuration mode.

This section includes the following topics:

- [Configuring a RADIUS Server for Use with the CSS](#)
- [Specifying a Primary RADIUS Server](#)
- [Specifying a Secondary RADIUS Server](#)
- [Configuring the RADIUS Server Retransmits](#)
- [Configuring the RADIUS Server Dead-Time](#)
- [Showing RADIUS Server Configuration Information](#)

After configuring the RADIUS server, enable RADIUS authentication for console and virtual logins (if the username and password pair is not in the local user database) through the **virtual authentication** and **console authentication** commands. See the “[Controlling Remote Access to the CSS](#)” section for details on the two commands.

Configuring a RADIUS Server for Use with the CSS

This section provides background information on the setup of a RADIUS server. It is intended as a guide to help ensure proper communication with a RADIUS server and a CSS operating as a RADIUS client.

The following sections summarize the recommended settings for the Cisco Secure Access Control Server (ACS) when used as a centralized RADIUS server with the CSS.

Configuring Authentication Settings

To configure the authentication settings on Cisco Secure ACS, go to the Network Configuration section of the Cisco Secure ACS HTML interface, the Add AAA Client page, and complete the following fields:

- AAA Client Hostname - Enter a name you want assigned to the CSS.
- AAA Client IP Address - Enter the IP address of the CSS Ethernet Management port or of a CSS circuit (depending on how the CSS is configured to communicate with the Cisco Secure ACS).

- **Key** - Enter the shared secret that the CSS and Cisco Secure ACS use to authenticate transactions. For correct operation, you must specify the identical shared secret on both the Cisco Secure ACS and the CSS. The key is case-sensitive.
- **Authenticate Using** - Select the **RADIUS (IETF)** network security protocol to use the standard IETF RADIUS attributes with the CSS.

Configuring Authorization Settings

To determine the privilege level of users accessing the CSS, you must configure the user accounts on the RADIUS server.

To configure the group authorization settings:

1. From the Group Setup section of the Cisco Secure ACS HTML interface, Group Setup Select page, select the group for which you want to configure RADIUS settings.
2. From the Group Settings section of the Cisco Secure ACS HTML interface, click the **IETF RADIUS Attributes, [006] Service-Type** checkbox. Then select **Administrative**. Administrative is required to enable RADIUS authentication for privileged user (SuperUser) connection with the CSS.

To add a user to a group, go to the **User Setup** section of the Cisco Secure ACS HTML interface:

- On the User Setup Select page, specify a username.
- On the User Setup Edit page, specify the following:
 - Password Authentication - Select an applicable authentication type from the list.
 - Password - Specify and confirm a password.
 - Group - Select the previously created RADIUS group to which you want to assign the user.

Specifying a Primary RADIUS Server

Use the **radius-server primary** command to specify a primary RADIUS server used to authenticate user information from the CSS RADIUS client (console or virtual authentication). The syntax for this global configuration mode command is:

```
radius-server primary ip_address secret string {auth-port port_number}
```

Options and variables for this command are as follows:

- **primary** *ip_address* - The IP address or host name for the primary RADIUS server. Enter the address in either dotted-decimal IP notation (for example, 192.168.11.1) or mnemonic host-name format (for example, myhost.mydomain.com).
- **secret** *string* - The shared secret text string between the primary RADIUS server and the CSS RADIUS client. The shared secret allows authentication transactions between the client and primary RADIUS server to occur. Enter the shared secret as a case-sensitive string with no spaces (16 characters maximum).
- **auth-port** *port_number* - (Optional) The UDP port on the primary RADIUS server allocated to receive authentication packets from the RADIUS client. Valid entries are 0 to 65535. The default is 1645.

To specify a primary RADIUS server, enter:

```
(config)# radius-server primary 172.27.56.76 secret Hello  
auth-port 30658
```

To remove a primary RADIUS server, enter:

```
(config)# no radius-server primary
```

Specifying a Secondary RADIUS Server

Use the **radius-server secondary** command to specify a secondary RADIUS server to authenticate user information from the CSS RADIUS client (console or virtual authentication). The CSS directs authentication requests to the secondary RADIUS server when the specified RADIUS primary server is unavailable.

**Note**

Configuration of a secondary RADIUS server is optional.

The syntax for this global configuration mode command is:

```
radius-server secondary ip_address secret string { auth-port port_number }
```

Options and variables for this command are as follows:

- **secondary** *ip_address* - The IP address or host name for the secondary RADIUS server. Enter the address in either dotted-decimal IP notation (for example, 192.168.11.1) or mnemonic host-name format (for example, myhost.mydomain.com).
- **secret** *string* - The shared secret text string between the secondary RADIUS server and the CSS RADIUS client. The shared secret allows authentication transactions between the client and secondary RADIUS server to occur. Enter the shared secret as a case-sensitive string with no spaces (16 characters maximum).
- **auth-port** *port_number* - (Optional) The UDP port on the primary RADIUS server allocated to receive authentication packets from the RADIUS client. Valid entries are 0 to 65535. The default is 1645.

To specify a secondary RADIUS server, enter:

```
(config)# radius-server secondary 172.27.56.79 secret Hello  
auth-port 30658
```

To remove a secondary RADIUS server, enter:

```
(config)# no radius-server secondary
```

Configuring the RADIUS Server Timeouts

Use the **radius-server timeout** command to specify the time interval that the CSS waits for the RADIUS server (primary or secondary) to reply to an authentication request before retransmitting requests to the RADIUS server. You configure the number of retransmitted requests to the server through the **radius-server retransmit** command (see the [“Configuring the RADIUS Server Retransmits”](#) section). Valid entries are 1 to 255 seconds. The default is 10 seconds.

For example, to configure the RADIUS server timeout interval to 1 minute (60 seconds), enter:

```
(config)# radius-server timeout 60
```

To reset the RADIUS server retransmit request to the default of 10 seconds, enter:

```
(config)# no radius-server timeout
```

Configuring the RADIUS Server Retransmits

Use the **radius-server retransmit** command to specify the number of times the CSS retransmits an authentication request to a timed-out RADIUS server before considering the server dead and stopping transmission. If a secondary RADIUS server has been identified, the server is selected as the active server. Valid entries are 1 to 30 retries. The default is 3 retransmissions.

If the RADIUS server does not respond to the CSS retransmitted requests, the CSS considers the server as dead, stops transmitting to the server, and starts the dead timer as defined through the **radius-server dead-time** command (see the [“Configuring the RADIUS Server Dead-Time”](#) section). If a secondary server is configured, the CSS transmits the requests to the secondary server. If the secondary server does not respond to the request, the CSS considers the server dead and starts the dead timer. If there is no active server, the CSS stops transmitting requests until the primary RADIUS server becomes alive.

For example, to configure the number of RADIUS server retransmissions to 5, enter:

```
(config)# radius-server retransmit 5
```

To reset the RADIUS server retransmit request to the default of 3 retransmissions, enter:

```
(config)# no radius-server retransmit
```

Configuring the RADIUS Server Dead-Time

Use the **radius-server dead-time** command to set the time interval in which the CSS verifies whether a nonresponsive server is operational. During the set time interval, the CSS sends probe access-request packets to verify that the RADIUS server (primary or secondary) is available and can receive authentication requests. The dead-time interval starts when the server does not respond to the number of authentication request transmissions configured through the **radius-server retransmit** command. When the server responds to a probe access-request packet, the CSS transmits the authentication request to the server. Valid entries are 1 to 255 seconds. The default is 5 seconds.

To configure the RADIUS server dead-time to 15 seconds, with probe access-requests enabled, enter:

```
(config)# radius-server dead-time 15
```

To reset the RADIUS server dead-time request to the default of 5 seconds, enter:

```
(config)# no radius-server dead-time
```

Showing RADIUS Server Configuration Information

Use the **show radius** command to display information and statistics about the RADIUS server configuration. The syntax and options for the command are as follows:

- **show radius config [primary|secondary|all]** - Displays RADIUS configuration information for a specific server or all servers, identified by type
- **show radius stat [primary|secondary|all]** - Displays RADIUS authentication statistics for a specific server or all servers, identified by type

To view the configuration for a RADIUS primary server, enter:

```
(config)# show radius config primary
```

To view the authentication statistics for a RADIUS secondary server, enter:

```
(config)# show radius stats secondary
```

[Table 11-3](#) describes the fields in the **show radius config** command output.

Table 11-3 Field Descriptions for the show radius config Command

Field	Description
Server IP Address	The IP address or host name for the specified RADIUS server
Secret	The shared secret text string between the specified RADIUS server and the CSS RADIUS client
Port	The UDP port on the specified RADIUS server allocated to receive authentication packets from the CSS RADIUS client; the default port number is 1645
State	The operational stats of the RADIUS server (ALIVE, DOWN, UNKNOWN)
Dead Timer	The time interval (in seconds) that the CSS probes a nonresponsive RADIUS server (primary or secondary) to determine whether it is operational and can receive authentication requests
Timeout	The interval (in seconds) that the CSS RADIUS client waits for the RADIUS server to reply to an authentication request before retransmitting requests to the RADIUS server
Retransmit Limit	The number of times the CSS RADIUS client retransmits an authentication request to a timed out RADIUS server before stopping transmission to that server
Probes	The packets that the CSS RADIUS client automatically transmits as a means to determine whether the RADIUS server is still available and can receive authentication requests

Table 11-4 describes the fields in the **show radius stat** output.

Table 11-4 Field Descriptions for the show radius stat Command

Field	Description
Server IP address	The IP address or host name of the specified RADIUS server
Accepts	The number of times the RADIUS server accepts an authentication request from the CSS RADIUS client

Table 11-4 *Field Descriptions for the show radius stat Command (continued)*

Field	Description
Requests	The number of times the CSS RADIUS client issues an authentication request to the RADIUS server
Retransmits	The number of times the CSS RADIUS client retransmits an authentication request to the active RADIUS server after a timeout occurred
Rejects	The number of times the CSS RADIUS client receives a reject notification from the RADIUS server while trying to establish an authentication request
Bad Responses	The number of times the CSS RADIUS client receives a bad transmission from the RADIUS server
Bad Authenticators	The number of times the RADIUS server denies an authentication request from the CSS RADIUS client
Pending Requests	The number of pending authentication requests to the RADIUS server
Timeouts	The number of times the CSS RADIUS client reached the specified timeout interval while waiting for the RADIUS server to reply to an authentication request
Discarded Authentication Requests	The number of authentication requests that were discarded while the primary or secondary RADIUS server was down

Configuring the CSS as a Client of a TACACS+ Server

The Terminal Access Controller Access Control System (TACACS+) protocol provides access control for routers, network access servers (NAS), or other devices through one or more daemon servers. TACACS+ encrypts all traffic between the NAS and daemon using TCP communications for reliable delivery.

You can configure the CSS as a client of a TACACS+ server to provide a method for authentication of users, and a method of authorization and accounting of configuration and nonconfiguration commands.

This section includes the following topics:

- [Configuring TACACS+ Server User Accounts for Use with the CSS](#)
- [Defining a TACACS+ Server](#)
- [Defining a Global Encryption Key](#)
- [Setting the Global CSS TACACS+ Timeout Period](#)
- [Setting TACACS+ Authorization](#)
- [Setting TACACS+ Accounting](#)
- [Showing TACACS+ Server Configuration Information](#)

After you configure the TACACS+ server on the CSS, configure TACACS+ authentication for virtual or console authentication. See the “[Controlling Remote Access to the CSS](#)” section for details.

Configuring TACACS+ Server User Accounts for Use with the CSS

This section provides background information on the setup of a TACACS+ server. It is intended as a guide to help ensure proper communication with a TACACS+ server and a CSS operating as a TACACS+ client.

The following sections summarize the recommended Cisco Secure Access Control Server (ACS) TACACS+ user authentication and authorization settings.

Configuring Authentication Settings

To configure the authentication settings on Cisco Secure ACS, go to the Network Configuration section of the Cisco Secure ACS HTML interface, the Add AAA Client page, and complete the following fields:

- AAA Client Hostname - Enter a name you want assigned to the CSS.
- AAA Client IP Address - Enter the IP address of the CSS Ethernet management port or of a CSS circuit (depending on how the CSS is configured to communicate with the Cisco Secure ACS).
- Key - Enter the shared secret that the CSS and Cisco Secure ACS use to authenticate transactions. For correct operation, you must specify the identical shared secret on both the Cisco Secure ACS and the CSS. The key is case-sensitive.
- Authenticate Using - Select **TACACS+ (Cisco IOS)**.

Configuring Authorization Settings

To determine the privilege level of users accessing the CSS, you must configure the user accounts on the TACACS+ server to permit or deny execution of the **privilege** command. The CSS queries the TACACS+ server for authorization to execute the **privilege** command. If the server allows the **privilege** command, the user is granted privileged (SuperUser and configuration modes) access to the CSS. If the server denies the **privilege** command, the user is granted nonprivileged (User mode) access to the CSS.

To configure the group authorization settings:

1. From the Group Setup section of the Cisco Secure ACS HTML interface, Group Setup Select page, select the group for which you want to configure TACACS+ settings.
2. On the Shell Command Authorization Set page, click the **Per Group Command Authorization** checkbox

3. Under **Unmatched Cisco IOS Commands**, either permit or deny execution of the privilege command:
 - For a group that has SuperUser privileges on the CSS, select **Permit**. A SuperUser can issue any CSS command.
 - For a group that has User privileges on the CSS, select **Deny**. A user can issue CSS commands that does not change the CSS configuration; for example, **show** commands.

An alternative way to configure the group authorization settings is as follows:

1. Select **Shared Profile Components, Shell Command Authorization Sets** page.
2. Click the **Add** button to add a set or to edit an existing set.
3. Enter a name and description.
4. Proceed next to Unmatched Commands, either permit or deny execution of the privilege command:
 - For a user that has SuperUser privileges on the CSS, click **Permit**. A SuperUser can issue any CSS command.
 - For a user that has User privileges on the CSS, click **Deny**. A user can issue CSS commands that do not change the CSS configuration; for example, **show** commands.
5. From the Group Setup section, Group Setup Select page, select the group for which you want to configure TACACS+ settings.
6. On the Shell Command Authorization Set section, select **Assign a Shell Command Authorization Set for any network device**.
7. Select the set from the list.

To add a user to a group, go to the **User Setup** section of the Cisco Secure ACS HTML interface:

- On the User Setup Select page, specify a username.
- On the User Setup Edit page, specify the following:
 - Password Authentication - Select an applicable authentication type from the list.
 - Password - Specify and confirm a password.
 - Group - Select the previously created TACACS+ group to which you want to assign the user.

Defining a TACACS+ Server

The TACACS+ server contains the TACACS+ authentication, authorization, and accounting databases. You can designate a maximum of three servers on the CSS. However, the CSS uses only one server at a time. The CSS selects the server based upon availability, giving preference to the configured primary server. The CSS sends periodic TCP keepalive probes at a frequency of every five seconds to the TACACS+ server to determine its operational state: Alive, Dying, or Dead. The TCP keepalive frequency is not user-configurable in the CSS.



Note

For general guidelines on the recommended setup of a TACACS+ server (the Cisco Secure Access Control Server in this example), see the [“Configuring TACACS+ Server User Accounts for Use with the CSS”](#) section.

To apply a TACACS+ global attribute, such as the timeout period or shared secret, to a TACACS+ server, you must configure the global attribute before you configure the server. To apply a modified global attribute to a configured CSS TACACS+ server, remove the server and reconfigure it.

Use the **tacacs-server** command to define a server. You must provide the IP address and port number for the server. You can optionally define the timeout period and encryption key and designate the server as the primary server.

The syntax for this global configuration command is:

```
tacacs-server ip_address port {timeout [“cleartext_key”|des_key]}  
  {primary}
```

The variables and options for this command are as follows:

- *ip_address* - The IP address of the TACACS+ server. Enter the IP address in dotted-decimal format.
- *port* - The TCP port of TACACS+ server. The default port is 49. You can enter a port number from 1 to 65535.
- *timeout* - (Optional) The amount of time to wait for a response from the server. Enter a number from 1 to 255. The default is 5 seconds. Defining this option overrides the **tacacs-server timeout** command. For more information on the TACACS+ timeout period and setting a global timeout, see the [“Setting the Global CSS TACACS+ Timeout Period”](#) section.

- “*cleartext_key*”|*des_key* - (Optional) The shared secret between the CSS and the server. You must define an encryption key to encrypt TACACS+ packet transactions between the CSS and the TACACS+ server. If you do not define an encryption key, packets are not encrypted.

The shared secret value is identical to the one on the TACACS+ server. The shared secret key can be either clear text entered in quotes or the DES-encrypted secret entered without quotes. The clear text key is DES-encrypted before it is placed in the running configuration. Either key type can have a maximum of 100 characters.

Defining this option overrides the **tacacs-server key** command. For more information on defining a global encryption key, see the “[Defining a Global Encryption Key](#)” section.

- **primary** - (Optional) Assigns the TACACS+ server precedence over the other configured servers. You can specify only one primary server.



Note

If you need to change a timeout period or the shared secret for a specific server, you must delete the server and redefine it with the updated parameter.

For example, to define a primary TACACS+ server at IP address 192.168.11.1 with a default port of 49, a timeout period of 12 seconds, and a clear text shared secret of summary, enter:

```
#(config) tacacs-server 192.168.11.1 12 "summary" primary
```

To delete a TACACS+ server at IP address 192.168.11.1 with a default port of 49, enter:

```
#(config) no tacacs-server 192.168.11.1 49
```

After configuring the TACACS+ server, enable TACACS+ authentication for console and virtual logins (if the username and password pair is not in the local user database) through the **virtual authentication** and **console authentication** commands. See the “[Controlling Remote Access to the CSS](#)” section for information about the two commands.

Defining a Global Encryption Key

The CSS allows you to define a global encryption key for communications with all configured TACACS+ servers. To encrypt TACACS+ packet transactions between the CSS and the TACACS+ server, you must define an encryption key. If you do not define an encryption key, packets are not encrypted. The key is a shared secret value that is identical to the one on the TACACS+ server. Use the **tacacs-server key** command to specify a shared secret between the CSS and the server.

A shared secret defined when specifying a TACACS+ server overrides the global secret (see the “[Defining a TACACS+ Server](#)” section). To apply a modified global shared secret to a configured CSS TACACS+ server, remove the server and reconfigure it.

The shared secret key can be either clear text entered in quotes or the DES-encrypted secret. The clear text key is DES-encrypted before it is placed in the running configuration. Either key types can have a maximum of 100 characters.

For example, to define the clear text key, enter:

```
 #(config) tacacs-server key "market"
```

To define a DES-encrypted key, enter:

```
 #(config) tacacs-server key acskefterefesdtx
```

To remove the key, enter:

```
 #(config) no tacacs-server key
```

Setting the Global CSS TACACS+ Timeout Period

The CSS allows you to define a global TACACS+ timeout period for use with all configured TACACS+ servers. To determine the availability of the TACACS+ servers, the CSS sends periodic TCP keepalive probes to them. If the server does not respond to the probe within the timeout period, the CSS considers the server unavailable.

If the CSS attempts to contact the server and does not receive a response within the defined timeout value, it uses another server. The next configured server is contacted and the process repeated. If a second (or third) TACACS+ server has been identified, the CSS selects that server as the active server.

If the CSS cannot reach all three TACACS+ servers, users are not authenticated and cannot log in to the CSS unless TACACS+ is used in combination with a RADIUS or local server, as defined through the **virtual** command or the **console** command. See the “[Controlling Remote Access to the CSS](#)” section for details about the two commands.

By default, the global timeout period is 5 seconds. Use the **tacacs-server timeout** command to change the timeout period. Enter a number from 1 to 255.

The timeout period defined when specifying a TACACS+ server overrides the global timeout period (see the “[Defining a TACACS+ Server](#)” section). To apply a modified global timeout period to a configured CSS TACACS+ server, remove the server and reconfigure it.

For example, to set the timeout period to 60 seconds, enter:

```
 #(config) tacacs-server timeout 60
```

To reset the timeout period to the default of 5 seconds, enter:

```
 #(config) no tacacs-server timeout
```

Setting TACACS+ Authorization

TACACS+ authorization allows the TACACS+ server to control specific CSS commands that the user can execute. CSS authorization divides the command set into two categories:

- Configuration commands that change the CSS running configuration.
- Nonconfiguration commands that do not change the running configuration. These commands include, but are not limited to, mode transition, show, and administrative commands.

By default, authorization is disabled. When authorization is enabled, the TACACS+ server is responsible for granting permission or denying all attempts to issue commands.

When you enable authorization, the exchange between the TACACS+ server and the CSS causes a delay in executing the command. Failure of the TACACS+ server results in the failure of all authorization requests and the suspension of user activity unless another server is reachable. To enable users to execute commands in this case, configure a failover authentication method to a local user database. Users must log back in to the CSS.

Use the **tacacs-server authorize config** command to enable authorization of all commands that change the running configuration. For example:

```
#(config) tacacs-server authorize config
```

Use the **tacacs-server authorize non-config** command to enable authorization of all commands that do not change the running configuration. For example:

```
#(config) tacacs-server authorize non-config
```

Use the **no** form of these commands to disable authorization. For example, to disable authorization for commands that affect the running configuration, enter:

```
#(config) no tacacs-server authorize config
```

To disable authorization for commands that do not affect the running configuration, enter:

```
#(config) no tacacs-server authorize non-config
```

Setting TACACS+ Accounting

TACACS+ accounting allows the TACACS+ server to receive an accounting report for commands that the user can execute. CSS accounting divides the command set into two categories:

- Configuration commands that change the CSS running configuration.
- Nonconfiguration commands that do not change the running configuration. These commands include, but are not limited to, mode transition commands, show commands, and administrative commands.

By default, the CSS disables accounting. When you enable accounting, you can account for configuration commands, nonconfiguration commands, or both.

**Note**

Failure of the TACACS+ server does not result in the suspension of user activity.

Use the **tacacs-server account config** command to enable the TACACS+ server to receive accounting reports for all commands that change the running configuration. For example:

```
 #(config) tacacs-server account config
```

Use the **tacacs-server account non-config** command to enable the TACACS+ server to receive accounting reports for all commands that do not change the running configuration. For example:

```
 #(config) tacacs-server account non-config
```

Use the **no** form of these commands to disable accounting. For example, to disable accounting for commands that affect the running configuration, enter:

```
 #(config) no tacacs-server account config
```

To disable accounting for commands that do not affect the running configuration, enter:

```
 #(config) no tacacs-server account non-config
```

Showing TACACS+ Server Configuration Information

Use the **show tacacs-server** command to display the TACACS+ server configuration information. To view this information, enter:

```
(config)# show tacacs-server
```

Table 11-5 describes the fields in the **show tacacs-server** command output.

Table 11-5 Field Descriptions for the show tacacs-server Command

Field	Description
IP/Port	The TACACS+ server IP address and port number
State	The operational state of the server (Alive, Dying, or Dead) determined by the internal TCP Keepalive
Primary	Indicates whether this record is the primary TACACS+ server
Authen	The number of authentication requests made to the TACACS+ server
Author	The number of authorization requests made to the TACACS+ server
Account	The number of accounting requests made to the TACACS+ server
Global Timeout	How long the CSS waits for a response from a TACACS+ server
Global Key	Shared secret, used by all TACACS+ servers, unless individually configured for the server
Authorize Config Commands	Indicates whether configuration commands receive authorization
Authorize Non-Config	Indicates whether nonconfiguration commands receive authorization

Controlling Remote Access to the CSS

To control access to the CSS, you can configure the CSS to authenticate remote (virtual) or console users. The CSS can authenticate users by using the local user database, RADIUS server, or TACACS+ server. You can also allow user access without authenticating or disallowing all remote user access to the CSS.

You can set a maximum of three authentication methods: a primary, secondary, or tertiary authentication method. The primary method is the first authentication method that the CSS tries. If the primary authentication method fails (for example, the RADIUS server is down or is unreachable), the CSS tries the secondary method. And if the secondary method fails, then the CSS tries the tertiary method. In the event the tertiary method also fails, the CSS displays a message that authentication has failed.

The CSS does not attempt a secondary or tertiary authentication method under the following conditions:

- If the authentication method is **local**, and the local username is not found in the local user database.
- If the authentication method is **local** and the local username is found in the local user database, but the password is invalid.
- If the authentication method is **radius**, and the RADIUS server rejects the primary authentication request from the CSS.
- If the authentication method is **tacacs**, and the TACACS+ server rejects the primary authentication request from the CSS.

Before you can use RADIUS or TACACS+ as either the virtual authentication method or the console authentication method, you must enable communication with the RADIUS or TACACS+ security server. Use either the **radius-server** command (see the “[Configuring the CSS as a Client of a RADIUS Server](#)” section) or the **tacacs-server** command (see the “[Configuring the CSS as a Client of a TACACS+ Server](#)” section).

This section includes the following topics:

- [Configuring Virtual Authentication](#)
- [Configuring Console Authentication](#)

To display virtual and console authentication settings, use the **show user-database** command (refer to [Chapter 3, Managing the CSS Software](#)).

Configuring Virtual Authentication

Virtual authentication allows remote users to log in to the CSS when they are using FTP, Telnet, SSHD, or the Device Management user interface with or without requiring a username and password. The CSS can also deny access to all remote users.

You can configure the CSS to authenticate users by using the local database, RADIUS server, or TACACS+ server. By default, the CSS uses the local database as the primary method to authenticate users and disallows user access for the secondary and tertiary method.

Use the **virtual authentication** command to configure the primary, secondary, or tertiary virtual authentication method. The syntax for this global configuration command is:

```
virtual authentication [primary|secondary|tertiary  
[local|radius|tacacs|disallowed]]
```

The options for this command are as follows:

- **primary** - Defines the first authentication method that the CSS uses. The default primary virtual authentication method is the local user database.
- **secondary** - Defines the second authentication method that the CSS uses if the first method fails. The default secondary virtual authentication method is to disallow all user access.



Note If you are configuring a TACACS+ server as the primary authentication method, define a secondary authentication method, such as **local**.

- **tertiary** - Defines the third authentication method that the CSS uses if the second method fails. The default tertiary virtual authentication method is to disallow all user access.
- **local** - The CSS uses the local user database for authentication.
- **radius** - The CSS uses the configured RADIUS server for authentication.
- **tacacs** - The CSS uses the configured TACACS+ server for authentication.
- **disallowed** - The CSS disallows access by all remote users. Entering this option does not terminate existing connections.

To remove users currently logged in to the CSS, use the **disconnect** command.

To define the TACACS+ server as the primary virtual authentication method, enter:

```
 #(config) virtual authentication primary tacacs
```

To define local user database as the secondary virtual authentication method, enter:

```
 #(config) virtual authentication secondary local
```

Configuring Console Authentication

Console authentication allows users to log in to the CSS through a terminal connected to the console port with or without requiring a username and password. The CSS cannot disallow user access as a primary authentication method; however, it can disallow user access as a secondary or tertiary authentication method.

You can configure the CSS to authenticate users by using the local database, RADIUS server, or TACACS+ server. By default, the CSS uses the local database as the primary method to authenticate users and disallows user access for the secondary and tertiary method.

Use the **console authentication** command to configure the primary, secondary, or tertiary console authentication method. The syntax for this global configuration command is:

```
 console authentication [primary [local|radius|tacacs|none]  
 [secondary|tertiary [local|radius|tacacs|none|disallowed]]
```

The options for this command are as follows:

- **primary** - Defines the first authentication method that the CSS uses. The default primary console authentication method is the local user database.
- **local** - The CSS uses the local user database for authentication.
- **radius** - The CSS uses the configured RADIUS server for authentication.
- **tacacs** - The CSS uses the configured TACACS+ server for authentication.
- **none** - The CSS uses no authentication method. All users can access the CSS.
- **secondary** - Defines the second authentication method that the CSS uses if the first method fails. The default secondary console authentication method is to disallow all user access.



Note If you are configuring a TACACS+ server as the primary authentication method, define a secondary authentication method, such as **local**. If you do not configure a secondary method and use the default of **disallowed**, you have the possibility of being locked out of the CSS.

- **tertiary** - Defines the third authentication method that the CSS uses if the second method fails. The default tertiary console authentication method is to disallow all user access.
- **disallowed** - The CSS disallows access by all users (secondary or tertiary authentication method only). Entering this option does not terminate existing connections.

To remove users currently logged in to the CSS, use the **disconnect** command.

To define the TACACS+ server as the primary console authentication method, enter:

```
#(config) console authentication primary tacacs
```

To define local user database as the secondary console authentication method, enter:

```
#(config) console authentication secondary local
```

To disable authentication on the console port allowing users to access the CSS without a username and password, enter:

```
#(config) no console authentication
```

Controlling Access to the CSS

Use the **restrict** and **no restrict** commands to enable or disable console, FTP, SNMP, SSH, Telnet, user database, XML, and web management data transfer to the CSS. CSS access through a console, FTP, SSHD, SNMP, and Telnet is enabled by default. The CSS supports a maximum of four FTP sessions and a maximum of four Telnet sessions.

Specifying the **restrict** command does not prevent the CSS from listening for connection attempts on the restricted port. For TCP connections, the CSS completes the TCP 3-way handshake, then terminates the connection with an error to prevent any data transfer from occurring. For UDP SNMP connections, the CSS simply discards the packets.

To secure restricted ports from unauthorized access, configure ACL clauses to deny packets destined to these ports, while permitting normal traffic to flow through the CSS. You can also use ACLs to secure the CSS itself. Refer to the *Cisco Content Services Switch Basic Configuration Guide* for information about configuring ACLs for the CSS.

Enabling Access to the CSS

To enable console, FTP, SNMP, SSH, Telnet, user database, XML, and web management access to the CSS, use the following **no restrict** commands:

- **no restrict console** - Enables console access to the CSS (enabled by default)
- **no restrict ftp** - Enables FTP access to the CSS (enabled by default)
- **no restrict ssh** - Enables SSHD access to the CSS (enabled by default)
- **no restrict snmp** - Enables SNMP access to the CSS (enabled by default)
- **no restrict telnet** - Enables Telnet access to the CSS (enabled by default)
- **no restrict user-database** - Enables users to clear the running-config file and create or modify usernames. Only administrator and technician users can perform these tasks (enabled by default).
- **no restrict xml** - Enables XML access to the CSS (disabled by default)
- **no restrict web-mgmt** - Enables Device Management user interface access to the CSS (disabled by default)

**Note**

Disable Telnet access when you want to use the Secure Shell Host (SSH) server. For information about configuring SSH, see the “[Configuring the Secure Shell Daemon Protocol](#)” section.

For example, to enable Device Management user interface access, enter:

```
(config)# no restrict web-mgmt
```

Refer to [Chapter 12, Configuring Simple Network Management Protocol \(SNMP\)](#) for details on configuring the Simple Network Management Protocol (SNMP) features on your CSS. For details on making web-based configuration changes to the CSS using Extensible Markup Language (XML), refer to the *Cisco Content Services Switch Advanced Configuration Guide*. For details on using the Device Management user interface, refer to the *Cisco Content Services Switch Device Management User's Guide*.

Disabling Access to the CSS

To disable console, FTP, SNMP, SSH, Telnet, user database, XML, and web management access to the CSS, use the following **restrict** commands:

- **restrict console** - Disables console access to the CSS (enabled by default)
- **restrict ftp** - Disables FTP access to the CSS (enabled by default)
- **restrict snmp** - Disables SNMP access to the CSS (enabled by default)
- **restrict ssh** - Disables SSHD access to the CSS (enabled by default)
- **restrict telnet** - Disables Telnet access to the CSS (enabled by default)
- **restrict user-database** - Prevents users from clearing the running-config file and creating or modifying usernames. Only administrator and technician users can perform these tasks (enabled by default).
- **restrict xml** - Disables XML access to the CSS (disabled by default)
- **restrict web-mgmt** - Disables web management access to the CSS (disabled by default)

For example, to disable Telnet access, enter:

```
(config)# restrict telnet
```

Where to Go Next

[Chapter 12, Configuring Simple Network Management Protocol \(SNMP\)](#) describes how to configure SNMP on the CSS.