2 Configuring Users and Authentication Services

Providing security against unauthorized network access poses an important challenge to network administrators. An increase in Internet usage and the expansion of a private network are just two conditions that can make this challenge even more difficult.

Configuring users and authentication services allow you to define and control the access that users have to various network services. Doing so is imperative to ensuring that inappropriate access to sensitive information, deliberate or accidental, is not established.

The OmniCore routing switch provides a full suite of commands that allow you to create and modify user names for access to the Command Line Interface (CLI). Support for security services such as Access Control, Remote Authentication Dial-In User Service (RADIUS), and Terminal Access Controller Access Control System “plus” (TACACS+) are also provided.

Users and Authentication Services Commands

The major commands in the OmniCore CLI for configuring users and authentication services are listed in the following table (for TACACS+ authentication, see *TACACS+ Commands* on page 2-11). Other commands are available for fine-tuning your users and authentication services configuration. To see a complete list of these commands or for more information regarding the commands used in this chapter, see the *OmniCore CLI Reference Manual.*

Users and Authentication Services Commands

Parameter	Default	Description
utils access-control	0.0.0.0/0.0.0.0 (source IP address value), 0/0 (source port number value)	Creates an access control list (ACL) entry.
utils access-control snmp-access	deny	Identifies SNMP access control for the specified ACL entry.
utils access-control telnet-access	deny	Identifies Telnet access control for the specified ACL entry.
utils user auth-level	read-only	Defines authorization access for users wishing to access the CLI.
utils user length-checking	fixed minimum length of six characters	Designates the password length-checking mode.
utils user name	no default	Creates user names for access to the CLI.
utils user passwd	no default	Changes the CLI access password for the specified user.
utils user radius primary-server	0.0.0.0	Defines the primary RADIUS server for network user authentication.
utils user radius retries	3	Defines the RADIUS server retry value.

Users and Authentication Services Commands (Continued)

utils user radius second-server	0.0.0.0	Defines the secondary RADIUS server for network user authentication.
utils user radius secret	null string	Defines the RADIUS server secret.
utils user radius status	disable	Enables RADIUS on the OmniCore routing switch.
utils user radius timeout	1	Specifies the server response timeout value.
utils user radius udp-port	1812	Specifies the User Datagram Protocol (UDP) port with which the client will communicate with the RADIUS server.
utils user type	local	Changes the password type for the specified user.
utils user unconfigured	disallow	Defines the mode for authenticating unconfigured users.

Configuring Users and Passwords

A user must have a valid user name and password to access the CLI. The CLI is a text-based interface that is used to configure and monitor the OmniCore routing switch.

Configuring users and passwords requires one or more of the following tasks:

- Defining the password length-checking mode
- Defining the authentication mode for unconfigured users
- Creating a user name and password
- Defining a user's access privilege
- Defining a user's password type
- Changing a user's password
- Deleting a user name
- Viewing user name information

Defining the Password Length-Checking Mode

The password length-checking mode allows the CLI to accept or reject submitted passwords based on certain attributes. It also helps to diminish the number of common word passwords (i.e., easily discernible passwords, such as date of birth, system user name, or the user's real name) that unauthorized users may attempt to use. There are three possible password length-checking modes: *none*, *<1-80 characters>*, and *smart*.

If the mode is set to *none*, a password of any length or attribute can be assigned. If the mode is set to a fixed minimum length checking value (from 1 to 80 characters), all assigned passwords must meet the specified minimum value. If the mode is set to *smart*, a prospective password will be accepted only if the supplied text string meets one of the following conditions:

- It is at least six characters long.
- It contains at least one non-alphabetic (digit or symbol) and two alphabetic characters.

- It consists of all alphabetic characters of the same case (upper or lower) and is at least six characters long.
- It consists of all alphabetic characters, both uppercase and lowercase, and is at least five characters long.
- It consists only of non-alphanumeric (symbol) characters, or a mixture of uppercase and lowercase characters and numerals, and is at least four characters long.

The default password length-checking mode is a fixed minimum length of six characters. In the following example, a fixed minimum length checking value of five characters is specified.

```
OmniCore> utils user length-checking 5
```

Defining the Authentication Mode for Unconfigured Users

An unconfigured user is one for which no configuration information (user name and password) is locally stored by the OmniCore routing switch. If the authentication mode is set to *disallow*, all unconfigured users will be denied access to the switch. If the mode is set to *radius*, then an available RADIUS server can authenticate unconfigured users.

In this example, unconfigured users will be authenticated via a RADIUS server.

```
OmniCore> utils user unconfigured radius
```

Creating a User Name and Password

A user name establishes a means whereby the designated user can access the CLI. When creating a user name, you will be prompted to enter a password. Ensure that the submitted password meets the requirements of the designated password length-checking mode. Please note that when entering a password, no characters are displayed. In this example, a user name for John is created.

```
OmniCore> utils user

OmniCore/utils/user> John add
Enter password for "John":
Re-enter password:
Fri May 21, 1999 04:52:31.740: User John created from console.
```

Defining User Access Privilege

Upon successfully logging into the CLI, a user will have access to commands according to his or her access privilege level. There are three levels of user access:

- Super-user/admin – Users with this type of access can read and modify all CLI settings, including those available to privileged users. This access level is available by logging in as *admin* and supplying the valid password.
- Privileged – Users with privileged access can read and modify most CLI settings (some are available only to users with super-user/admin access). The keyword *write* is used to assign privileged access to a user.
- All users – Users with non-privileged access can view, but not modify, most CLI settings (some can be viewed only by users with super-user/admin or privileged access). The keyword *read-only* is used to assign non-privileged access to a user.

Note that only privileged (read-write) and non-privileged (read-only) access can be explicitly assigned to a user. The default access level for all new users names is non-privileged. In this example, user John is assigned with privileged access.

```
OmniCore/utils/user> John auth-level write
Fri May 21, 1999 04:57:31.840: User John authlevel changed from console.
```

Defining the Password Type

A user's password type specifies whether the password is stored and authenticated locally by the OmniCore routing switch (*local*), or by a RADIUS server (*radius*). The default type for all new user names is *local*. This example permits a RADIUS server to store and authenticate John's password.

```
OmniCore/utils/user> John type radius
Fri May 21, 1999 04:55:04.020: User john authtype changed from console.
```

Changing a Password

Any user, regardless of his or her access privilege, can change his or her password. Ensure that the submitted password meets the requirements of the designated password length-checking mode. Please note that when entering a password, no characters are displayed.

If you are changing your own password, you do not need to enter your user name.

```
OmniCore/utils/user> passwd
Enter new password:
Re-enter new password:
Fri May 21, 1999 04:53:52.120: User John password changed from console.
```

Only those with super-user/admin access can change the password for another user, as shown here.

```
OmniCore/utils/user> passwd Joe
Enter new password:
Re-enter new password:
Fri May 21, 1999 04:54:13.220: User Joe password changed from console.
```

Deleting a User Name

A user name can be deleted in the event a user no longer requires access to the switch. In this example, Joe's user name is deleted.

```
OmniCore/utils/user> Joe delete
Fri May 21, 1999 04:58:48.440: User Joe deleted from console.
```

Note that if a user name is deleted while it is currently being used, it will not be granted access only during future login attempts. The current session is not affected.

Viewing User Name Information

The user name table allows you to view comprehensive information for all CLI users, whether they are currently logged in or not. The Location column is used to ascertain where users are logged in from (i.e., the modem/console port or the IP address of a Telnet session). If a user name's Location entry is blank, that user is not currently logged in. Only those with super-user/admin access can view the user name table.

```
OmniCore/utils/user> show
User Name     Pwd Type     Authen. Level     Location
---------     --------     -------------     --------
admin         local        super-user        console
John          radius       read-write

   Password Length Checking: Smart
   Unconfigured Users      : radius
   Logged in               : admin
```

Configuring the Access Control List

Access control is the process of determining and enforcing the access rights each host has to services that the OmniCore routing switch provides, such as Telnet and SNMP. Access control information is maintained in the access control list (ACL). The ACL is made up of individual entries, each of which details a host's or set of hosts' (subnet) access rights.

Each ACL entry features two identifying values: a source IP address and a source port number. The source IP address value is combined with a subnet mask number to identify a single host or subnet. The source port number value identifies all hosts on a specified port. Each ACL entry must satisfy one of the following formats (note that the zeroes are required values for the desired entry type):

Access Control List Requirements

Entry Type	<*ipaddr*> Value	<*mask*> Value	<*slot*> Value	<*port*> Value
Default (always exists)	0.0.0.0	0.0.0.0	0	0
Source IP Address	x.x.x.x	y.y.y.y	0	0
Source Port Number	0.0.0.0	0.0.0.0	a	b
Source IP Address *and* Port Number	x.x.x.x	y.y.y.y	a	b

where *x.x.x.x* is an IP address, *y.y.y.y* is a subnet mask number, *a* is a slot number, and *b* is a port number, the values of which you supply. Note that the <*ipaddr*> and <*mask*> values must be chosen such that there are no bits in the <*ipaddr*> value which are not covered by the <*mask*> value.

When a host attempts to access a service on the switch, the ACL entry representing that host must match the access rights for the requested service. Entries are searched for the best possible match; the more specific values you supply, the more exact the entry. Entries are searched for a match in the following order:

- A matching source IP address *and* a matching source port number
- Only a matching source IP address
- Only a matching source port number
- Default entry (specifying any IP address and any port)

The default entry of 0.0.0.0/0.0.0.0 (any IP address) and 0/0 (any port) always exists for any IP address and/or port combination which does not constitute a match. Note that the default entry automatically exists and cannot be deleted.

The following procedure describes how to create and delete ACL entries, and how to modify access rights for those entries. Note that only those with super-user/admin access can create or delete ACL entries.

Follow these steps to configure the Access Control List:

1. Modify the access rights for the default entry, which by default is defined with *read-write* access control to SNMP and with *permit* access control to Telnet. For this example, the default entry is assigned with *deny* access to Telnet and with *read-only* access to SNMP.

```
OmniCore> utils
OmniCore/utils> access-control 0.0.0.0 0.0.0.0 0 0
OmniCore/utils/access-control=0.0.0.0,0.0.0.0,0,0> telnet-access deny
OmniCore/utils/access-control=0.0.0.0,0.0.0.0,0,0> snmp-access read-only
```

2. (Optional) Create additional ACL entries, all of which are by default defined with *deny* access control to both SNMP and Telnet.

```
OmniCore/utils/access-control=0.0.0.0,0.0.0.0,0,0> ..
OmniCore/utils> access-control 10.0.3.34 255.255.0.0 ethernet 5 4 create
OmniCore/utils> access-control 10.0.45.0 255.0.0.0 0 0 create
OmniCore/utils> access-control 192.168.3.1 255.255.255.255 0 0 create
OmniCore/utils> access-control 0.0.0.0 0.0.0.0 3 5 create
```

3. (Optional) Set the Telnet access privilege for each desired ACL entry.

```
OmniCore/utils> access-control 0.0.0.0 0.0.0.0 3 5 telnet-access permit
OmniCore/utils> access-control 10.0.45.0 255.0.0.0 0 0 telnet-access reject
```

4. (Optional) Set the SNMP access privilege for each desired ACL entry.

```
OmniCore/utils> access-control 0.0.0.0 0.0.0.0 3 5 snmp-access read-only
OmniCore/utils> access-control 192.168.3.1 255.255.255.255 0 0 snmp-access reject
```

5. (Optional) Delete an ACL entry.

```
OmniCore/utils> access-control 192.168.3.2 255.255.255.0 6 15 delete
```

```
OmniCore/utils> access-control show
IP Address    Mask              Slot      Port   Telnet   SNMP
-----------   ----              ----      ----   ------ - ----
0.0.0.0       0.0.0.0           0         0      deny     read-only
0.0.0.0       0.0.0.0           3         5      permit   read-only
10.0.3.34     255.255.0.0       5         4      deny     deny
10.0.45.0     255.0.0.0         0         0      reject   deny
192.168.3.1   255.255.255.255   0         0      deny     reject
```

Configuring RADIUS

Remote Authentication Dial-In User Service (RADIUS) is designed to provide a protected, central location where network security information is maintained and authenticated. RADIUS therefore guards against unauthorized network access by maintaining a database of user profiles.

RADIUS architecture consists of RADIUS servers and clients. Each server uses a database of user profiles to authorize or deny requests from a client wishing to authenticate a user. The OmniCore routing switch implements a RADIUS client to allow for centralized authentication.

Follow these steps to configure RADIUS:

1. Enable RADIUS. Note that if RADIUS is disabled, users who have been configured for authentication through RADIUS will not be able to log onto the system.

```
OmniCore> utils user radius

OmniCore/utils/user/radius> status enable
```

2. Define the primary RADIUS server to be used for network user authentication.

```
OmniCore/utils/user/radius> primary-server 10.0.101.141
```

3. (Optional) Define the secondary RADIUS server that will be used in the event the primary RADIUS server does not respond to a query.

```
OmniCore/utils/user/radius> second-server 10.0.0.101

OmniCore/utils/user/radius> show
RADIUS Protocol Status                    :enable
Primary Authentication Server             :10.0.101.141
Secondary Authentication Server           :10.0.0.101
UDP Port                                  :1812
Number of Request Retries                 :3
Server Response Timeout                   :1  sec
```

4. Define the RADIUS server secret to be shared by the system and the RADIUS server. Only those with super-user/admin access can define the server secret. For added security, no characters will be displayed as you enter the desired text.

```
OmniCore/utils/user/radius> secret
Enter new secret:
Re-enter new secret:
```

5. (Optional) Specify that a RADIUS server will authenticate unconfigured users. An unconfigured user is one for which no configuration information (user name and password) is locally stored by the OmniCore routing switch.

```
OmniCore/utils/user/radius> ..

OmniCore/utils/user> unconfigured radius
```

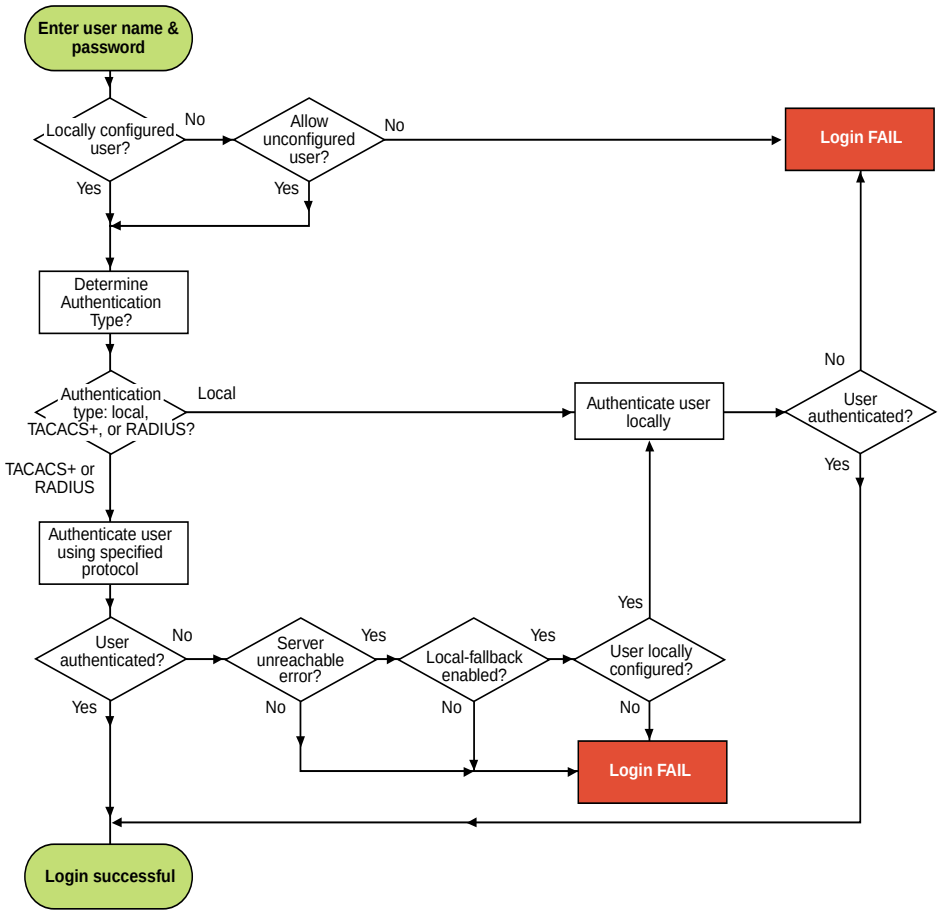
Configuring TACACS+

Terminal Access Controller Access Control System plus (TACACS+) protocol provides access control for routers, network-access servers and other networked computing devices via one or more centralized servers. TACACS+ provides separate authentication, authorization and accounting services. It is implemented on the OmniCore routing switch to provide user authentication, command authorization (including shell-exec), and command accounting using a designated TACACS+ daemon.

The TACACS+ daemon uses a centralized configuration file containing user-name and password entries, user privilege levels, and user-executable commands. TACACS+ relies on TCP for its transport and listens on commonly known port 49. TACACS+ services, e.g. authorization, authentication, and accounting are enabled and disabled using the switch's command line interface (CLI).

Authentication

Authentication is the action of determining who a user is (entity). Authentication on an OmniCore routing switch is mandatory, however, using TACACS+ for authentication is not mandatory and is a site configured option. The authentication login process is shown below.



Authentication Login Flow Process

When enabled, TACACS+ authentication is performed for local switch-configured users of *type* TACACS+, and for users not locally configured on the switch if the switch's *unconfigured* option is set to *tacacsplus*.

Authentication always takes place when the user first logs onto the OmniCore routing switch, and is used to verify (authenticate) the existence of a user and password combination in the daemon's configuration. The client implementation supports SecurId authentication using the "Password" prompt for the PIN/CODE input.

Authorization

Authorization is the action of determining what a user is allowed to do. In TACACS+, authorization can be used to customize the service for a particular user or users. Examples of when authorization would be performed are: when a user first logs on and wants to start a executable shell (CLI session), copy a file, or set a value. The TACACS+ daemon will respond by permitting or denying the request.

In an OmniCore routing switch, the authorization exchange is used to verify that the user has the authority to execute a specified command, see *Commands Sent to TACACS+ for Authorization* on page 2-12. Authorization is also used in the login process to learn the user's TACACS+ privilege level, and to determine if the user can start an executable shell (CLI session).

When enabled, TACACS+ authorization is attempted for all users of the switch, regardless of the method used to authenticate the user (see step 7, *Enabling Authorization* on page 2-15).

TACACS+ Privilege Level and Switch Authority Level

The privilege level for a TACACS+ user is determined during the shell-exec authorization process, and is always performed for users authenticating with TACACS+. The configuration file for the daemon may have an executable privilege level (see example below), ***n***, explicitly specified for the user (or user-group of which the user is a member) from 0 to 15, where 0 denotes the minimum privilege (read-only) and 15 the maximum privilege (admin).

Example (from CISCO freeware):

```
user = adminuser {
  member = admingroup
  login = cleartext tacplus
}

group = admingroup {
 service = exec {
    priv-lvl = n
    }
}
```

The privilege level is returned by the daemon as an "attribute-value" pair in the shell-exec authorization response packet. If no privilege level has been configured for the user (on the daemon), then, depending on the daemon, the shell-exec authorization may fail, or the daemon may return a privilege level of 1 for the user. If shell-exec authorization fails due to no privilege level the network administrator must configure a level for the user on the daemon before successful login to the switch.

Since there are three recognized levels of authority on the switch (0 = admin, 1 = read-write, and 2 = read-only), and sixteen levels for TACACS+, the learned TACACS+ privilege level must be mapped to a switch authority level. The minimum TACACS+ privilege level necessary for read-write access on the switch can be configured using the *rwprivlvl* command through the CLI.

Privilege levels *must* be assigned to users (or groups) on the TACACS+ daemon in the following manner:

Privilege Levels of Authority

TACACS+ Privilege Level	Switch Authority Level	
15	0	admin (super-user)
2-14	1 If user's TACACS+ privilege level => privilege level (rwprivlvl) of switch	read-write
	2 If user's TACACS+ privilege level < privilege level (rwprivlvl) of switch	read-only
0-1	2	read-only

Accounting

Accounting, as implemented on the OmniCore routing switch, is the action of recording what a user is attempting to do or has done. Accounting is a service independent of authentication or authorization. An account log specified by the TACACS+ daemon is used to keep a record of the user's activities on the switch. The TACACS+ accounting protocol must be enabled for accounting to succeed. Items on the switch that are currently logged are:

- User login and logout
- Total login time, in seconds
- CLI shell commands, whether successfully executed or not

TACACS+ Commands

The TACACS+ commands are listed in the following tables. For other commands or more information regarding TACACS+ commands, see the *OmniCore CLI Reference Manual*. Users can access the TACACS+ submenu via the CLI *utils user tacacs+* menu. Items can only be changed by a user with administrative authority.

Executing CLI Commands

Permission to execute commands is filtered at the switch prior to sending the authorization request to the TACACS+ daemon:

- If the TACACS+ authorization protocol is disabled, permission will be granted based on the user's authority-level at the switch.
- If the TACACS+ authorization protocol is enabled, authorization will be deferred to the TACACS+ daemon, see *Commands Sent to TACACS+ for Authorization* on page 2-12.

TACACS+ Commands

Command	Default	Description
utils user tacacs+ authen-status	disable	Enables or disables the TACACS+ authentication protocol. This must be enabled in order for TACACS+ users to login to the switch.
utils user tacacs+ author-status	disable	Enables or disables the TACACS+ authorization protocol.
utils user tacacs+ account-status	disable	Enables or disables the TACACS+ accounting protocol.
utils user tacacs+ primary-server	no default	IP address of primary TACACS+ daemon (server)
utils user tacacs+ secondary-server	no default	IP address of secondary TACACS+ daemon (server)
utils user tacacs+ timeout	5 seconds	Response timeout. This the total amount of time the TACACS+ client will wait for a response from the daemon.
utils user tacacs+ rwprivlvl	2	Minimum TACACS+ privilege level required for switch read-write authority.
utils user tacacs+ secret	no default	Sets TACACS+ encryption key. This key must match the key that is in use by the daemon or transmission error will occur.

Commands Sent to TACACS+ for Authorization

System users should be aware that the following OmniCore commands are sent to the TACACS+ daemon for authorization. For descriptions of these commands, refer to the *OmniCore CLI Reference Manual.*

- add
- atping
- copy
- delete
- dir
- format
- get
- install
- ipxping
- load-config
- passwd
- ping
- put
- reboot
- rename
- save
- set
- show
- telnet
- trace
- view
- write-config

Commands Not Sent to TACACS+ for Authorization

The following OmniCore commands are not sent to the TACACS+ daemon for authorization. For descriptions of these commands, refer to the *OmniCore CLI Reference Manual.*

- login
- logout
- quit
- exit
- help
- history
- tree
- ? (interactive help)

Configuring TACACS+

Exercise care when setting up the TACACS+ client on the OmniCore routing switch or a "lock-out" condition can occur. A lock-out condition may be encountered if the MD5 encryption keys do not match or the primary and secondary addresses are incorrect. (For information on recovering from a lock-out condition, see *Recovering From Lock-Out* on page 2-16.)

TACACS+ has a base set of CLI commands and menus accessible through the *utils user tacacs+* menu. Configuring the OmniCore routing switch for TACACS+ services consists of the following tasks.

- Setting up the user.
- Setting the secret key.
- Specifying the primary and secondary server.
- Verifying servers are accessible.
- Enabling the local-fallback option.
- Enabling TACACS+ Authentication.
- Enabling TACACS+ Authorization.
- Enabling TACACS+ Accounting.

1. Setting Up the User

a. Locally Configured Users

Configure local users for TACACS+ authentication with the following CLI command: *utils user type tacacsplus*. If you plan to authenticate RADIUS users with TACACS+, set the user's type to TACACS+. For example:

```
OmniCore> utils user JohnDoe type tacacsplus
```

✦ Note ✦

Administrators may also want to set the local “admin” user type to TACACS+ to force TACACS+ authenication on this user.

b. Authority Level

Assign the user an authority level with the following CLI command: *utils user auth-level*. For example:

```
OmniCore> utils user JohnDoe auth-level write
```

c. Unconfigured Users Option

Users not locally configured on the switch can be authenticated by TACACS+ by setting the CLI *utils user unconfigured* command to *tacacsplus*. For example:

```
OmniCore> utils user unconfigured tacacsplus
```

2. Setting the Secret Key

Enter the secret key using the CLI *utils user tacacs+ secret* command. The secret key on the switch must match the daemon's key exactly for correct deciphering of the TACACS+ packet data. A key string with spaces included must be enclosed in double-quotes. For examples:

```
OmniCore> utils user tacacs+ secret “secretKeystring with spaces”
OmniCore> utils user tacacs+ secret secretKeystringNoSpaces
```

If the keys do not match, the packet data will be unrecognizable and TACACS+ will report an error. If no key is set, the switch will *not* encrypt the packet data and will set the unencrypted flag in the TACACS+ header to 0x01. For security reasons, a key should be set. However, a zero-length key may be set by typing "" (two double-quotes) for the string.

3. Specifying Primary and Secondary Server

At least one address must be specified in order to attempt a connection over TCP. The OmniCore routing switch uses the well-known port 49 to establish a connection with the daemon. TACACS+ will attempt to connect to the primary server first, followed by the secondary. If no server is reachable via the specified addresses, the local-fallback option (see *Enabling Local-Fallback* on page 2-14) will attempt to compensate.

a. Primary Server

To specify a primary server, use the CLI *utils user tacacs+ primary-server* command. For example:

```
OmniCore> utils user tacacs+ primary-server 10.102.0.0
```

b. Secondary Server

To specify a secondary server, use the CLI *utils user tacacs+ secondary-server* command. For example:

```
OmniCore> utils user tacacs+ secondary-server 10.102.0.1
```

4. Verifying Servers are Accessible

Ping the primary and secondary servers to verify they are accessible, for example:

```
OmniCore> ping 10.102.0.0
```

5. Enabling Local-Fallback

Ensure the CLI *utils user local-fallback* command is enabled (the default is enabled). For example:

```
OmniCore> utils user local-fallback enable
```

This option aides in recovering the switch from a “lock-out” condition. A lock-out condition may be encountered if the MD5 encryption keys do not match, the primary and secondary addresses are incorrect, or the current user is not configured on the TACACS+ daemon explicitly or by default profile. In the case of an unreachable daemon, such as incorrect IP address, daemon not present, or the network is down or unreachable, the switch will grant access based on its local knowledge of the user and if the local-fallback option is enabled.

6. Enabling Authentication

❖ Important! ❖

It is important that the secret key and server addresses are correctly configured prior to enabling TACACS+ aAuthentication, authorization, or accounting services.

For successful TACACS+ user authentication, you must enable TACACS+ user authentication with the CLI *utils user tacacs+ authen-status* command. For example:

```
OmniCore> utils user tacacs+ authen-status enable
```

7. Enabling Authorization

❖ Important! ❖

It is important that the secret key and server addresses are correctly configured prior to enabling TACACS+ authentication, authorization, or accounting services.

Before enabling TACACS+ authorization, ensure the user is configured specifically or by-default on the daemon. Once enabled, all authorization will go to through TACACS+, regardless of the method used to authenticate the user.

For command authorization to flow through TACACS+, the switch's TACACS+ authorization feature must be enabled. To enable, use the CLI *utils user tacacs+ author-status* command. For example:

```
OmniCore> utils user tacacs+ author-status enable
```

When enabled, commands from all users will be authorized using TACACS+ regardless of the method used to authenticate the user. For authorization to succeed (whether the command is permitted or denied), the administrator should configure the default user profile on the daemon. This allows users of the switch that are not configured on the daemon to pass command authorization. If the DEFAULT or UNKNOWN_USER profile is not configured, the daemon will return an error for any user it fails to recognize, and the authorization will fail. The following stanza shows the default user configuration profile (CISCO freeware) of the TACACS+ server.

```
user = DEFAULT {
  member = admingroup
  login = nopassword
}
```

In this example, the default user would inherit the privileges of the group called "admingroup".

8. Enabling Accounting

❖ Important! ❖

It is important that the secret key and server addresses are correctly configured prior to enabling TACACS+ authentication, authorization, or accounting services.

For successful TACACS+ command accounting, the TACACS+ Accounting service must be enabled. To enable, use the CLI *utils user tacacs+ account-status* command. For example:

```
OmniCore> utils user tacacs+ account-status enable
```

Accounting will be performed via TACACS+ for all users regardless of the method used to authenticate the user

Recovering From Lock-Out

If local-fallback is enabled (see *Enabling Local-Fallback* on page 2-14) a lock-out condition may be overcome by disrupting the "reachability" of the server. The easiest way to do this is to temporarily stop the daemon or make it inaccessible by pulling the physical link (TACACS+ network connector) from the switch. Once the server is not responding, the switch will fallback to authenticate or authorize based on the local switch configuration.

❖ Note ❖

There is a delay time at switch reboot for configured IP interfaces and routes to populate. Since TACACS+ relies on TCP for its transport, TACACS+ interaction during this time may fail and go to local-fallback if enabled.