Chapter 6

Using Commands and Statements to Configure the SRC Software

This chapter describes how to use the CLI to configure the SRC software. Topics include:

Understanding CLI Configuration Mode

In configuration mode, you can configure properties for the SRC software, such as properties for the Juniper Networks database, SRC components, user access, and system properties. You can configure all SRC properties for the C-series platform, but only a subset of these properties if you are running SRC software on a Solaris platform.

A configuration is stored as a hierarchy of statements. In configuration mode, you create the specific hierarchy of configuration statements that you want to use. When you have finished entering the statements, you commit them, which activates the configuration.

You can create the hierarchy interactively at the CLI, or you can a load configuration from a file that you create. To activate the configuration, you commit it.

For information about loading a configuration from a file, see *Chapter 10, Managing Configurations*.

For information about CLI support for SRC installations on Solaris platforms, see *Chapter 3, SRC CLI Basics*.

Configuration Mode Commands

Table 9 summarizes each CLI configuration mode command. The commands are listed alphabetically.

Table 9: Summary of Configuration Mode Commands

Command	Description
commit	Commit the set of changes to the database and cause the changes to take operational effect.
delete	Delete a statement or identifier. All subordinate statements and identifiers contained within the specified statement path are deleted with it.
edit	Move inside the specified statement hierarchy. If the statement does not exist, it is created.
exit	Exit the current level of the statement hierarchy, returning to the level before the last edit command, or exit from configuration mode. The quit and exit commands are synonyms.
help	Display help about available configuration statements.
history	Display the previous commands entered at the CLI.
insert	Insert an identifier into an existing hierarchy.
load	Load a configuration from a file. Your current location in the configuration hierarchy is ignored when the load operation occurs.
quit	Exit the current level of the statement hierarchy, returning to the level before the last edit command, or exit from configuration mode. The quit and exit commands are synonyms.
rename	Rename an existing configuration statement or identifier.
rollback	Return to the previously committed configuration. The software saves only the last committed configuration.
run	Run an operational mode CLI command without exiting from configuration mode.
save	Save the configuration to an ASCII file. The contents of the current level of the statement hierarchy (and below) are saved, along with the statement hierarchy containing it. This action allows a section of the configuration to be saved, while fully specifying the statement hierarchy.
set	Create a statement hierarchy and set identifier values. This command is similar to edit except that your current level in the hierarchy does not change.
show	Display the current configuration.
top	Return to the top level of configuration command mode, which is indicated by the [edit] banner.
up	Move up one level in the statement hierarchy.

For more information about configuration mode commands, see *SRC-PE CLI Command Reference*.

Configuration Statements

You configure SRC properties by including statements in the configuration. A statement consists of the following parts:

- Keyword—Fixed text
- Identifier (Optional)—Identifying name that you define, such as the name of an interface, or a username, which allows you and the CLI to discriminate among a collection of statements

Table 10 describes top-level CLI configuration mode statements.

Table 10: Configuration Mode Top-Level Statements

Statement	Description
interfaces	Configure interfaces on the C-series platform.
policies	Configure routing policies.
redirect-server	Configure the redirect server.
routing-options	Configure static routes.
services	Define subscriber services.
shared	Configure ACP, admission control, congestion points, auth-cache, network devices, NIC, and SAE.
slot	Configure properties for a component, such as ACP, Juniper Policy Server, network information collector, and SAE on a slot.
snmp	Configure Simple Network Management Protocol (SNMP) community strings, interfaces, traps, and notifications.
subscribers	Configure subscriber definitions.
system	Configure systemwide properties, including the hostname, domain name, Domain Name System (DNS) server, user logins and permissions, and software processes.

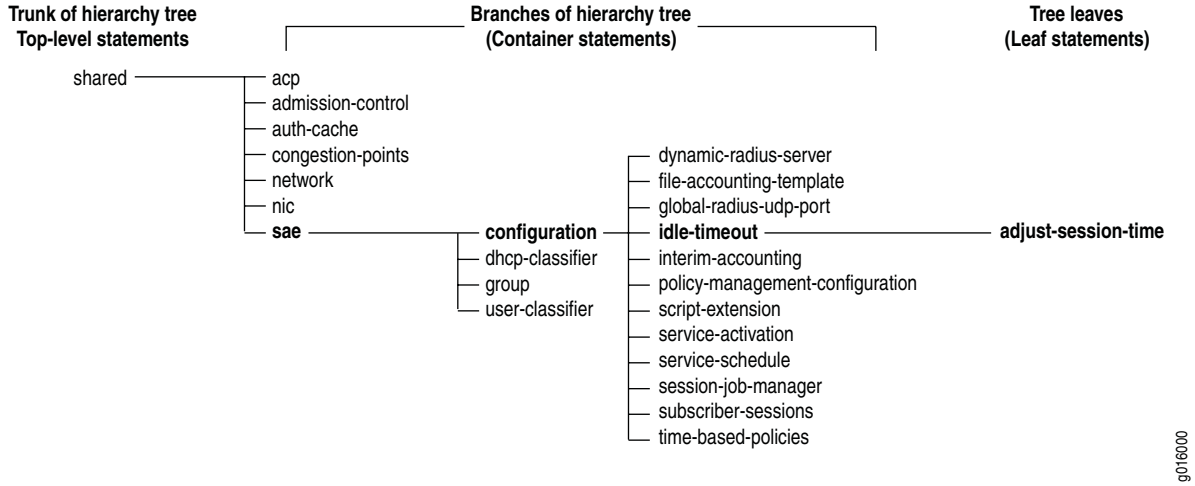
For information about specific configuration statements, see the *SRC-PE CLI Command Reference*.

Configuration Statement Hierarchy

The SRC software configuration consists of a hierarchy of *statements*. There are two types of statements: *container statements*, which are statements that contain other statements, and *leaf statements*, which do not contain other statements. All the container and leaf statements together form the *configuration hierarchy*.

Figure 9 shows container statements and leaf statements in the sae hierarchy. To view this hierarchy at the CLI, the editing level must be set to expert.

Figure 9: Sample Configuration Mode Hierarchy of Statements



Each statement at the top level of the configuration hierarchy resides at the trunk (or root level) of a hierarchy tree. The top-level statements are container statements, containing other statements that form the tree branches. The leaf statements are the leaves of the hierarchy tree. An individual hierarchy of statements, which starts at the trunk of the hierarchy tree, is called a *statement path*. Figure 9 illustrates the hierarchy tree, showing a statement path for the portion of the shared configuration hierarchy that configures the idle timeout for the SAE.

The shared statement is a top-level statement at the trunk of the configuration tree. The acp, admission-control, auth-cache, congestion-points, network, nic, and sae statements are all subordinate container statements of the shared statement (they are branches of the shared hierarchy tree).The configuration and the idle-timeout statements are successive branches in the hierarchy under the sae branch. The adjust-session-time statement is a leaf on the tree, which, in this case, specifies that when a session is terminated by an idle timeout, the session time reported in the accounting stop message is automatically reduced by the idle time.

The CLI represents the statement path shown in Figure 9 on page 54 as [shared sae configuration idle-timeout], and displays the configuration as follows:

```
shared {
    sae {
        configuration {
            idle-timeout {
                adjust-session-time;
            }
        }
    }
}
```

The CLI indents each level in the hierarchy to indicate each statement's relative position in the hierarchy and generally sets off each level with braces, using an open brace at the beginning of each hierarchy level and a closing brace at the end. Each leaf statement ends with a semicolon.

Entering and Exiting Configuration Mode

You configure the SRC software by entering configuration mode and creating a hierarchy of configuration mode statements.

Users must have configure permission to view and use the configure command to enter configuration mode. When in configuration mode, users can view and modify only those statements for which they have access privileges set.

Entering Configuration Mode

To enter configuration mode:

- Use the configure command.

  ```
  user@host> configure
  Entering configuration mode.
  ```

When you enter configuration mode, the following configuration mode commands are available:

```
[edit]
user@host# ?
Possible completions:
    commit              Commit a set of changes
    delete              Delete a configuration statement or identifier
    edit                Specify edit level in hierarchy
    exit                Exit from this level
    help                Display help about commands and statements
    history             Display command history
    insert              Insert an identifier
    load                Load configuration from an ASCII file
    rename              Rename a statement or identifier
    rollback            Discard current set of changes
    run                 Run an operational mode command
    save                Save configuration to an ASCII file
    set                 Set a configuration property
    show                Display configuration information
    top                 Return to top level of configuration mode
    up                  Move up one level in hierarchy
```

Exiting from Configuration Mode

To exit configuration mode:

- Use the exit configuration-mode command from any level.

  or

  Use the exit command from the top level.

For example:

```
[edit shared sae configuration time-based-policies]
user@host# exit configuration-mode
Exiting configuration mode.
user@host>

[edit]
user@host# exit
Exiting configuration mode.
user@host>
```

If you try to exit from configuration mode by using the exit command and the configuration contains changes that have not been committed, you see a message:

```
[edit shared sae configuration time-based-policies]
user@host# exit configuration-mode
Exiting configuration mode.
The configuration has been changed but not committed.

user@host>
```

Modifying the Configuration

To configure the SRC software or to modify an existing configuration, you add statements to the configuration. For each statement hierarchy, you create the hierarchy starting with a statement at the top level and continuing with statements that move progressively lower in the hierarchy.

To modify the hierarchy, you use two configuration mode commands:

- edit—Moves to a particular hierarchy level. If that hierarchy level does not exist, the edit command creates it. The edit command has the following syntax:

  edit <*statement-path*>

  where *statement-path* is the hierarchy to the configuration statement and the statement itself.

- set—Creates a configuration statement and sets identifier values. After you issue a set command, you remain at the same level in the hierarchy. The set command has the following syntax:

  set <*statement-path*> *statement* <*identifier*>

  where

  statement-path is the hierarchy to the configuration statement and the statement itself. If you have already moved to the statement's hierarchy level, you can omit the statement path.

  statement is the configuration statement itself.

identifier is a string that identifies an instance of a statement.

You cannot use the edit command to change the value of identifiers. You must use the set command.

Entering Values for Statement Options

When values include the following characters—space, single quotation marks (‘), double quotation marks (“), curly braces ({}) , brackets ([ ]), or commas (,)—you must enclose the value in quotation marks (“ “) or use a backslash (\) before the character.

To enter words or letters separated by a space, such as a full name with a first name and last name, enclose the words in quotation marks. For example:

"Chris Bee"

To enter multiple values, separate values with a space, and enclose the values with brackets. For example:

[192.0.2.24 192.0.4.25]

To enter a number using a regular expression, use backslashes (\) to escape the brackets. For example:

\[0-9\]

Displaying the Current Configuration

You can display the current configuration from operational mode or from configuration mode. In configuration mode, you can display the configuration at the specified hierarchy level.

To display the current configuration from configuration mode:

- Use the show command.

  [edit]
  user@host# **show** <*statement-path*>

  or

  [edit <*hierarchy-level*>]
  user@host# **show**

To display the current configuration from operational mode:

- Use the show configuration command.

  user@host> **show configuration**

The configuration statements appear in a fixed order; however, when you configure the C-series platform, you can enter statements in any order.

If you omit a required statement at a particular hierarchy level, when you issue the show or show configuration command, a message indicates which statement is missing. As long as a mandatory statement is missing, the CLI continues to display this message each time you issue a show or show configuration command.

For example, the following output includes a warning that lists mandatory attributes that need to be configured:

```
[edit]
user@host# show
. . .
shared {
  sae {
    configuration {
      aggregate-services {
        keepalive-time 172800;
        warning: missing mandatory attribute(s): 'keepalive-retry-time',
'activation-deactivation-time', 'failed-notification-retry-time'
      }
    }
   . . .
  }
}
```

Examples: Displaying the Current Configuration

Configure timers for aggregate services from the [edit] hierarchy level, and then view the configuration from the same hierarchy level:

```
[edit]
user@host# set shared sae configuration aggregate-services keepalive-time 86400
[edit]
user@host# set shared sae configuration aggregate-services keepalive-retry-time 900
[edit]
user@host# set shared sae configuration aggregate-services activation-deactivation-time 900
[edit]
user@host# set shared sae configuration aggregate-services failed-notification-retry-time 86400

[edit]
user@host# show
. . .
shared {
  sae {
    configuration {
      aggregate-services {
        keepalive-time 86400;
        keepalive-retry-time 900;
        activation-deactivation-time 900;
        failed-notification-retry-time 86400;
      }
    }
   . . .
  }
}
```

Display a configuration at a specific hierarchy level:

```
[edit]
user@host# show shared sae configuration aggregate-services
keepalive-time 172800;
keepalive-retry-time 900;
activation-deactivation-time 900;
failed-notification-retry-time 86400;
```

Move to a lower level in the hierarchy, [edit shared sae configuration aggregate-services], and then display the configuration at that level:

```
[edit]
user@host# edit shared sae configuration aggregate-services

[edit shared sae configuration aggregate-services]
user@host# show
keepalive-time 172800;
keepalive-retry-time 900;
activation-deactivation-time 900;
failed-notification-retry-time 86400;
```

Display all of the last committed configuration from operational mode:

```
[edit]
user@host# set shared sae configuration aggregate-services keepalive-time 172800
[edit]
user@host# set shared sae configuration aggregate-services keepalive-retry-time 900
[edit]
user@host# set shared sae configuration aggregate-services activation-deactivation-time 900
[edit]
user@host# set shared sae configuration aggregate-services failed-notification-retry-time 86400
[edit]
user@host# show
user@host# commit
commit complete.
[edit]
user@host# quit
Exiting configuration mode.

user@host> show configuration
. . .
shared {
  sae {
    configuration {
      aggregate-services {
        keepalive-time 172800;
        keepalive-retry-time 900;
        activation-deactivation-time 900;
        failed-notification-retry-time 86400;
      }
    }
   . . .
  }
}
```

Adding Configuration Statements and Identifiers

When you use the **?** help to view a list of possible command completions, the output includes symbols that provide more information about the statement. The following symbols can appear in a list:

- Angle bracket (>) before the statement name indicates that the statement is a container statement and that you can define other statements at levels below it.
- No angle bracket (>) before the statement name indicates that the statement is a leaf statement; you cannot define other statements at hierarchy levels below it.
- Plus sign (+) before the statement name indicates that the statement can contain a set of values. To specify a set, include the values in brackets.
- Asterisk (*) before a statement name indicates a required statement or option that is not configured.
- Plus/Asterisk (+ *) before a statement name indicates a required option that can contain a set of values.

The following example at the [edit system services] hierarchy level shows that authentication-order, domain-search, and name-server can contain more than one value.

```
[edit system]
user@host# show ?
Possible completions:
  <[Enter]>            Execute this command
+ authentication-order Order in which authentication methods are invoked
+ domain-search        List of domain names to search
  host-name            Hostname for SDX platform
> ldap                 LDAP properties
> login                Login properties
+ name-server          DNS name servers
> ntp                  NTP configuration
> radius-server        RADIUS server configuration
> services             System services configuration
> syslog               System log configuration
> tacplus-server       TACACS+ server configuration
  time-zone            Time zone definition name
  |                    Pipe through a command
```

The following example at the [edit shared sae configuration driver] hierarchy level shows that mac-cache-expiration and unauthenticated-subscriber-dn are required statements.

```
[edit shared sae configuration driver]
user@host# show ?
Possible completions:
  <[Enter]>            Execute this command
> junos                Parameters the SAE uses to manage JUNOS routing platforms
> junose               Parameters the SAE uses to manage JUNOSe routers
* mac-cache-expiration Time that a subscriber profile remains in SAE cache (0..I
NF s)
> pcmm                 Parameters that SAE uses to manage PCMM devices
> scripts              Scripts to customize setup of SAE connections to devices
> simulated            Parameters that the SAE uses for simulated drivers
```

```
> snmp                  Global default SNMP communities
> third-party           Parameters that SAE uses to manage third-party devices
* unauthenticated-subscriber-dn
                        DN of an unauthenticated subscriber profile
  virtual-portal-address
                        IP address of the portal server
  |                     Pipe through a command
```

If you do not type an option for a statement that requires one, a message indicates the type of information expected. In this example, you need to type an area number as an identifier to complete the logger name:

```
[edit]
user@host# set shared sae configuration logger
                                                 ^
syntax error, expecting <identifier>.
```

Deleting a Statement from the Configuration

Deleting a statement or an identifier effectively "unconfigures" the functionality associated with that statement or identifier, returning that functionality to its default condition.

To delete a statement or identifier:

- Use the delete configuration mode command.

  user@host# **delete** *<statement-path> <identifier>*

When you delete a statement, the statement and all its subordinate statements and identifiers are removed from the configuration.

For statements that can have more than one identifier, when you delete one identifier, only that identifier is deleted. The other identifiers in the statement remain.

To delete the entire hierarchy starting at the current hierarchy level:

- In configuration mode, use the delete command. Do not specify a statement or an identifier.

  When you omit the statement or identifier, you are prompted to confirm the deletion. For example:

  ```
  [edit]
  user@host# delete
  Delete everything under this level? [yes, no] (no)?
  ```

Examples: Deleting a Statement from the Configuration

Configure the aggregate-services statements, then delete these statements from the [edit] level. Using the delete command effectively unconfigures the SAE properties for aggregate-services in the SRC software:

```
[edit]
user@host# set shared sae configuration aggregate-services keepalive-time 172800
[edit]
user@host# set shared sae configuration aggregate-services keepalive-retry-time 900
[edit]
user@host# set shared sae configuration aggregate-services activation-deactivation-time 900
[edit]
user@host# set shared sae configuration aggregate-services failed-notification-retry-time 86400

[edit]
user@host# show
. . .
shared {
  sae {
    configuration {
      aggregate-services {
        keepalive-time 172800;
        keepalive-retry-time 900;
        activation-deactivation-time 900;
        failed-notification-retry-time 86400;
      }
    }
   . . .
  }
}

[edit]
user@host# delete shared sae configuration aggregate-services

[edit]
user@host# show shared sae configuration aggregate-services
```

Configure the aggregate-services statements, then delete these statements from the [edit shared sae configuration aggregate-services] level:

```
[edit]
user@host# set shared sae configuration aggregate-services keepalive-time 172800
[edit]
user@host# set shared sae configuration aggregate-services keepalive-retry-time 900
[edit]
user@host# set shared sae configuration aggregate-services activation-deactivation-time 900
[edit]
user@host# set shared sae configuration aggregate-services failed-notification-retry-time 86400
[edit]
user@host# edit shared sae configuration aggregate-services

[edit shared sae configuration aggregate-services]
user@host# delete
Delete everything under this level? [yes,no] (no) yes

[edit shared sae configuration aggregate-services]
user@host# show

[edit shared sae configuration aggregate-services]
user@host#
```

Remove the configuration for a specific property (routing-options):

```
[edit]
user@host# set routing-options static route 192.0.2.20/24 reject

[edit]
user@host# show
. . .
static {
  route 192.0.2.20/24 {
    reject;
  }
}
. . .
[edit]
user@host# delete routing-options

[edit]
user@host# edit routing-options

[edit routing-options]
user@host# show
```

Renaming an Identifier

To modify a configuration, you can rename an identifier that already exists. You can do this either by deleting the identifier (using the delete command) and then adding the renamed identifier (using the set and edit commands), or you can rename the identifier using the rename mode command:

user@host# **rename** <*statement-path*> *identifier1* **to** *identifier2*

Example: Renaming an Identifier

Change the Network Time Protocol (NTP) server address to 10.0.0.6:

```
[edit]
user@host# rename system ntp server 10.0.0.7 to server 10.0.0.6
```

Inserting a New Identifier

You can enter most statements and identifiers in any order. Regardless of the order in which you enter the configuration statements, the CLI always displays the configuration in a strict order. However, in a few cases the ordering of the statements matters because the configuration statements create a sequence that is analyzed in order.

For example, rules for interface, subscriber, and DHCP classification scripts are evaluated in the order in which they appear in the configuration. If you add a rule that you want to be evaluated before an existing rule, you need to modify the ordering of the rules,.To modify a portion of the configuration in which the statement order matters:

- Use the insert configuration mode command:

  user@host# **insert** *<statement-path> identifier1* **(before | after)** *identifier2*

If you do not use the insert command, but instead simply configure the identifier, it is placed at the end of the list of similar identifiers.

You use the insert command to reorder identifiers that you have already configured.

Examples: Inserting a New Identifier

Add a new subscriber classification rule and insert it before existing rules at the [edit shared sae user-classifier] hierarchy level:

```
[edit shared sae user-classifier]
user@host> show
rule rule-2 {
  target <-retailerDn->??sub?(uniqueID=<-userName->);
  condition {
    loginType == "SYNC";
  }
}
rule rule-3 {
  target <-unauthenticatedUserDn->;
  condition {
    loginType == "TOKEN";
    loginType == "PUBLIC";
  }
}
rule rule-4 {
  target <-retailerDn->??sub?(uniqueID=<-userName->);
  condition {
    retailerDn != "";
    & userName != "";
  }
}

[edit shared sae user-classifier]
user@host# set rule new target "[<-unauthenticatedUserDn->]"

[edit shared sae user-classifier]
user@host# set rule new condition "loginType=="AuthADDR""

[edit shared sae user-classifier]
user@host# insert rule new before rule-2

[edit shared sae user-classifier]
user@host# show
rule new {
  target "[<-unauthenticatedUserDn->]";
  condition {
    loginType==AuthADDR;
  }
}
```

```
rule rule-2 {
  target <-retailerDn->??sub?(uniqueID=<-userName->);
  condition {
    loginType == "SYNC";
  }
}
rule rule-3 {
  target <-unauthenticatedUserDn->;
  condition {
    loginType == "TOKEN";
    loginType == "PUBLIC";
  }
}
rule rule-4 {
  target <-retailerDn->??sub?(uniqueID=<-userName->);
  condition {
    retailerDn != "";
    & userName != "";
  }
}
```

Verifying a Configuration

To verify that the syntax of a configuration is correct:

- Use the configuration mode commit check command.

```
[edit]
user@host# commit check
configuration check succeeds
[edit]
user@host#
```

If there is an error in the configuration syntax, the commit check command returns a message that indicates the location of the error. For example:

```
[edit]
user@host# commit check
[edit shared sae configuration driver]
  missing mandatory attribute(s): 'unauthenticated-subscriber-dn',
'mac-cache-expiration'
```

Committing a Configuration

To save software configuration changes to the directory and activate the configuration:

- Use the commit configuration mode command.

```
[edit]
user@host# commit
commit complete
[edit]
user@host#
```

When you enter the commit command, the software reviews the configuration for syntax errors (commit check). Then, if the syntax is correct, the configuration is activated and becomes the active configuration.

You can issue the commit command from any hierarchy level.

If the configuration contains syntax errors, a message indicates the location of the error, and the configuration is not activated. The error message has the following format:

```
[edit edit-path]
     'offending-statement;'
       error-message
```

For example:

```
user@host# commit
[edit system login user Chris Bee class]
Undefined class 'newClass'
```

We recommend that you correct the error before recommitting the configuration. To return quickly to the hierarchy level where the error is located, copy the path from the last line of the message and paste it at the configuration mode prompt at the [edit] hierarchy level.

When you commit a configuration, you commit the entire configuration in its current form. If more than one user is modifying the configuration, committing it saves and activates the changes of all the users.

Committing a Configuration and Exiting Configuration Mode

To save software configuration changes, activate the configuration, and exit configuration mode. Use the commit and-quit configuration mode command. This command succeeds only if the configuration contains no errors.

```
[edit]
user@host# commit and-quit
commit complete
exiting configuration mode
user@host>
```

When Multiple Users Configure the Software

A number of users can be working in configuration mode simultaneously, and they all can be making changes to the configuration. All changes made by all users are visible to everyone editing the configuration—the changes become visible as soon as the user presses the Enter key at the end of a command that changes the configuration, such as set, edit, or delete.

When any of the users editing the configuration issues a commit command, all changes made by all users are checked and activated.

If, when you enter configuration mode, the configuration contains changes that have not been committed, a message appears:

```
user@host> configure
Entering configuration mode
The configuration has been changed but not committed
[edit]
user@host>
```