Chapter 1

Managing Services with the SRC CLI

This chapter describes how to manage services with the SRC CLI.

You can also use SRC configuration applications to configure the SRC software on a Solaris platform. See *Chapter 2, Managing Services on a Solaris Platform*.

Topics in this chapter include:

Overview of Services

The SRC software supports four types of services:

- Normal—Policy-based service.
- Aggregate—Group of services, handled as a unit.
- Infrastructure—Service that can be provisioned only once and then activated a number of times for one or more subscribers across network devices.
- Script—Custom service into which you can insert or reference a script that provisions policies on a number of systems across a network, including networks that contain devices that do not have supported device drivers.

Use aggregate and infrastructure services together to apply policies across JUNOSe routers and JUNOS routing platforms, and other systems that have a supported device driver.

Use script services to create customized service implementations, such as a service to configure firewall policies on a device that does not have a supported device driver—for example, a Juniper Networks NetScreen-5GT appliance.

Automatic Service Activation

You can configure a permanent service—a service that the SAE automatically activates when it starts a subscriber session for subscribers who use that service. A typical application of this feature is to automatically activate a particular video service for all subscribers associated with a particular retailer. You can allow subscribers to deactivate the service, or prohibit them from deactivating it, after the SAE has automatically activated it. To make a service permanent, set the permanent option in the service configuration.

Enabling the Service Configuration on the SRC CLI

Before you can configure services with the SRC CLI, you must enable the policy, service, and subscriber editor on the SRC CLI. To do so:

In operational mode, enter the following command:

user@host> **enable component policy-service-subscriber**

Before You Configure Services

Before you configure services:

- Plan the services that you want to make available to subscribers.
- Configure the policies for a service to use. For information about configuring policies, see *Defining Policies to Manage Traffic* on page 139.

Adding a Normal Service

Use the following configuration statements to add normal services to the global service scope:

```
services global service name {
    description description;
    type (normal);
    category category;
    url url;
    policy-group policy-group;
    authentication-required;
    authorization-plug-in [authorization-plug-in...];
    tracking-plug-in [tracking-plug-in...];
    session-timeout session-timeout;
    idle-timeout idle-timeout;
    accounting-interim-interval accounting-interim-interval;
    radius-class radius-class;
    status (inactive | active);
    activate-only;
    permanent;
    available;
    secret;
    shared-service-name shared-service-name;
}
```

Use the following configuration statements to add normal services to a service scope:

```
services scope name service name {
    description description;
    type (normal | aggregate | script | infrastructure);
    category category;
    url url;
    policy-group policy-group;
    authentication-required;
    authorization-plug-in [authorization-plug-in...];
    tracking-plug-in [tracking-plug-in...];
    session-timeout session-timeout;
    idle-timeout idle-timeout;
    accounting-interim-interval accounting-interim-interval;
    radius-class radius-class;
    status (inactive | active);
    activate-only;
    permanent;
    available;
    secret;
    shared-service-name shared-service-name;
}
```

To add a normal service:

1. From configuration mode, enter the service configuration. In this sample procedure, the service is configured in the global service scope, and Video-Gold is the name of the service.

   user@host# **edit services global service Video-Gold**

2. (Optional) Enter a description for the service.

   [edit services global service Video-Gold]
   user@host# **set description** *description*

3. Configure the type of service.

   [edit services global service Video-Gold]
   user@host# **set type normal**

4. (Optional) Configure the category of the service for other SRC applications, such as portals.

   [edit services global service Video-Gold]
   user@host# **set category** *category*

5. (Optional) Configure the link used in SRC applications, such as portals.

   [edit services global service Video-Gold]
   user@host# **set url** *url*

6. (Optional) Configure the policy group that is applied when the service is activated.

   [edit services global service Video-Gold]
   user@host# **set policy-group** *policy-group*

7. (Optional) Enable authentication required for services activated by portals.

   [edit services global service Video-Gold]
   user@host# **set authentication-required**

8. (Optional) Configure the plug-in(s) used to authorize the service.

   [edit services global service Video-Gold]
   user@host# **set authorization-plug-in** [*authorization-plug-in...*];

9. (Optional) Configure the plug-in(s) used to collect accounting data for the service.

   [edit services global service Video-Gold]
   user@host# **set tracking-plug-in** [*tracking-plug-in...*]

10. (Optional) Configure the time after which the service session is deactivated.

    [edit services global service Video-Gold]
    user@host# **set session-timeout** *session-timeout*

11. (Optional) Configure the idle time after which the SAE deactivates service.

    [edit services global service Video-Gold]
    user@host# **set idle-timeout** *idle-timeout*

12. (Optional) Configure the time between interim accounting messages.

    [edit services global service Video-Gold]
    user@host# **set accounting-interim-interval** *accounting-interim-interval*

13. (Optional) Configure the value of the RADIUS class attribute in accounting messages.

    [edit services global service Video-Gold]
    user@host# **set radius-class** *radius-class*

14. (Optional) Configure the status of the service.

    [edit services global service Video-Gold]
    user@host# **set status** (inactive | active)

15. (Optional) Configure whether the SAE can deactivate this service

    [edit services global service Video-Gold]
    user@host# **set activate-only**

16. (Optional) Enable automatic activation when the service is subscribed.

    [edit services global service Video-Gold]
    user@host# **set permanent**

17. (Optional) Specify whether a subscriber can activate a service.

    [edit services global service Video-Gold]
    user@host# **set available**

18. (Optional) Specify whether the service is visible only to administrators who have permission to see secret information.

    [edit services global service Video-Gold]
    user@host# **set secret**

19. (Optional) Verify your configuration.

```
[edit services global service Video-Gold]
user@host# show
description "Example for content provider allowing high speed access";
type normal;
category Video;
url http://video.server.com;
policy-group /sample/common/content-provider-tiered;
radius-class Video-Gold;
status active;
}
```

Setting Parameter Values for Services

Using parameters, you can define general settings in one SRC object and provide specific values for that setting in another object. For example, you can define the general settings for a rate limiter in a policy, insert a parameter for a rate in the policy, and provide specific values for the rate in each service that uses this policy. For information about the concept of parameters, see *Chapter 15, Defining and Acquiring Values for Parameters*.

Use the following configuration statements to configure parameters for services in the global service scope:

```
services global service name parameter {
    gateway-ip-address gateway-ip-address;
    service-ip-address service-ip-address;
    service-ip-mask service-ip-mask;
    service-port service-port;
    substitution [substitution...];
    session-volume-quota session-volume-quota;
}
```

Use the following configuration statements to configure parameters for services in a service scope:

```
services scope name service name parameter {
    gateway-ip-address gateway-ip-address;
    service-ip-address service-ip-address;
    service-ip-mask service-ip-mask;
    service-port service-port;
    substitution [substitution...];
    session-volume-quota session-volume-quota;
}
```

To configure parameters for services:

1. From configuration mode, enter the service parameter configuration. In this sample procedure, the service called Video-Gold is configured in the global service scope.

   user@host# **edit services global service Video-Gold parameter**

2. (Optional) Configure the actual IP address of the gateway router. This value is substituted for the policy global parameter called gateway_ipAddress.

   [edit services global service Video-Gold parameter]
   user@host# **set gateway-ip-address** *gateway-ip-address*

3. (Optional) Configure the actual IP address of the host(s) that provides the service. This value is substituted for the policy global parameter called service_ipAddress.

   [edit services global service Video-Gold parameter]
   user@host# **set service-ip-address** *service-ip-address*

4. (Optional) Configure the actual IP mask for the service. This value is substituted for the policy global parameter called service_ipMask.

   [edit services global service Video-Gold parameter]
   user@host# **set service-ip-mask** *service-ip-mask*

5. (Optional) Configure the actual port for the service. This value is substituted for the policy global parameter called service_port.

   [edit services global service Video-Gold parameter]
   user@host# **set service-port** *service-port*

6. (Optional) Configure actual values for other parameters.

   [edit services global service Video-Gold parameter]
   user@host# **set substitution** [*substitution...*]

7. (Optional) Configure the quota for the volume of data for service sessions. The SRC software uses this value as the default for service sessions created for this service.

   [edit services global service Video-Gold parameter]
   user@host# **set session-volume-quota** *session-volume-quota*

8. (Optional) Verify your configuration.

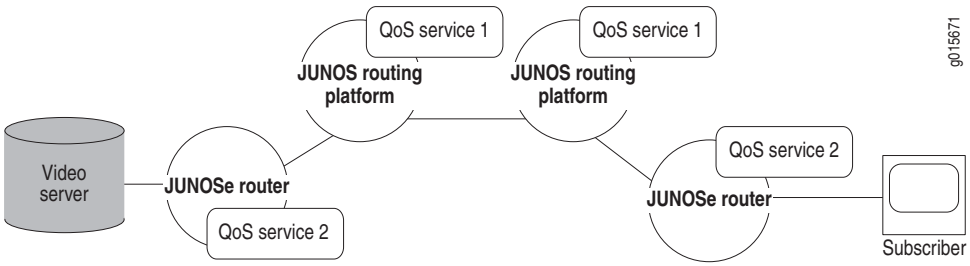
```
[edit services global service Video-Gold parameter]
user@host# show
service-ip-address 10.10.40.16;
service-ip-mask 255.255.255.240;
substitution "bw = 5000000";
```

Aggregating Services

An aggregate service comprises a number of individual services. Combining services lets the SRC software treat the services within an aggregate service as a unit. When an aggregate service becomes active, it tries to activate all the services within it.

An aggregate service can distribute the activation of a number of services within the aggregate across one or more SAEs in an SRC network. This specialized service is ideal for supporting voice over IP (VoIP) and video on demand. To deliver these types of features to subscribers, you can configure bidirectional or unidirectional quality of service (QoS) services based on policies provisioned across a number of interfaces on one or more SAE-managed network devices in an SRC network. Figure 1 shows a sample aggregate service that provides end-to-end QoS for video on demand, with QoS service 1 and QoS service 2 activated on Juniper Networks routers in the path between the video server and the subscriber.

Figure 1: Sample Configuration of an Aggregate Service



The services included in an aggregate service manage policies in the usual manner. The aggregate service does not directly manage any policies on a network device.

Fragment Services

The services that make up an aggregate service are referred to as fragment services. This term provides a way to distinguish between services that are included in an aggregate service and those that are not. The fragment services can be any type of service that the SAE supports, except another aggregate service.

Subscriber Reference Expressions for Fragment Services

The configuration for each fragment service includes a subscriber reference expression, a phrase that identifies the subscriber sessions that activate the fragment service. The subscriber reference expression defines the subscriber session by subscriber IP address, distinguished name (DN), interface name, login name, or associated virtual router.

To use aggregate services requires that the network information collector (NIC) be configured. Use a configuration scenario that provides a key for the type of subscriber reference expression defined for the fragment service. For example, if the subscriber reference expression is a DN, the NIC key is also a DN. In this case, you could use the NIC configuration scenario OnePopDnSharedIp, which uses a DN as a key.

For more information about the NIC configuration scenarios and the types of resolutions performed by these scenarios, see *SRC-PE Network Guide, Chapter 21, NIC Configuration Scenarios*.

Mandatory Services

A fragment service that must be active for an aggregate service to become active is called a mandatory service. When you configure an aggregate service, you specify which services, if any, are mandatory. For example, you could specify that rate-limiting services for a video-on-demand connection be mandatory to ensure call quality.

Redundant Services

When you configure an aggregate service, you can configure fragment services to provide redundancy for each other. Fragment services that share the same redundancy group name provide redundancy.

For an aggregate service to become active, at least one fragment service from each redundancy group must become active. For example, if you configure two services, S1 and S2, and assign the same redundancy group name to each of these services, S1 and S2 provide redundancy for each other if one becomes disabled.

While an aggregate service is active, the SAE tries to keep all fragment services within it active. An aggregate service and any of its active fragment services become inactive if a mandatory fragment service or an entire redundancy group becomes inactive.

Aggregate Service Sessions

An aggregate service session coordinates the activation of the services within it. It runs on the same SAE where it starts. The aggregate service session is created in the router driver that hosts the subscriber session that starts the service. An individual service session for a fragment service can be activated in the same SAE or another SAE on the SRC network.

Understanding how aggregate service sessions are managed can help you troubleshoot service activation or service deactivation issues that might arise. The SRC software provides a set of configurable timers that helps control session management.

For information about the timers that you can use to troubleshoot aggregate services, see *Configuring Timers for Aggregate Services* on page 18.

Session Activation

An aggregate service becomes active when:

- All mandatory services are active.

  If a mandatory service does not start, the SAE deactivates any fragment services that are active.

- If there are no mandatory services, at least one service is active.

If any fragment services that are not mandatory services do not become active, the aggregate service continues to try to start them. How long the aggregate service tries to activate fragment services depends on the settings for activation-deactivation time.

When an aggregate service becomes active, it monitors the services that are part of the aggregate service.

NOTE: Depending on your implementation, accounting software could detect that a fragment service session became active even though the associated aggregate service did not become active, resulting in the fragment services being deactivated.

You can configure your accounting software to ignore the activation of the fragment session when an aggregate service session fails. This way, a customer is not billed for an aggregate service that was not received.

Session Deactivation

When the SAE deactivates an aggregate service, the aggregate service session tries to deactivate the services within it. The SAE deactivates an aggregate service when all fragment services stop. If one of these services remains active, the aggregate service stays in memory until the service session ends. The SAE periodically tries to stop the active fragment session until the maximum retry time is reached, at which time it deactivates the aggregate service. As a result, the aggregate service session can remain in memory after the associated subscriber session ends.

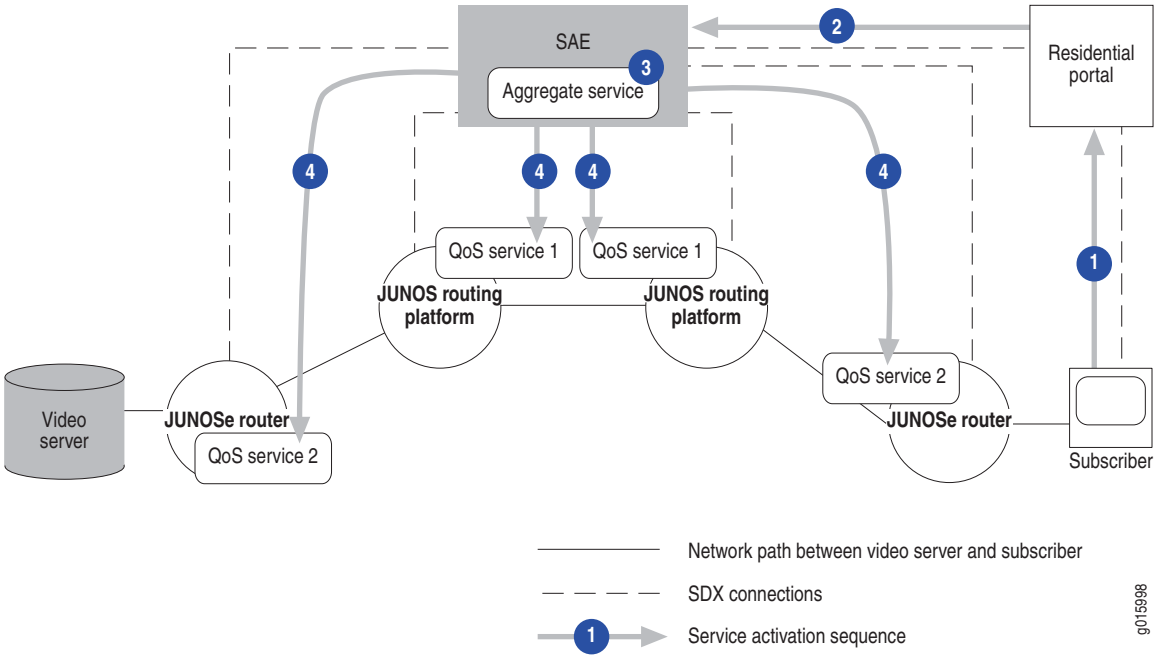
Session Monitoring

An aggregate service session exchanges keepalive messages with a session management process for remote fragment services. This way, if a service session is removed from a router while the SAE is not managing the router, such as when the Common Open Policy Service (COPS) client stops on a JUNOSe router or the configuration database is reset on a JUNOS routing platform, the SAE associated with the router receives notification that the keepalive message failed.

Service Activation

Aggregate services are activated in a way similar to any other service, but with the additional requirement of activating the associated fragment services. Figure 2 shows a sample service activation for a video-on-demand service.

Figure 2: Aggregate Service Activation



The following process describes the service activation for a video-on-demand service, with Steps 1–4 illustrated in Figure 2.

1. A subscriber requests a video-on-demand service through a residential portal.
2. The residential portal requests the service through the SAE.
3. The SAE activates a subscription for the associated aggregate service, and a session for the aggregate service becomes active.
4. The aggregate service coordinates with the SAE, and the SAE tries to activate the fragment services that have been configured for the aggregate service.

The aggregate service becomes active when:

- All mandatory services are active.
- If there are no mandatory services, at least one fragment service is active.
- For redundant fragment services, at least one fragment service configured for a redundancy group becomes active.

The aggregate service initiates accounting, if accounting has been configured.

After the aggregate service becomes active, it monitors fragment services to ensure that they are still active. When the subscriber or the video server ends the video-on-demand session, the aggregate service tries to terminate active fragment services.

For detailed information about service activation, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 3, Subscriber Logins and Service Activation*.

Before You Configure an Aggregate Service

Before you configure an aggregate service:

1. Plan the aggregate service:
   - Plan which fragment services will constitute the aggregate service.
   - Plan the routers on which the fragment services are to be activated.
2. Configure the fragment services.

   See *Adding a Normal Service* on page 5.
3. If the aggregate service includes services to be activated remotely, ensure that one or more NIC proxies are configured on each SAE.
4. Ensure that the NIC is configured to use a scenario that provides the appropriate type of key.

   See *Subscriber Reference Expressions for Fragment Services* on page 10.
5. Ensure that the SAEs can communicate with each other and the NIC host(s). Make sure that firewalls permit TCP and CORBA communication between the systems hosting the SAEs, and communication between the NIC host(s) and the SAE.

   See *SRC-PE Getting Started Guide, Chapter 16, Setting Up an SAE with the SRC CLI*.
6. Ensure that the communication between SAEs is secure.

   Follow the standards for your organization to ensure that communication between SAEs is protected.
7. If the aggregate service is to include a fragment service on a remote SAE, ensure that the remote fragment service can become active by verifying that the fragment service is loaded on the remote SAE.

How Parameters Are Passed From Aggregate Service to Fragment Service

There are two ways to set up parameters in aggregate and fragment services:

- If you use just a parameter name in the aggregate service, for example user_IpAddress, then the value of user_IpAddress in the aggregate session is bound to the name user_IpAddress in the fragment service.

- If you use user_IpAddress as the parameter name and fragSubrIp = user_IpAddress as a substitution in the aggregate service, user_IpAddress is given a different name in the fragment service session. The parameter name fragSubrIps in the fragment service session is bound to the value of user_IpAddress in the aggregate service session.

  Use this scheme to configure parameters and substitutions when the parameter in the aggregate service session has a name that is already used in the fragment for something else. A common example is user_IpAddress, which is usually defined in all service sessions. This scheme is also useful when you are aggregating services developed independently. You can call the aggregate service parameters whatever makes sense in that context, and name the fragment service parameters independently.

Configuring Service Fragments for an Aggregate Service

Use the following configuration statements to configure an aggregate service in the global service scope:

```
services global service name aggregate fragment name {
    expression expression;
    service service;
    mandatory;
    redundancy-group redundancy-group;
    subscription-required;
    substitution [substitution...];
}
```

Use the following configuration statements to configure an aggregate service in a service scope:

```
services scope name service name aggregate fragment name {
    expression expression;
    service service;
    mandatory;
    redundancy-group redundancy-group;
    subscription-required;
    substitution [substitution...];
}
```

To configure an aggregate service:

1. From configuration mode, enter the service aggregate configuration. In this sample procedure, the service called MirrorAggregate is configured in the scope configuration.

   user@host# **edit services scope TM service MirrorAggregate aggregate fragment 0**

2. Configure the subscriber reference expression that identifies the remote subscriber session that will host the fragment.

   [edit services scope TM service MirrorAggregate aggregate fragment 0]
   user@host# **set expression** *expression*

3. Configure the name of the service to be included in the aggregate service as a fragment service.

   [edit services scope TM service MirrorAggregate aggregate fragment 0]
   user@host# **set service** *service*

4. (Optional) Specify whether the fragment service must be active for the aggregate service to become active.

   [edit services scope TM service MirrorAggregate aggregate fragment 0]
   user@host# **set mandatory**

5. (Optional) Configure the group name to be applied to each fragment service that is to be part of a redundancy group.

   [edit services scope TM service MirrorAggregate aggregate fragment 0]
   user@host# **set redundancy-group** *redundancy-group*

6. (Optional) Specify whether a remote subscriber session is required to subscribe to the fragment service.

   [edit services scope TM service MirrorAggregate aggregate fragment 0]
   user@host# **set subscription-required**

7. (Optional) Configure the list of substitutions that are used as arguments for the fragment to become active.

   [edit services scope TM service MirrorAggregate aggregate fragment 0]
   user@host# **set substitution** [*substitution...*]

8. (Optional) Verify your configuration.

```
[edit services scope TM service MirrorAggregate aggregate fragment 0]
user@host# show
expression
"vr=\"<- substitution.vrNames ->\", interfaceName=\"FORWARDING_INTERFACE\"";
service MirrorFragment;
substitution fragSubrIps=subrIps;
```

Using Python Expressions in a Subscriber Reference Expression

You can compose Python expressions from one or more of the fields in Table 4 for the definition of a subscriber reference expression of a fragment service. You enter these expressions with the **expression** option of the **services scope** *name* **service** *name* **aggregate fragment** or **edit services global service** *name* **aggregate fragment** statement.

Table 4: Fields Used in Python Expressions for Aggregate Services

Field	Description
substitution.<xyz>	Value of the parameter <xyz>. Substitutions are acquired by means of the regular acquisition path for service sessions. The names of parameters are restricted to valid Python identifiers, such as 'ALPHA/"_" *(ALPHA/ DIGIT/"_")', with the exception of keywords, such as **for**, **if**, **while**, **return**, **and**, **or**, **not**, **def**, **class**, **try**, **except**. For the full list of Python keywords, see http://docs.python.org/ref/keywords.html.
loginType	The type of subscriber session, one of the following: ■ ASSIGNEDIP—An assigned IP login is triggered when an application accesses a subscriber object for an assigned IP subscriber that is not currently loaded into memory. (JUNOSe routers) ■ AUTHINTF—An authenticated interface login is triggered when an interface responds to authentication, such as authentication for a PPP session. (JUNOSe routers) ■ INTF—An interface login is triggered when an interface comes up and the interface classifier script determines that the SAE should manage that interface, unless the interface comes up as a result of an authenticated PPP session. (JUNOS routing platforms and JUNOSe routers) ■ ADDR—An address login is triggered when the DHCP server in the JUNOSe router provides a token IP address. (JUNOSe routers) ■ AUTHADDR—An authenticated address login is triggered when the DHCP server in the JUNOSe router provides a public IP address. (JUNOSe routers) ■ PORTAL—A portal login is triggered when the portal API is invoked by a JSP Web page to log in a subscriber. (JUNOS routing platforms and JUNOSe routers)
loginName	Login name provided by a subscriber
userName	Username portion of the loginName
domainName	Domain name portion of the loginName
serviceBundle	Content of the vendor-specific RADIUS attribute for service bundle
radiusClass	RADIUS class used for authorization
virtualRouterName	Name of virtual router in the format vrname@hostname
interfaceName	Name of the interface
ifAlias	Description of the interface configured on the router
ifDesc	Alternate name for the interface. This is the name used by the Simple Network Management Protocol (SNMP). On a JUNOSe router the format of the description is: ip<slot>/<port>.<subinterface> On a JUNOS routing platform, ifDesc is the same as interfaceName.

Table 4: Fields Used in Python Expressions for Aggregate Services (continued)

Field	Description
nasPortId	Port identifier of an interface, including the interface name and additional layer 2 information (for example, fastEthernet 3/1)
macAddress	Text representation of the MAC address for the DHCP subscriber (for example. 00:11:22:33:44:55)
retailerDn	Distinguished name of the retailer
nasIp	Network access server IP address of the router
dhcp	DHCP options. See *SRC-PE Subscribers and Subscriptions Guide, Chapter 6, Classifying Interfaces and Subscribers with the SRC CLI*.
primaryUserName	PPP or DHCP username. This name does not change when the subscriber logs in through a portal.

Configuration Examples for Aggregate Services

For configuration examples for aggregate services see the following chapters:

- *SRC Application Library Guide, Chapter 5, Mirroring Subscriber Traffic in the SRC Network*
- *SRC Application Library Guide, Chapter 9, Configuring Services and Subscriptions to Integrate IDP*

Configuring Timers for Aggregate Services

You can change the values for several timers to specify the intervals associated with monitoring and activating aggregate sessions. Use the following configuration statements to configure these timers and intervals:

```
shared sae configuration aggregate-services {
    keepalive-time keepalive-time;
    keepalive-retry-time keepalive-retry-time;
    activation-deactivation-time activation-deactivation-time;
    failed-notification-retry-time failed-notification-retry-time;
}
```

To configure timers used by aggregate services:

1. From configuration mode, enter the shared sae aggregate service configuration.

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

2. Configure the interval at which keepalive messages are sent between an aggregate service session and an associated remote service management session to verify that an aggregate service is active.

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

3. Configure the time to wait for an acknowledgement of a keepalive message before sending a new keepalive message if a response to a keepalive message is not received.

   [edit shared sae configuration aggregate-services]
   user@host# **set keepalive-retry-time** *keepalive-retry-time*

4. Configure the length of time to continue to try to activate or deactivate a fragment service session.

   [edit shared sae configuration aggregate-services]
   user@host# **set activation-deactivation-time** *activation-deactivation-time*

5. Configure the length of time to continue sending failure notifications if an aggregate service cannot reach a fragment service, or a fragment service cannot reach an aggregate service during shutdown of the aggregate service.

   [edit shared sae configuration aggregate-services]
   user@host# **set failed-notification-retry-time** *failed-notification-retry-time*

6. (Optional) Verify your configuration.

```
[edit shared sae configuration aggregate-services]
user@host# show
keepalive-time 150000;
keepalive-retry-time 900;
activation-deactivation-time 900;
failed-notification-retry-time 9200;
```

Sharing Service Provisioning

You can use infrastructure services to provision a service to be shared by a number of subscriber sessions. Infrastructure services are services that can be activated a number of times for one or more subscribers, but provisioned only once. Infrastructure services are designed to be shared among instances of aggregate services.

When an infrastructure service is activated, the SAE activates the service if a shared service session for the service is not already active; otherwise, it increments the usage counter for the service. When an infrastructure service is deactivated, the SAE decrements the usage counter for the shared session. When the last service session is deactivated, the shared session is also deactivated.

Although an infrastructure service is designed for use as a fragment service in an aggregate service, it can be used independently. As a fragment service, it can be bundled with other fragment services to deliver a service package in the aggregate service.

Adding an Infrastructure Service

To add an infrastructure service:

1. Add the service to be shared, as described in *Adding a Normal Service* on page 5.

2. Set the service type to infrastructure.

   [edit services global service Infrastructure]
   user@host# **set type** infrastructure

3. Configure the name of the service to be shared.

   [edit services global service Infrastructure]
   user@host# **set shared-service-name** *shared-service-name*

4. (Optional) Verify your configuration.

```
[edit services global service Infrastructure]
user@host# show
type infrastructure;
radius-class infrastructure;
status active;
shared-service-name Video-Bronze;
```

Extending Service Implementations with Script Services

You use services to provision policies on a number of systems across a network, including networks that do not contain a JUNOSe router or JUNOS routing platform. Script services provide an interface to call scripts that supply custom services. You can use script services to create custom service implementations, such as:

- Provisioning layer 2 devices, such as digital subscriber line access multiplexers (DSLAMs).
- Setting up network connections, such as MPLS tunnels.
- Provisioning policies for network devices that do not have a supported SAE device driver.
- Accessing functionality not currently supported by SAE device drivers but supported by other interfaces, such as RADIUS change of authorization (CoA) on JUNOSe routers and JunoScript provisioning on JUNOS routing platforms.

Perform the following to customize service implementations:

1. Writing Scripts for Script Services on page 21
2. Configuring Script Services on page 24

Writing Scripts for Script Services

The ScriptService service provider interface (SPI) provides a Java interface that a script service implements. For information about the ScriptService interface and the ServiceSessionInfo interface, see the script service documentation in the SRC software distribution in the folder *SDK/doc/sae* or in the SAE core API documentation on the Juniper Networks Web site at

http://www.juniper.net/techpubs/software/management/sdx/api-index.html

The implementation of the ScriptService interface activates the service. The SAE sends authentication and tracking events when it activates, modifies, or deactivates a script service session.

The SAE supports script services written in Java or Jython. For scripts written in Java, you must compile and package the implemented ScriptService to make it available for use by the SAE. A Java implementation can include more than one Java archive (JAR) file.

The SAE synchronizes methods used by the same instance of the ScriptService class. You do not need to provide synchronized implementation of the methods.

NOTE: The script service implementation can be called by different threads at the same time. If your script uses resources that are shared between different service instances, you are responsible for synchronizing access to those resources.

To write a script to be used by a script service:

1. Create a class that provides a default constructor and that implements the ScriptService interface.

2. Manage activation and manipulation of the service session by implementing the following ScriptService methods:

   - activateSession()—Activates the script service session.

   - deactivateSession()—Deactivates the script service session and returns any final accounting data for the script service session.

   - modifySession()—If the counters were reset during the modification, modifies the script service session and returns any accounting data.

   These methods are implemented by the script service. They perform the associated action (activate, deactivate, modify) when the SAE calls the method.

3. (Optional) Get information about service sessions by using methods on the ServiceSessionInfo interface.

4. (Optional) Provide accounting data, if used, by using the following ScriptService method:

   getAccountingData()—Polls for current accounting data and returns any current accounting data.

5. (Optional) Provide service status information by using the following ScriptService method:

   getState()—Returns session data to be stored persistently on the router. The SAE does not use this data but provides it to the script when a service session is restored after failover.

6. Manage the script service by using the following ScriptService methods:

   - initState() —Initializes a recovered script service session after a state synchronization.

   - discarded()—Provides notification that the service session has been discarded. Service sessions are discarded when the SAE loses connection to a router. A discarded service session continues to exist on the router and is restored after the connection to the router is reestablished by an SAE.

     The script service session releases any resources associated with a discarded session, but must not take any action to disrupt the service session.

   You can also use the stopService() method on the ServiceSessionInfo object to stop a service and remove the service from the SAE. For example, consider a script service that monitors a state that it creates outside the SAE. If the script detects that the service is not active, it can stop the service and remove it from the SAE. You could use this type of script service to start a daemon process and monitor the process to make sure that it is alive.

NOTE: The ScriptService SPI does not provide access to a router driver.

Example: Using the ScriptService SPI in Jython

The following example implements the ScriptService SPI in Jython.

```
class SampleService(ScriptService):
    def initSessionInfo(self, ssi):
        self.ssi = ssi

    def activateSession(self):
        print “Activating ServiceName %s” % ssi.serviceName

    def deactivateSession(self):
        print “Deactivating ServiceName %s” % ssi.serviceName
        return None

    def modifySession(self, ssi):
        self.ssi = ssi
        print “Modifying ServiceName %s” % ssi.serviceName
        return None

    def getAccountingData(self):
        print “Getting accounting data for ServiceName %s” % ssi.serviceName
        return None

    def getState(self):
        return None
```

```
    def initState(self, ssi, state):
        self.ssi = ssi
        pass

    def discarded(self):
        pass
```

Example: Using the ScriptService SPI in Java

The following example implements the ScriptService SPI in Java.

```
class SampleService implements ScriptService {
    private ServiceSessionInfo ssi;
    public SampleService() { }
    public void initSessionInfo(ServiceSessionInfo ssi) {
        this.ssi = ssi;
    }

    public void activateSession() {
        System.out.println("Activating ServiceName "+ssi.getServiceName());
    }

    public AccountingData deactivateSession() {
        System.out.println("Deactivating ServiceName
"+ssi.getServiceName());
        return null;
    }

    public AccountingData modifySessionSession(ServiceSessionInfo ssi) {
        this.ssi = ssi;
        System.out.println("Modifying ServiceName "+ssi.getServiceName());
        return null;
    }

    public AccountingData getAccountingData() {
        System.out.println("Getting accounting data for ServiceName
"+ssi.getServiceName());
        return null;
    }

    public byte[] getState() {
        return null;
    }

    public initState(ServiceSessionInfo ssi, byte[] state) {
        this.ssi = ssi;
    }

    public void discarded() {
    }
}
```

Configuring Script Services

Before you configure a script service, make sure that you know the location of the script file that the service will reference.

Use the following configuration statements to configure a script service in the global service scope:

```
services global service name script {
    script-type (url | python | java-class | java-archive);
    class-name class-name;
    file file;
}
```

Use the following configuration statements to configure a script service in a service scope:

```
services scope name service name script {
    script-type (url | python | java-class | java-archive);
    class-name class-name;
    file file;
}
```

To configure a script service:

1. Configure a normal service, but set the service type to **script**. See *Adding a Normal Service* on page 5.

2. From configuration mode, enter the service script configuration. In this sample procedure, the service called scriptService is configured in the scope configuration.

   ```
   user@host# edit services scope script service scriptService script
   ```

3. Configure the type of script that the script service uses.

   ```
   [edit services scope script service scriptService script]
   user@host# set script-type (url | python | java-class | java-archive)
   ```

4. For Java class and Python script services, configure the name of the class that implements the script service.

   ```
   [edit services scope script service scriptService script]
   user@host# set class-name class-name
   ```

5. Configure the URL of the script service or the path and filename of the service.

   ```
   [edit services scope script service scriptService script]
   user@host# set file file
   ```

6. (Optional) Verify your configuration.

   ```
   [edit services scope script service scriptService script]
   user@host# show
   script-type java-class;
   class-name net.juniper.smgt.script.service;
   file://opt/UMC/sae/var/scriptservice/script-service.jar;
   ```

Restricting Simultaneous Activation of Services

A mutex group defines a set of services that are mutually exclusive—services that the SAE cannot simultaneously activate for a particular subscriber. You can assign a service to more than one mutex group. When a subscriber requests activation of a particular service, the SAE determines which mutex groups contain that service. If the subscriber has current activations of other services listed in those mutex groups, the SAE proceeds in one of the following ways, depending on how you configured the mutex groups:

- Deactivates the other services listed in the mutex groups, and then activates the requested service.
- Refuses access to the requested service.

If the requested service is not listed in a mutex group, the SAE can activate the service regardless of any other services that the subscriber is using.

Restricting Simultaneous Activation of Persistent or Automatic Services

The SAE uses the following method to prevent simultaneous activation of mutually exclusive services that are configured for persistent activation or that are activated automatically when a subscriber logs in:

1. If you (or a subscriber) persistently activate an existing service or change a subscription to activate an existing service when a subscriber logs in, the SAE determines whether the service is specified in one or more mutex groups.
2. The SAE determines how each mutex group that lists the service is configured, and the SRC software acts accordingly.
    - If all the mutex groups that list the service allow automatic deactivation of services, the SRC software removes the persistent activations for the service and changes activate-on-login subscriptions to manual.
    - If any of the mutex groups does not allow automatic deactivation of services, the SRC software will not allow you to:
        - Persistently activate the service.
        - Change the subscription to activate the service when a subscriber logs in.

Adding a Mutex Group

Use the following configuration statements to configure a mutex group in the global service scope:

```
services global mutex-group name {
    auto-deactivate (yes | no);
    description description;
    services [services...];
}
```

Use the following configuration statements to configure a mutex group in a service scope:

```
services scope name mutex-group name {
    auto-deactivate (yes | no);
    description description;
    services [services...];
}
```

To add a mutex group:

1. From configuration mode, enter the mutex group configuration. In this sample procedure, the mutex group is called Video.

   user@host# **edit services global mutex-group Video**

2. Configure the method that the SAE uses to manage activation of services defined in this group.

   [edit services global mutex-group Video]
   user@host# **set auto-deactivate** (yes | no)

3. Enter a description for the service.

   [edit services global mutex-group Video]
   user@host# **set description** *description*

4. Configure the lists of services that the mutex group contains.

   [edit services global mutex-group Video]
   user@host# **set services** [*services...*]

5. (Optional) Verify your configuration.

```
[edit services global mutex-group Video]
user@host# show
auto-deactivate yes;
description "Video Services providing access to the same site with different
quality";
services [ Video-Bronze Video-Gold Video-Silver ];
```

Restricting and Customizing Services for Subscribers

Service scopes let you customize which services are to be delivered to specific organizations or specific locales. You can use service scopes to provision services for a group of subscribers by specifying:

- Particular services or mutex groups.
- Parameter substitutions that customize generic services.

A service scope is a collection of services and mutex groups, and optionally defines parameter substitutions for its associated services. For more information about parameter substitutions, see *Chapter 15, Defining and Acquiring Values for Parameters*. The object *o = Services* is the generic service scope—a collection of services and mutex groups available to all subscribers.

You can assign service scopes to virtual routers (VRs) and to some types of subscribers.

Assigning Service Scopes to Multiple VRs and Subscribers

You can also assign a service scope to multiple VRs and subscribers. For example, by assigning a service scope to a group of VRs, you can specify that a service is available only in the locations served by those VRs. If a subscriber of this service accesses the network from a location where you do not offer this service, the portal will not display the service, and the subscriber will not be able to use it.

If you assign a service scope to multiple VRs and subscribers, you specify a precedence—a numerical ranking—for each service scope. The lower the precedence value, the higher the ranking of the service scope. By default, the object *o = Services* has the highest precedence value and the lowest ranking.

Defining Multiple Scopes for a Service

If multiple service scopes that define the same service are assigned to a VR or subscriber, the SAE selects the parameters to use for the service as follows:

1. It selects the parameters that are defined by only one service scope.
2. If the same parameter is defined by more than one service scope, the SAE selects the parameter as follows:
    a. Selects the parameter associated with the service scope that has the lowest precedence value.
    b. If the parameter is defined by multiple service scopes with the same precedence value, selects the parameter defined by the service scope with the lowest alphanumerical name.

For example, consider the situation shown in Table 5, in which three scopes define several parameters for the same service.

Table 5: Parameter Selection Example

Service Scope Name	Precedence Value	Parameter Definitions
s1	1	description, policy group
s2	5	description, URL
s3	5	description, URL

The SAE will use the following parameter definitions for the service:

- Description from scope s1 (s1 has the lowest precedence value)
- Policy group from scope s1 (only s1 defines this parameter)
- URL from scope s2 (s2 has a lower alphanumeric name than s3)

You can also configure a generic Internet access service, and use service scopes to define the access parameters for different locations to use this service. If multiple service scopes that define this Internet access service are assigned to a VR, the SAE uses the precedence values to determine how to customize the service.

Configuring Service Scopes

The tasks to configure a service scope are:

1. Adding Service Scopes on page 28
2. Assigning Services and Mutex Groups to Service Scopes on page 29
3. Assigning Service Scopes to VRs or Subscribers on page 29

Adding Service Scopes

Use the following configuration statement to configure service scopes:

```
services scope name {
    precedence precedence;
}
```

To add a service scope:

1. From configuration mode, enter the service scope configuration. In this sample procedure, the scope is called EntJunos.

   ```
   user@host# edit services scope EntJunos
   ```

2. Configure the precedence of the service scope.

   ```
   [edit services scope EntJunos]
   user@host# set precedence precedence
   ```

3. (Optional) Verify your configuration.

```
[edit services scope EntJunos]
user@host# show
precedence 2;
```

Assigning Services and Mutex Groups to Service Scopes

To assign services and Mutex Groups to a scope:

- Add the service or mutex group at the edit services scope hierarchy level.

  For example, to add a service to a service scope called video, enter the following:

  user@host# **edit services scope video service Video-Gold**

Assigning Service Scopes to VRs or Subscribers

You can assign multiple service scopes to a VR or subscriber, and you can assign a service scope to multiple VRs and subscribers.

To assign a service scope:

1. Enter the configuration for the object to which you want to add the service scope. For example:

   user@host# **edit shared network device erx-node1 virtual-router default**

2. Assign a scope to the object.

   [edit shared network device erx-node1 virtual-router default]
   user@host# **set scope** *scope*

Service Scope Configuration Examples

The following sections provide two practical examples for using scopes to customize your service configuration.

Example: Delivering a Limited Set of Services to Organizations

You can use service scopes to create a limited set of services to be made available to specified organizations. For enterprise users, you could define a set of services available on the JUNOS routing platform.

To deliver a small set of services to specified enterprises:

1. Create a scope for the services to be made available. For example, see the EntJunos scope in the sample data.

   user@host> **show configuration services scope EntJunos**

2. Add services to the scope, such as those in the sample data in the EntJunos scope.

3. Assign the scope to one or more enterprise subscribers. For example, assign the EntJunos scope to the Acme enterprise.

   user@host# **edit subscribers retailer ENT subscriber-folder entAcme enterprise Acme**

   [edit subscribers retailer ENT subscriber-folder entAcme enterprise Acme]
   user@host# **set scope EntJunos**

4. Verify your configuration.

```
[edit subscribers retailer ENT subscriber-folder entAcme enterprise Acme]
user@host# show
scope EntJunos;
```

If you use a portal to manage enterprises, you see only the services for the specified scope from the portal. Other services are not visible to the IT managers who manage services and subscriptions from the enterprise service portal. To see the services available to Acme from Enterprise Manager Portal, see *SRC-PE Subscribers and Subscriptions Guide, Chapter 29, Managing Services with Enterprise Manager Portal*.

Example: Customizing Generic Services to Particular Regions

You could use service scopes to customize a generic audio service called Audio-Bronze on a regional basis. This example assumes that the network is configured so that VR boston serves the Boston subnet and VR chicago serves the Chicago subnet.

When the network starts operating, the SAE substitutes the parameters you specified in the service scope definition for the corresponding fields in the service subordinate to that scope.

To customize the new service Audio-Bronze for the Boston and Chicago subnets:

1. Add the Audio-Bronze service within a service scope called boston, and configure the IP address and mask used by VR boston in the parameter configuration.

   This IP address and mask determine an access point to the service provider's equipment.

   user@host# **edit services scope boston**

   [edit services scope boston]
   user@host# **edit service Audio-Bronze**

   [edit services scope boston service Audio-Bronze]
   user@host# **set parameter service-ip-address 10.10.40.33**

   [edit services scope boston service Audio-Bronze]
   user@host# **set parameter service-ip-mask 255.255.255.255**

2. Add another Audio-Bronze service within a service scope called scope_chicago, and specify the IP address and mask used by VR chicago.

   user@host# **edit services scope chicago**

   [edit services scope chicago]
   user@host# **edit service Audio-Bronze**

   [edit services scope chicago service Audio-Bronze]
   user@host# **set parameter service-ip-address 10.10.55.1**

   [edit services scope chicago service Audio-Bronze]
   user@host# set **parameter service-ip-mask 255.255.255.255**

3. Assign service scope boston to virtual router boston.

   user@host# **edit shared network device region_one virtual-router boston**

   [edit shared network device region_one virtual-router boston]
   user@host# **set scope boston**

4. Assign service scope chicago to virtual router chicago.

   user@host# **edit shared network device region_two virtual-router chicago**

   [edit shared network device region_two virtual-router chicago]
   user@host# **set scope chicago**

Restricting Service Activation

You can configure services that cannot be deactivated by an SAE API call; for example, service deactivated from a portal application. This feature is useful when a subscriber has access to several services that perform similar functions, and must use one and only one of those services at a time.

In this case, you must complete three actions:

1. Configure one of the services as a permanent service. This configuration causes the SAE to activate one of the services automatically when the SAE creates a subscriber session.

2. Configure each service to be activate only. This configuration prevents the SAE from deactivating the only active service of this type.

3. Add all services to a mutex group. This configuration allows the SAE to activate one of the other services and to deactivate the service that is currently active.

For example, a subscriber may be able to use one of three Internet access services, each of which offers different speeds. If you configure one of these services as a permanent service, the SAE activates this service for the subscriber automatically. Because all Internet access services are marked to be activate only, the subscriber cannot request deactivation of the default Internet access service. However, if the subscriber requests a faster Internet access service, the SAE activates the faster service and deactivates the default service, because the SAE cannot allow concurrent activation of multiple services assigned to the same mutex group.