Chapter 10

Overview of Configuring Plug-Ins for Solaris Platforms

This chapter describes how to use SDX Admin to configure plug-ins. It also shows how to configure internal and external plug-ins.

You can also use the following to configure plug-ins:

- To use the SRC CLI, see *Chapter 9, Configuring Internal, External, and Synchronization Plug-Ins with the SRC CLI*.
- To use the C-Web interface, see *SRC-PE C-Web Interface Configuration Guide, Chapter 27, Configuring Internal, External, and Synchronization Plug-Ins with the C-Web Interface*.

Topics in this chapter include:

Configuring Plug-Ins with SDX Admin

This section provides guidelines for configuring plug-ins in the SAE property file with SDX Admin or a text editor. See *Modifying the SAE Property File* on page 49 for information about accessing the property file.

Configuring External Plug-Ins

There are two properties that you define for external plug-ins: objectref and attr. You must define both of these properties. Use the syntax:

```
Plugin.<plug-in instance name>.objectref = <object reference>
Plugin.<plug-in instance name>.attr = <attribute>
```

- plug-in instance name—Name that you choose to identify a particular plug-in instance.
- object reference—Specifies the object reference of the plug-in. You can define the object reference by specifying the absolute path to the IOR file, the corbaloc URL, the COS naming service, or the actual IOR.

The following example identifies the object reference by its absolute path to the IOR file:

```
Plugin.admissionControl.objectref = file:///var/acp/acp.ior
```

- attribute—Comma-separated list of attributes that the SAE sends to the plug-in. See *Fields* on page 154 for a list of attributes.

NOTE: Configure only the attributes required. Specifying fewer attributes improves the performance of the SRC network.

Configuring Internal and Hosted Plug-Ins

To define plug-in instances for internal and hosted plug-ins, use the syntax:

```
Plugin.<plug-in instance name>.<property name> = <expression>
```

- plug-in instance name—Name that you choose to identify a particular plug-in instance.
- property name—Each plug-in type has a list of properties that you can define. Use those names to configure properties in the file. Property names are case sensitive. For information about the properties that you can assign, see the section that describes the associated plug-in.
- expression—Sets a value for the property name. For information about the valid values that you can assign to each property, see the section that describes the associated plug-in.

For internal and hosted plug-ins, you must define the class property, which identifies the Java class name of the plug-in. The following example identifies the Java class name for plug-in instance ldapAuth:

```
Plugin.ldapAuth.class = net.juniper.smgt.sae.plugin.LdapAuthenticator
```

For the Java class names of tracking plug-ins, see Table 16 on page 152. For the Java class names of authorization plug-ins, see Table 17 on page 161.

Defining RADIUS Packets

To create templates that define RADIUS packets in flexible RADIUS accounting and authentication plug-ins, use the syntax:

```
RadiusPacket.<template instance name>. <packet-type>.<id>[.type] =
<expression>
```

- template instance name—Name that you choose to identify the template.
- packet-type—Assign one of the values described in Table 18 on page 177.

- id[.type]—Identifies a RADIUS attribute; use as described in *Property* on page 179.
- expression—Assigns a value to the RADIUS attribute; use in the same way as described in *Value* on page 180.

Setting Up the Plug-In Instance to Use a Template

To set up a RADIUS plug-in to use a template, define the template property as follows:

```
Plugin.<plug-in instance name>.template = RadiusPacket.<template instance name>
```

For example, to use the stdAuth template in the flexRadiusAuth plugin instance:

```
Plugin.flexRadiusAuth.template = RadiusPacket.stdAuth
```

Configuring Event Publishers

To configure global and default retailer event publishers, use the following syntax:

```
<event publisher>=<list of plug-in instances>
```

- Event publisher—Name of property that identifies the event publisher. See *Configuring Global and Default Retailer Event Publishers* on page 185 for the property names of global and default retailer publishers.
- List of plug-in instances—Comma-separated list of plug-in instances to which you want the publisher to send events.

The following is the default event publisher configuration. It sets the global subscriber tracking and global service tracking publishers to send events to the fileAcct plug-in instance, and sets the default retailer publisher to send events to ldapAuth.

```
#global plug-ins
User.auth.plugins =
User.tracking.plugins = fileAcct
Service.auth.plugins =
Service.tracking.plugins = fileAcct
Subscription.auth.plugins =
# default user authentication
Retailer.auth.plugins = ldapAuth
Interface.tracking.plugins =
# default dhcp authentication
Retailer.dhcpauth.plugins =
```

Example: LDAP Authentication Plug-In

The following LDAP authentication plug-in searches for objects of class inetOrgPerson, where the username is stored as the common name (cn):

```
Plugin.ldapAuthFoo.class =\ com.junipernetworks.ssc.plugin.LdapAuthenticator
Plugin.ldapAuthFoo.method = search
Plugin.ldapAuthFoo.host = 10.1.2.3
```

```
Plugin.ldapAuthFoo.bindDN = cn=admin
Plugin.ldapAuthFoo.bindPW = {BASE64}c3Nw
Plugin.ldapAuthFoo.filter = (objectclass=inetOrgPerson)
Plugin.ldapAuthFoo.nameAttr = cn
Plugin.ldapAuthFoo.pwdAttr = userPassword
```

Example: Basic RADIUS Accounting Plug-In

The following example configures the basic RADIUS accounting plug-in. The name of the plug-in instance is radiusAcct-1. It communicates with two peers: peer 0 over port 1813 at address 10.1.2.3 and peer 1 over port 1813 at 10.1.2.4. Load-balancing is set to failover. The RADIUS Calling-Station-Id is not sent to the plug-in.

```
Plugin.radiusAcct-1.class = net.juniper.smgt.sae.plugin.\
RadiusTrackingPluginEventListener
Plugin.radiusAcct-1.loadBalancingMode = failover
Plugin.radiusAcct-1.local.timeout = 10000
Plugin.radiusAcct-1.CallingStationId = no
Plugin.radiusAcct-1.peer.0.remote.address = 10.1.2.3
Plugin.radiusAcct-1.peer.0.remote.port = 1813
Plugin.radiusAcct-1.peer.0.remote.password = secret
Plugin.radiusAcct-1.peer.1.remote.password = {BASE64}c2Vjc
Plugin.radiusAcct-1.peer.1.remote.address = 10.1.2.4
Plugin.radiusAcct-1.peer.1.remote.port = 1813
Plugin.radiusAcct-1.peer.1.remote.password = secret
```