JBoss Communications JAIN SLEE Diameter Sh-Client Resource Adaptor User Guide

by Alexandre Mendonça

1. Document Conventions

This manual uses several conventions to highlight certain words and phrases and draw attention to specific pieces of information.

In PDF and paper editions, this manual uses typefaces drawn from the *Liberation Fonts* [https://fedorahosted.org/liberation-fonts/] set. The Liberation Fonts set is also used in HTML editions if the set is installed on your system. If not, alternative but equivalent typefaces are displayed. Note: Red Hat Enterprise Linux 5 and later includes the Liberation Fonts set by default.

1.1. Typographic Conventions

Four typographic conventions are used to call attention to specific words and phrases. These conventions, and the circumstances they apply to, are as follows.

`Mono-spaced Bold`

Used to highlight system input, including shell commands, file names and paths. Also used to highlight key caps and key-combinations. For example:

> To see the contents of the file `my_next_bestselling_novel` in your current working directory, enter the `cat my_next_bestselling_novel` command at the shell prompt and press **Enter** to execute the command.

The above includes a file name, a shell command and a key cap, all presented in Mono-spaced Bold and all distinguishable thanks to context.

Key-combinations can be distinguished from key caps by the hyphen connecting each part of a key-combination. For example:

> Press **Enter** to execute the command.
>
> Press **Ctrl**+**Alt**+**F1** to switch to the first virtual terminal. Press **Ctrl**+**Alt**+**F7** to return to your X-Windows session.

The first sentence highlights the particular key cap to press. The second highlights two sets of three key caps, each set pressed simultaneously.

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in `Mono-spaced Bold`. For example:

> File-related classes include `filesystem` for file systems, `file` for files, and `dir` for directories. Each class has its own associated set of permissions.

Proportional Bold

This denotes words or phrases encountered on a system, including application names; dialogue box text; labelled buttons; check-box and radio button labels; menu titles and sub-menu titles. For example:

> Choose **System > Preferences > Mouse** from the main menu bar to launch **Mouse Preferences**. In the **Buttons** tab, click the **Left-handed mouse** check box and click **Close** to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).
>
> To insert a special character into a **gedit** file, choose **Applications > Accessories > Character Map** from the main menu bar. Next, choose **Search > Find** from the **Character Map** menu bar, type the name of the character in the **Search** field and click **Next**. The character you sought will be highlighted in the **Character Table**. Double-click this highlighted character to place it in the **Text to copy** field and then click the **Copy** button. Now switch back to your document and choose **Edit > Paste** from the **gedit** menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in Proportional Bold and all distinguishable by context.

Note the **>** shorthand used to indicate traversal through a menu and its sub-menus. This is to avoid the difficult-to-follow 'Select **Mouse** from the **Preferences** sub-menu in the **System** menu of the main menu bar' approach.

`Mono-spaced Bold Italic` or ***Proportional Bold Italic***

Whether Mono-spaced Bold or Proportional Bold, the addition of Italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

> To connect to a remote machine using ssh, type `ssh` *`username@domain.name`* at a shell prompt. If the remote machine is `example.com` and your username on that machine is john, type `ssh john@example.com`.
>
> The `mount -o remount` *`file-system`* command remounts the named file system. For example, to remount the `/home` file system, the command is `mount -o remount /home`.
>
> To see the version of a currently installed package, use the `rpm -q` *`package`* command. It will return a result as follows: *`package-version-release`*.

Note the words in bold italics above username, domain.name, file-system, package, version and release. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

> When the Apache HTTP Server accepts requests, it dispatches child processes or threads to handle them. This group of child processes or threads is known as

a *server-pool*. Under Apache HTTP Server 2.0, the responsibility for creating and maintaining these server-pools has been abstracted to a group of modules called *Multi-Processing Modules* (*MPMs*). Unlike other modules, only one module from the MPM group can be loaded by the Apache HTTP Server.

1.2. Pull-quote Conventions

Two, commonly multi-line, data types are set off visually from the surrounding text.

Output sent to a terminal is set in `Mono-spaced Roman` and presented thus:

```
books        Desktop   documentation  drafts  mss    photos   stuff  svn
books_tests  Desktop1  downloads      images  notes  scripts  svgs
```

Source-code listings are also set in `Mono-spaced Roman` but are presented and highlighted as follows:

```
package org.jboss.book.jca.ex1;

import javax.naming.InitialContext;

public class ExClient
{
   public static void main(String args[])
      throws Exception
   {
      InitialContext iniCtx = new InitialContext();
      Object         ref    = iniCtx.lookup("EchoBean");
      EchoHome       home   = (EchoHome) ref;
      Echo           echo   = home.create();

      System.out.println("Created Echo");

      System.out.println("Echo.echo('Hello') = " + echo.echo("Hello"));
   }

}
```

1.3. Notes and Warnings

Finally, we use three visual styles to draw attention to information that might otherwise be overlooked.

Note

A note is a tip or shortcut or alternative approach to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on a trick that makes your life easier.

Important

Important boxes detail things that are easily missed: configuration changes that only apply to the current session, or services that need restarting before an update will apply. Ignoring Important boxes won't cause data loss but may cause irritation and frustration.

Warning

A Warning should not be ignored. Ignoring warnings will most likely cause data loss.

2. Provide feedback to the authors!

If you find a typographical error in this manual, or if you have thought of a way to make this manual better, we would love to hear from you! Please submit a report in the the *Issue Tracker* [http://bugzilla.redhat.com/bugzilla/], against the product **JBoss Communications JAIN SLEE Diameter Sh-Client Resource Adaptor**, or contact the authors.

When submitting a bug report, be sure to mention the manual's identifier: JAIN_SLEE_DIAMETER_SH_CLIENT_RA_User_Guide

If you have a suggestion for improving the documentation, try to be as specific as possible when describing it. If you have found an error, please include the section number and some of the surrounding text so we can find it easily.

Introduction to JBoss Communications JAIN SLEE Diameter Sh-Client Resource Adaptor

This resource adaptor provides a Diameter API for JAIN SLEE applications, according to Sh inteface based on Diameter protocol.

Sh is a Diameter application that allows a Diameter server and a Diameter client to:

- download and update transparent and non-transparent user data;
- request and send notifications on changes on user data

Events represent Diameter Sh messages received by the Diameter stack. Different events types are specified for each Diameter request or answer. Since this RA only operates on the client-side, events are fired on client activities.

The Activities are defined by RA Type to ease use of RA. Activities represent Diameter session between two peers. SLEE applications use activities to create, send and receive messages.

Resource Adaptor Type

Diameter Sh-Client Resource Adaptor Type is defined by Mobicents team as part of effort to standardize RA Types.

2.1. Activities

Diameter Sh-Client Type 2.5.0.FINAL defines the following Activities:

net.java.slee.resource.diameter.sh.client.ShClientActivity
: This type of activity represents client side of Sh session. User-Data-Request (UDR), Profile-Update-Request (PUR) and Subscribe-Notifications-Request (SNR) can be created and sent in this Activity, receiving the respective Answer (or timeout) later on this Activity.

  This activity type can be created with call to the `createShClientActivity` method of `net.java.slee.resource.diameter.sh.client.ShClientProvider`. It ends once underlying Sh client session ends.

  This activity sessions are simple Request-Answer sessions, meaning that once the Answer is sent or received it is terminated.

net.java.slee.resource.diameter.sh.client.ShClientSubscriptionActivity
: This type of activity represents a subscription to User Data notifications for Sh session, based on User Identity. Push-Notification-Request (PNR) messages are received in this Activity and respective Answers are sent from it. Subscribe-Notifications-Request (SNR) can be created and sent in this Activity, receiving the respective Answer (or timeout) later on this Activity.

  This activity type can be created with call to the `createShClientSubscriptionActivity` method of `net.java.slee.resource.diameter.sh.client.ShClientProvider`. It ends once SNR with UNSUBSCRIBE as Subs-Req-Type is sent (can be made using the Activity method `sendUnsubscribeRequest`.)

All activities define methods required to properly function and expose necessary information to JAIN SLEE services. Sh Client Activity is defined as follows:

```
void sendUserDataRequest(UserDataRequest message) throws IOException;

void sendSubscribeNotificationsRequest(SubscribeNotificationsRequest message)
    throws IOException;

void sendProfileUpdateRequest(ProfileUpdateRequest message) throws IOException;
```

void sendUserDataRequest(UserDataRequest message) throws IOException;
This method sends a User-Data-Request message asynchronously.

void sendSubscribeNotificationsRequest(SubscribeNotificationsRequest message) throws IOException;
This method sends a Subscribe-Notifications-Request message asynchronously.

void sendProfileUpdateRequest(ProfileUpdateRequest message) throws IOException;
This method sends a Profile-Update-Request message asynchronously.

Sh Client Subscription Activity is defined as follows:

```
ShClientMessageFactory getClientMessageFactory();

void sendSubscribeNotificationsRequest(SubscribeNotificationsRequest request)
  throws IOException;

void sendUnsubscribeRequest() throws IOException;

PushNotificationAnswer createPushNotificationAnswer();

void sendPushNotificationAnswer(PushNotificationAnswer answer) throws IOException;

void sendPushNotificationAnswer(long resultCode,boolean isExperimentalResultCode)
  throws IOException;

PushNotificationAnswer createPushNotificationAnswer(long resultCode,
  boolean isExperimaental);

UserIdentityAvp getSubscribedUserIdentity();
```

ShClientMessageFactory getClientMessageFactory();
This method returns a 'customized' message factory to manually create Sh Client Messages for this Activity.

void sendSubscribeNotificationsRequest(SubscribeNotificationsRequest request) throws IOException;
This method sends a Subscribe-Notifications-Request message.

void sendUnsubscribeRequest() throws IOException;
This method send a Subscribe-Notifications-Request message containing the AVPs required to UNSUBSCRIBE from the user that this activity represents a subscription to.

PushNotificationAnswer createPushNotificationAnswer();

This method creates Push-Notification-Answer for received Push-Notification-Request. It returns null if a Push-Notification-Request has not been received.

void sendPushNotificationAnswer(PushNotificationAnswer answer) throws IOException;

This method sends a manually-constructed Push-Notification-Answer to the peer that sent the Push-Notification-Request.

void sendPushNotificationAnswer(long resultCode,boolean isExperimentalResultCode) throws IOException;

This is a convenience method to create and send a Push-Notification-Answer containing a Result-Code or Experimental-Result AVP populated with the given value.

PushNotificationAnswer createPushNotificationAnswer(long resultCode, boolean isExperimaental);

This method creates Push-Notification-Answer for received Push-Notification-Request, with the specified Result-Code. It returns null if a Push-Notification-Request has not been received.

UserIdentityAvp getSubscribedUserIdentity();

This method returns the User-Identity for the subscription in the HSS represented by this activity.

Note

It is safe to type cast all the mentioned Diameter Activities to it's super interface `net.java.slee.resource.diameter.base.DiameterActivity` defined in Diameter Base Activities section.

2.2. Events

Diameter Sh-Client Resource Adaptor Type declares all the Diameter Sh messages related to client operations.

The following tables shows which events are fired on each activity.

Table 2.1. Events received on Sh Client Activity

Name	Vendor	Version	Class
net.java.slee.resource.diameter.sh.events.UserDataAnswer	java.net	0.8	net.java.slee.resource.diameter.sh.events.UserDataAnswer
net.java.slee.resource.diameter.sh.events.ProfileUpdateAnswer	java.net	0.8	net.java.slee.resource.diameter.sh.events.ProfileUpdateAnswer

Name	Vendor	Version	Class
net.java.slee.resource.diameter.sh.events.SubscribeNotificationsAnswer	java.net	0.8	net.java.slee.resource.diameter.sh.events.SubscribeNotificationsAnswer

Table 2.2. Events received on Sh Client Subscription Activity

Name	Vendor	Version	Class
net.java.slee.resource.diameter.sh.events.PushNotificationRequest	java.net	0.8	net.java.slee.resource.diameter.sh.events.PushNotificationRequest

Important

Spaces where introduced in `Name` and `Event Class` column values, to correctly render the table. Please remove them when using copy/paste.

2.3. Activity Context Interface Factory

The JBoss Communications Diameter Sh-Client Activity Context Interface Factory is defined as follows:

```
package net.java.slee.resource.diameter.sh.client;

import javax.slee.ActivityContextInterface;
import javax.slee.UnrecognizedActivityException;

public interface ShClientActivityContextInterfaceFactory {

    ActivityContextInterface getActivityContextInterface(ShClientActivity activity)
        throws UnrecognizedActivityException;

    ActivityContextInterface getActivityContextInterface(ShClientSubscriptionActivity activity)
        throws UnrecognizedActivityException;

}
```

2.4. Resource Adaptor Interface

The JBoss Communications Diameter Sh-Client Resource Adaptor SBB Interface provides SBBs with access to the Diameter objects required for creating and sending messages. It is defined as follows:

```
package net.java.slee.resource.diameter.sh.client;

import java.io.IOException;

import net.java.slee.resource.diameter.base.CreateActivityException;
import net.java.slee.resource.diameter.base.events.avp.DiameterIdentity;
import net.java.slee.resource.diameter.sh.DiameterShAvpFactory;
import net.java.slee.resource.diameter.sh.events.ProfileUpdateAnswer;
import net.java.slee.resource.diameter.sh.events.ProfileUpdateRequest;
import net.java.slee.resource.diameter.sh.events.SubscribeNotificationsAnswer;
import net.java.slee.resource.diameter.sh.events.SubscribeNotificationsRequest;
import net.java.slee.resource.diameter.sh.events.UserDataAnswer;
import net.java.slee.resource.diameter.sh.events.UserDataRequest;

public interface ShClientProvider {

    public ShClientMessageFactory getClientMessageFactory();

    public DiameterShAvpFactory getClientAvpFactory();

    public ShClientActivity createShClientActivity() throws CreateActivityException;

    public ShClientSubscriptionActivity createShClientSubscriptionActivity()
        throws CreateActivityException;

  public UserDataAnswer userDataRequest(UserDataRequest message) throws IOException;

    public ProfileUpdateAnswer profileUpdateRequest(ProfileUpdateRequest message)
        throws IOException;

    public SubscribeNotificationsAnswer subscribeNotificationsRequest(
        SubscribeNotificationsRequest message) throws IOException;

    public DiameterIdentity[] getConnectedPeers();

    public int getPeerCount();
```

```
  }
```

public ShClientMessageFactory getClientMessageFactory();

This method returns a `ShClientMessageFactory` implementation to be used to create `DiameterMessage` objects.

public DiameterShAvpFactory getClientAvpFactory();

This method returns a `DiameterShAvpFactory` implementation to be used to create `DiameterAvp` objects.

public ShClientActivity createShClientActivity() throws CreateActivityException;

This method creates a new Sh Client activity to send and receive Diameter Sh messages.

public ShClientSubscriptionActivity createShClientSubscriptionActivity() throws CreateActivityException;

This method creates a new Sh Client Subscription activity to send and receive Diameter Sh subscription related messages.

public UserDataAnswer userDataRequest(UserDataRequest message) throws IOException;

This method sends a synchronous UserDataRequest which will block until an answer is received from the peer.

public ProfileUpdateAnswer profileUpdateRequest(ProfileUpdateRequest message) throws IOException;

This method sends a synchronous ProfileUpdateRequest which will block until an answer is received from the peer.

public SubscribeNotificationsAnswer subscribeNotificationsRequest(SubscribeNotificationsRequest message) throws IOException;

This method sends a synchronous SubscribeNotificationsRequest which will block until an answer is received from the peer.

public DiameterIdentity[] getConnectedPeers();

This method returns the identities of peers this Diameter resource adaptor is connected to.

public int getPeerCount();

This method returns the number of peers this Diameter resource adaptor is connected to.

2.5. Restrictions

Current Resource Adaptor Type has no defined restrictions.

2.6. Sbb Code Examples

Simple Client-Side Example that creates and sends an User-Data-Request and receives an User-Data-Answer.

```
/* Method for creating and sending UDR with pre-defined values. */
private void doSimpleTestsSendUDR() {

    try {
        ShClientActivity basicClientActivity = this.provider.createShClientActivity();

        ActivityContextInterface localACI = acif.getActivityContextInterface(basicClientActivity);
        logger.info(" On TimerEvent: ACI created for basicClientActivity");

        localACI.attach(getSbbContext().getSbbLocalObject());

        DiameterIdentity[] peers = provider.getConnectedPeers();

        UserDataRequest udr = ((ShClientMessageFactory)basicClientActivity.
            getDiameterMessageFactory()).createUserDataRequest();

        List<DiameterAvp> avps = new ArrayList<DiameterAvp>();

       avps.add(avpFactory.getBaseFactory().createAvp(Avp.DESTINATION_HOST, ("aaa://" +
            destinationIP + ":" + destinationPort).getBytes() ));
        avps.add(avpFactory.getBaseFactory().createAvp(Avp.DESTINATION_REALM,
            destinationRealm.getBytes()));
        UserIdentityAvp ui=avpFactory.createUserIdentity();
        ui.setPublicIdentity("sip:subscriber@mobicents.org");

        avps.add(ui);

        udr.setExtensionAvps(avps.toArray(new DiameterAvp[avps.size()]));
        udr.setAuthSessionState(AuthSessionStateType.STATE_MAINTAINED);
        udr.setDataReference(DataReferenceType.IMS_PUBLIC_IDENTITY);
        basicClientActivity.sendUserDataRequest(udr);
    }
    catch (Exception e) {
      tracer.error("Failure trying to create/send UDR.", e);
    }
}

...

/* Method for handling ACA messages. Just print the Result-Code AVP. */
public void onUserDataRequestAnswer(UserDataAnswer uda, ActivityContextInterface aci)
    if (tracer.isInfoEnabled()) {
```

```
            tracer.info("User-Data-Answer received. Result-Code[" + uda.getResultCode() + "].");
        }
    }
```

Resource Adaptor Implementation

This RA uses the JBoss Communications Diameter Stack, an improvement over *jDiameter Stack* [http://jdiameter.dev.java.net]. The stack is the result of the work done by JBoss Communications Diameter and jDiameter development teams, and source code is provided in all releases.

3.1. Configuration

The Resource Adaptor supports configuration only at Resource Adaptor Entity creation time, the following table enumerates the configuration properties:

Table 3.1. Resource Adaptor's Configuration Properties

Property Name	Description	Property Type	Default Value
authApplicationIds	List of supported Authorization Application Ids in form of {vendor}:{application-id}, separated by comma ','	java.lang.String	10415:16777217

Important

JAIN SLEE 1.1 Specification requires values set for properties without a default value, which means the configuration for those properties are mandatory, otherwise the Resource Adaptor Entity creation will fail!

3.2. Default Resource Adaptor Entities

There is a single Resource Adaptor Entity created when deploying the Resource Adaptor, named `DiameterShClient`. The `DiameterShClient` entity uses the default Resource Adaptor configuration, specified in *Section 3.1, "Configuration"*.

The `DiameterShClient` entity is also bound to Resource Adaptor Link Name `DiameterShClient`, to use it in an Sbb add the following XML to its descriptor:

```
<resource-adaptor-type-binding>
  <resource-adaptor-type-ref>
    <resource-adaptor-type-name>Diameter Sh-Client</resource-adaptor-type-name>
    <resource-adaptor-type-vendor>java.net</resource-adaptor-type-vendor>
```

```
    <resource-adaptor-type-version>0.8.1</resource-adaptor-type-version>
  </resource-adaptor-type-ref>

  <activity-context-interface-factory-name>
    slee/resources/JDiameterShClientResourceAdaptor/java.net/0.8.1/acif
  </activity-context-interface-factory-name>

  <resource-adaptor-entity-binding>
    <resource-adaptor-object-name>
       slee/resources/diameter-sh-client-ra-interface
    </resource-adaptor-object-name>
    <resource-adaptor-entity-link>DiameterShClient</resource-adaptor-entity-link>
  </resource-adaptor-entity-binding>
</resource-adaptor-type-binding>
```

3.3. Traces and Alarms

3.3.1. Tracers

Each Resource Adaptor Entity uses a single JAIN SLEE 1.1 Tracer, named `DiameterShClientResourceAdaptor`. The related Log4j Logger category, which can be used to change the Tracer level from Log4j configuration, is `javax.slee.RAEntityNotification[entity=DiameterShClient]`.

3.3.2. Alarms

No alarms are set by this Resource Adaptor.

Setup

4.1. Pre-Install Requirements and Prerequisites

Ensure that the following requirements have been met before continuing with the install.

4.1.1. Hardware Requirements

The Resource Adaptor hardware's main concern is RAM memory and Java Heap size, the more the better. For instance, while the underlying JBoss Communications JAIN SLEE may run with 1GB of RAM, 8GB is needed to achieve performance higher than 800 new requests per second.

Of course, memory is only needed to store the Resource Adaptor state, the faster the CPU more requests per second are supported, yet no particular CPU is a real requirement to use the RA.

4.1.2. Software Prerequisites

The RA requires JBoss Communications JAIN SLEE properly set and Mobicents Diameter Multiplexer (MUX), which includes the stack, and JBoss Communications Diameter Base RA to be properly installed too.

4.2. JBoss Communications JAIN SLEE Diameter Sh-Client Resource Adaptor Source Code

4.2.1. Release Source Code Building

1. **Downloading the source code**

Important

Subversion is used to manage its source code. Instructions for using Subversion, including install, can be found at *http://svnbook.red-bean.com*

Use SVN to checkout a specific release source, the base URL is ?, then add the specific release version, lets consider 2.5.0.FINAL.

```
[usr]$ svn co ?/2.5.0.FINAL slee-ra-diameter-sh-client-2.5.0.FINAL
```

2. **Building the source code**

Important

Maven 2.0.9 (or higher) is used to build the release. Instructions for using Maven2, including install, can be found at *http://maven.apache.org*

Use Maven to build the deployable unit binary.

```
[usr]$ cd slee-ra-diameter-sh-client-2.5.0.FINAL
[usr]$ mvn install
```

Once the process finishes you should have the `deployable-unit` jar file in the `target` directory, if JBoss Communications JAIN SLEE is installed and environment variable JBOSS_HOME is pointing to its underlying JBoss Enterprise Application Platform directory, then the deployable unit jar will also be deployed in the container.

4.2.2. Development Trunk Source Building

Similar process as for *Section 4.2.1, "Release Source Code Building"*, the only change is the SVN source code URL, which is NOT AVAILABLE.

4.3. Installing JBoss Communications JAIN SLEE Diameter Sh-Client Resource Adaptor

To install the Resource Adaptor simply execute provided ant script `build.xml` default target:

```
[usr]$ ant
```

The script will copy the RA deployable unit jar to the `default` JBoss Communications JAIN SLEE server profile deploy directory, to deploy to another server profile use the argument `-Dnode=`.

4.4. Uninstalling JBoss Communications JAIN SLEE Diameter Sh-Client Resource Adaptor

To uninstall the Resource Adaptor simply execute provided ant script `build.xml undeploy` target:

```
[usr]$ ant undeploy
```

The script will delete the RA deployable unit jar from the `default` JBoss Communications JAIN SLEE server profile deploy directory, to undeploy from another server profile use the argument `-Dnode=`.

Clustering

Currently JBoss Communications Diameter Sh-Client RA implementation does not support clustering.

Appendix A. Revision History

Revision History
Revision 1.0 Tue Feb 09 2010 AlexandreMendonça
Creation of the JBoss Communications JAIN SLEE Diameter Sh-Client RA User Guide.

Index

F