JBoss Communications
JAIN SLEE Diameter CCA
Resource Adaptor User Guide

by Eduardo Martins, Bartosz Baranowski, and 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 CCA Resource Adaptor**, or contact the authors.

When submitting a bug report, be sure to mention the manual's identifier: JAIN_SLEE_DIAMETER_CCA_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 CCA Resource Adaptor

This resource adaptor provides DIAMETER API for JSLEE applications. *CCA* [http://tools.ietf.org/html/rfc4006] is a RFC specification of DIAMETER extension to allow real time credit control of end user services, ie. SIP services, equipment access, network access, ... etc.

Events represent DIAMETER CCA messages received by stack, that is requests and answers with command code equal to 272. Events are fired either on client or server activity.

The activities are custom 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 CCA Resource Adaptor Type is defined by Mobicents team as part of effort to standarize RA Types.

2.1. Activities

Diameter CCA Type 2.6.1.FINAL defines two types of activities:

net.java.slee.resource.diameter.cca.CreditControlClientSession
: This type of activity represents client side of credit control session. It is a source of answers to credit control requests issued with this activity. It is also source of re-authentication requests.

  This activity type can be created with call to one of methods of CreditControlProvider. It ends once underlying credit control session ends. State machine for client credit control can be found *here* [http://tools.ietf.org/html/rfc4006#section-5] and *here* [http://tools.ietf.org/html/rfc4006#section-6] .

net.java.slee.resource.diameter.cca.CreditControlServerSession
: This type of activity represents server side of credit control session. It is source of credit control requests issued from client side and answers to re-auth requests.

  This activity type is created explicitly for incoming requests by Resource Adaptor. It ends once underlying credit control session ends. State machine for server credit control can be found *here* [http://tools.ietf.org/html/rfc4006#section-5] and *here* [http://tools.ietf.org/html/rfc4006#section-6]

Both activities define methods required to properly function and expose necessary information to JSLEE services. Common part for each is defined as follows:

```
public CreditControlSessionState getState();

public String getSessionId();

public CreditControlAVPFactory getCCAAvpFactory();

public CreditControlMessageFactory getCCAMessageFactory();
```

CreditControlSessionState getState();
: returns current state of credit control session. State directly follows definition from credit control *rfc* [http://tools.ietf.org/html/rfc4006] .

String getSessionId();
returns session Id of underlying diameter session.

CreditControlAVPFactory getCCAAvpFactory()
returns AVP factory capable of creating specifc AVPs for credit control. It also exposes means of accessing base AVP factory.

CreditControlMessageFactory getCCAMessageFactory();
returns message factory capable of creating credit control messages. It also exposes means of accessing base factory.

Client type activity interface is defined as follows:

```
CreditControlRequest createCreditControlRequest();

void sendCreditControlRequest(CreditControlRequest ccr) throws IOException;

void sendInitialCreditControlRequest(CreditControlRequest ccr) throws IOException;

void sendReAuthAnswer(ReAuthAnswer rar) throws IOException;

void sendUpdateCreditControlRequest(CreditControlRequest ccr) throws IOException;

void sendTerminationCreditControlRequest(CreditControlRequest ccr) throws IOException;
```

CreditControlRequest createCreditControlRequest();
Creates credit control request for this activity. Fills AVPs appropriate for this session..

void sendCreditControlRequest(CreditControlRequest ccr) throws IOException;
sends credit control request, does not perform any other operations.

void sendInitialCreditControlRequest(CreditControlRequest ccr) throws IOException;
sends credit control request. Ensures that its initial request by setting proper AVP .

void sendUpdateCreditControlRequest(CreditControlRequest ccr) throws IOException;
sends credit control request. Ensures that its update request by setting proper AVP .

void sendTerminationCreditControlRequest(CreditControlRequest ccr) throws IOException;
sends credit control request. Ensures that its termination request by setting proper AVP .

void sendReAuthAnswer(ReAuthAnswer rar) throws IOException;
sends re-authentication answer on this session.

Server type activity interface is defined as follows:

```
CreditControlAnswer createCreditControlAnswer();

void sendCreditControlAnswer(CreditControlAnswer cca) throws IOException;

void sendReAuthRequest(ReAuthRequest rar) throws IOException;
```

CreditControlAnswer createCreditControlAnswer();
creates credit control answer for last received request.

void sendCreditControlAnswer(CreditControlAnswer cca) throws IOException;
sends credit control answer through this session.

void sendReAuthRequest(ReAuthRequest rar) throws IOException;
sends re-authentication request through this session.

Note

It is safe to type cast activities to interface defined in Base RA Type: net.java.slee.resource.diameter.base.DiameterActivity

2.2. Events

Credit Control declares only two additional events, to those defined in Base RA Type, namely CreditControlAnswer and CreditControlRequest. However, it also reuses Base events.

Table 2.1. Events received on Server Activity

Name	Vendor	Version	Class
net.java.slee.resource.diameter.cca.events.CreditControlRequest	java.net	0.8	net.java.slee.resource.diameter.cca.events.CreditControlRequest
net.java.slee.resource.diameter.base.events.ReAuthRequest	java.net	0.8	net.java.slee.resource.diameter.base.events.ReAuthRequest

Table 2.2. Events received on Client Activity

Name	Vendor	Version	Class
net.java. slee.resource .diameter. cca.events. CreditControlAnswer	java.net	0.8	net.java. slee.resource.diameter.cca.events. CreditControlAnswer
net.java. slee.resource .diameter. base.events. ReAuthAnswer	java.net	0.8	net.java. slee.resource.diameter.base.events. ReAuthAnswer

Table 2.3. Events received on both activities

Name	Vendor	Version	Class
net.java.slee.resource. diameter.base.events. SessionTerminationRequest	java.net	0.8	net.java.slee.resource. diameter.base.events. SessionTerminationRequest
net.java.slee.resource. diameter.base.events. SessionTerminationAnswer	java.net	0.8	net.java.slee.resource. diameter.base.events. SessionTerminationAnswer
net.java.slee.resource. diameter.base.events. AbortSessionRequest	java.net	0.8	net.java.slee.resource. diameter.base.events. AbortSessionRequest
net.java.slee.resource. diameter.base.events. AbortSessionAnswer	java.net	0.8	net.java.slee.resource. diameter.base.events. AbortSessionAnswer
net.java.slee.resource. diameter.base.events. AccountingRequest	java.net	0.8	net.java.slee.resource. diameter.base.events. AccountingRequest
net.java.slee.resource. diameter.base.events. AccountingAnswer	java.net	0.8	net.java.slee.resource. diameter.base.events. AccountingAnswer
net.java.slee.resource. diameter.base.events. ErrorAnswer	java.net	0.8	net.java.slee.resource. diameter.base.events. ErrorAnswer
net.java.slee.resource. diameter.base.events. ExtensionDiameterMessage	java.net	0.8	net.java.slee.resource. diameter.base.events. ExtensionDiameterMessage

Important

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

2.3. Activity Context Interface Factory

Activity context interface factory is defined as follows:

```
package net.java.slee.resource.diameter.cca;

import javax.slee.ActivityContextInterface;

public interface CreditControlActivityContextInterfaceFactory {

  public ActivityContextInterface getActivityContextInterface(CreditControlClientSession cccs);

  public ActivityContextInterface getActivityContextInterface(CreditControlServerSession ccss);

}
```

2.4. Resource Adaptor Interface

Resource Adaptor SBB Interface provides SBBs with means of accessing RA objects required for interaction in DIAMETER world. It is defined as follows:

```
  package net.java.slee.resource.diameter.cca;

  import net.java.slee.resource.diameter.base.CreateActivityException;
  import net.java.slee.resource.diameter.base.events.avp.DiameterIdentity;

  public interface CreditControlProvider {

    CreditControlClientSession createClientSession() throws CreateActivityException;
```

```
    CreditControlClientSession createClientSession(DiameterIdentity destinationHost,
      DiameterIdentity destinationRealm) throws CreateActivityException;

    CreditControlMessageFactory getCreditControlMessageFactory();

    CreditControlAVPFactory getCreditControlAVPFactory();

    int getPeerCount();

    DiameterIdentity[] getConnectedPeers();

  }
```

CreditControlClientSession createClientSession() throws CreateActivityException;
Creates new client session. Its destination is unknown, its determined on send.

CreditControlClientSession createClientSession(DiameterIdentity destinationHost, DiameterIdentity destinationRealm) throws CreateActivityException;
Creates new client session. Its destination is passed upon creation.

CreditControlMessageFactory getCreditControlMessageFactory();
Retrieve default message factory.

CreditControlAVPFactory getCreditControlAVPFactory();
Retrieve default AVP factory.

int getPeerCount();
Get number of currently connected peers. This may change during runtime, depends on network situation.

DiameterIdentity[] getConnectedPeers();
Get currently connected peers. This may change during runtime, depends on network situation.

2.5. Restrictions

Current Resource Adaptor Type has no defined restrictions.

2.6. Sbb Code Examples

Simple client side requesting credits, with EVENT_TYPE credit control request:

```
private void doSendEventCCR(int currencyCode,int value) {
```

```
try {
    //Create session.
    CreditControlClientSession session = this.provider.createClientSession();
    ActivityContextInterface localACI = this.acif.getActivityContextInterface(session);
    localACI.attach(this.getSbbContext().getSbbLocalObject());

    CreditControlRequest request = session.createCreditControlRequest();

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

    avps.add(avpFactory.getBaseFactory().createAvp(Avp.ORIGIN_HOST,
        ("aaa://" + originIP + ":" + originPort).getBytes()));
    avps.add(avpFactory.getBaseFactory().createAvp(Avp.ORIGIN_REALM,
        originRealm.getBytes()));
    //or dedicated:
    //request.setDestinationHost(destinationHost);
    //request.setDestinationRealm(destinationRealm);
    avps.add(avpFactory.getBaseFactory().createAvp(Avp.DESTINATION_HOST,
        ("aaa://" + destinationIP + ":" + destinationPort).getBytes()));
    avps.add(avpFactory.getBaseFactory().createAvp(Avp.DESTINATION_REALM,
        destinationRealm.getBytes()));
    //Event Type request has request type set to '4'
    avps.add(avpFactory.getBaseFactory().createAvp(
        CreditControlAVPCodes.CC_Request_Type, 4l));
    avps.add(avpFactory.getBaseFactory().createAvp(
        CreditControlAVPCodes.CC_Request_Number, 0l));
        RequestedServiceUnitAvp rsu = this.avpFactory.createRequestedServiceUnit();
        CcMoneyAvp ccMoney = this.avpFactory.createCcMoney();
    ccMoney.setCurrencyCode(currencyCode);
    UnitValueAvp unitValue = this.avpFactory.createUnitValue();
    unitValue.setValueDigits(value);
    ccMoney.setUnitValue(unitValue);

    rsu.setCreditControlMoneyAvp(ccMoney);
        avps.add(rsu);
        avps.add(avpFactory.getBaseFactory().createAvp(
        CreditControlAVPCodes.Requested_Action, 0));
    //use extension avps to fill message or dedicated setters/getters
    request.setExtensionAvps(avps.toArray(new DiameterAvp[avps.size()]));
    // Now create and send

    if (logger.isInfoEnabled())
        logger.info("About to send:\n" + request);
        session.sendCreditControlRequest(request);
```

```
    } catch (Exception e) {
        logger.error("Failed to create/send Credit-Control-Request.", e);
    }
}


    ....

  public void onCreditControlAnswer(CreditControlAnswer answer, ActivityContextInterface aci) {
        logger.info("Received CCA with Result-Code[" + answer.getResultCode() + "].");
        switch (answer.getCcRequestType().getValue()) {
        case 1:
        case 2:
        case 3:
          logger.error("Received 'session' credit control answer. Expecting EVENT_TYPE answer");
            break;
        case 4:
           //result code 2xxx is success
           if(answer.getResultCode()/1000 == 2 )
           {
              //assume msg is correct
              GrantedServiceUnitAvp gsu = answer.getGrantedServiceUnit();
              CcMoneyAvp money = gsu.getCreditControlMoneyAvp();
              doSpendMoney(money);
           }else
           {
              //error, this will pring formated msg dump.
              logger.warn("Result of CC operation is a failure:\n"+answer);
           }
           break;
        }
    }
```

Server application to handle simple direct debiting:

```
public void onCreditControlRequest(CreditControlRequest request,
    ActivityContextInterface aci) {
    if (logger.isInfoEnabled())
                                        logger.info("Received     Credit-Control-Request     (Application-
Id[" + request.getHeader().getApplicationId() + "].");
```

```
// INITIAL_REQUEST(1), UPDATE_REQUEST(2), TERMINATION_REQUEST(3),
// EVENT_REQUEST(4)
CreditControlServerSession session = (CreditControlServerSession) aci.getActivity();
CreditControlAnswer answer = null;
if(request.getRequestedAction() == RequestedActionType.DIRECT_DEBITING)
{

    switch (request.getCcRequestType().getValue()) {
    case 1:
        try {
            if (logger.isInfoEnabled())
                logger.info("Got INITIAL_REQUEST(1).");

            if (getSentInitialAnswer()) {
                logger.error("Error. Initial answer already sent! Aborting.");
                return;
            }

            answer = session.createCreditControlAnswer();
            if(userHasCredit(request.getUserName(),request.getRequestedServiceUnit()))
            {
                GrantedServiceUnitAvp gsu = chargeUser(request.getRequestedServiceUnit());
                answer.setGrantedServiceUnit(gsu);
                answer.setResultCode(2001);
            }else
            {
                //4012 == CREDIT_LIMIT_REACHED
                answer.setResultCode(4012);
            }
            if (logger.isInfoEnabled()) {
                logger.info("Processed Credit-Control-Request:\n" + request);
                logger.info("Sending Credit-Control-Answer:\n" + answer);
            }

            session.sendCreditControlAnswer(answer);
            this.setSentInitialAnswer(true);
        } catch (Exception e) {
                                logger.error("Failed to create/send Credit-Control-Answer to reply
INITIAL_REQUEST(1).", e);
        }
        break;
    case 2:
        try {
            if (logger.isInfoEnabled())
```

```
                logger.info("Got UPDATE_REQUEST(2).");

            if (getSentUpdateAnswer()) {
                logger.error("Error. Update answer already sent! Aborting.");
                return;
            }

            answer = session.createCreditControlAnswer();
            if(userHasCredit(request.getUserName(),request.getRequestedServiceUnit()))
            {
                GrantedServiceUnitAvp gsu = chargeUser(request.getRequestedServiceUnit());
                answer.setGrantedServiceUnit(gsu);
                answer.setResultCode(2001);
            }else
            {
                //4012 == CREDIT_LIMIT_REACHED
                answer.setResultCode(4012);
            }
            if (logger.isInfoEnabled()) {
                logger.info("Processed Credit-Control-Request:\n" + request);
                logger.info("Sending Credit-Control-Answer:\n" + answer);
            }
            session.sendCreditControlAnswer(answer);
            setSentUpdateAnswer(true);
        } catch (Exception e) {
                            logger.error("Failed to create/send Credit-Control-Answer to reply
UPDATE_REQUEST(2).", e);
        }
        break;
    case 3:
        try {
            if (logger.isInfoEnabled())
                logger.info("Got TERMINATION_REQUEST(3).");

            if (getSentTerminationAnswer()) {
                logger.error("Error. Termination answer already sent! Aborting.");
                return;
            }

            answer = session.createCreditControlAnswer();
            if(userHasCredit(request.getUserName(),request.getRequestedServiceUnit()))
            {
                GrantedServiceUnitAvp gsu = chargeUser(request.getRequestedServiceUnit());
                answer.setGrantedServiceUnit(gsu);
```

```
                answer.setResultCode(2001);
            }else
            {
                //4012 == CREDIT_LIMIT_REACHED
                answer.setResultCode(4012);
            }
            if (logger.isInfoEnabled()) {
                logger.info("Processed Credit-Control-Request:\n" + request);
                logger.info("Sending Credit-Control-Answer:\n" + answer);
            }
            session.sendCreditControlAnswer(answer);
            setSentTerminationAnswer(true);
        } catch (Exception e) {
                            logger.error("Failed to create/send Credit-Control-Answer to reply
TERMINATION_REQUEST(3).", e);
        }
        break;
    case 4:
        try {
            if (logger.isInfoEnabled())
                logger.info("Got EVENT_REQUEST(4).");

            answer = session.createCreditControlAnswer();
            if(userHasCredit(request.getUserName(),request.getRequestedServiceUnit()))
            {
                GrantedServiceUnitAvp gsu = chargeUser(request.getRequestedServiceUnit());
                answer.setGrantedServiceUnit(gsu);
                answer.setResultCode(2001);
            }else
            {
                //4012 == CREDIT_LIMIT_REACHED
                answer.setResultCode(4012);
            }
            if (logger.isInfoEnabled())
                logger.info("Sending Credit-Control-Answer:\n" + answer);

            session.sendCreditControlAnswer(answer);
        } catch (Exception e) {
                            logger.error("Failed to create/send Credit-Control-Answer to reply
EVENT_REQUEST(4).", e);
        }
        break;

    default:
```

```
                                    logger.error("Unexpected CC-Request-Type in message:
" + request.getCcRequestType() + ". Aborting...");
        }
        }else
        {
            answer = session.createCreditControlAnswer();
            //4011 == DIAMETER_CREDIT_CONTROL_NOT_APPLICABLE
            answer.setResultCode(4011);
            session.sendCreditControlAnswer(answer);
        }
    }
```

Resource Adaptor Implementation

This RA uses the JBoss Communications diameter stack to provide service. The stack is the result of the work done by JBoss Communications JAIN SLEE and Diameter development teams.

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	0:4

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 `DiameterCCA`. The `DiameterCCA` entity uses the default Resource Adaptor configuration, specified in *Section 3.1, "Configuration"*.

The `DiameterCCA` entity is also bound to Resource Adaptor Link Name `DiameterCCA`, 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>DiameterCCA</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/CCAResourceAdaptor/java.net/0.8.1/acif
        </activity-context-interface-factory-name>

        <resource-adaptor-entity-binding>
            <resource-adaptor-object-name>
                slee/resources/diameter-cca-ra-interface
            </resource-adaptor-object-name>
            <resource-adaptor-entity-link>DiameterCCA
            </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 `DiameterCCAResourceAdaptor`. The related Log4j Logger category, which can be used to change the Tracer level from Log4j configuration, is `javax.slee.RAEntityNotification[entity=DiameterCCAResourceAdaptor]`.

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. It is recomended that for higher loads than 400 calls per second JVM has atleast 1GB heap.

4.1.2. Software Prerequisites

The RA requires JBoss Communications JAIN SLEE properly set.

4.2. JBoss Communications JAIN SLEE Diameter CCA 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 2.6.1.FINAL.

```
[usr]$ svn co ?/2.6.1.FINAL diameter-cca-2.6.1.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 diameter-cca-2.6.1.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.

Note

Diameter CCA depends on Base RA and MUX, please refer to separate documentation for proper setup.

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 CCA 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 CCA 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

5.1. Failover

The Diameter stack used by the JBoss Communications JAIN SLEE Diameter CCA Resource Adaptor supports application session failover, with specific session state being replicated, thus only available for Application sessions. Failover of application activities is transparent to SLEE applications. This means that SLEE applications must be in charge of properly adapting its state machine to recover generic session on node failure.

5.2. Load Balancing

Currently, the only available balancing mechanism is provided by Diameter stack. It depends on *RFC 3588* [http://tools.ietf.org/html/rfc3588] algorithm to select one peer from realm serving the desired application.

Appendix A. Revision History

Revision History
Revision 1.0 Tue Feb 3 2010 BartoszBaranowski
Creation of the JBoss Communications JAIN SLEE Diameter CCA RA User Guide.

Index

F