JBoss Communications
JAIN SLEE Diameter Cx/Dx
Resource Adaptor User Guide

by Alexandre Mendonça

Preface v
1. Document Conventions v
1.1. Typographic Conventions v
1.2. Pull-quote Conventions vii
1.3. Notes and Warnings vii
2. Provide feedback to the authors! viii
1. Introduction to JBoss Communications JAIN SLEE Diameter Cx/Dx Resource Adaptor 1
2. Resource Adaptor Type 3
2.1. Activities 3
2.2. Events 8
2.3. Activity Context Interface Factory 9
2.4. Resource Adaptor Interface 10
2.5. Restrictions 11
2.6. Sbb Code Examples 11
3. Resource Adaptor Implementation 17
3.1. Configuration 17
3.2. Default Resource Adaptor Entities 17
3.3. Traces and Alarms 18
3.3.1. Tracers 18
3.3.2. Alarms 18
4. Setup 19
4.1. Pre-Install Requirements and Prerequisites 19
4.1.1. Hardware Requirements 19
4.1.2. Software Prerequisites 19
4.2. JBoss Communications JAIN SLEE Diameter Cx/Dx Resource Adaptor Source Code 19
4.2.1. Release Source Code Building 19
4.2.2. Development Trunk Source Building 20
4.3. Installing JBoss Communications JAIN SLEE Diameter Cx/Dx Resource Adaptor.. 20
4.4. Uninstalling JBoss Communications JAIN SLEE Diameter Cx/Dx Resource Adaptor 20
5. Clustering 23
A. Revision History 25
Index 27

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 Cx/Dx Resource Adaptor**, or contact the authors.

When submitting a bug report, be sure to mention the manual's identifier: JAIN_SLEE_DIAMETER_CX_DX_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 Cx/Dx Resource Adaptor

This resource adaptor provides a Diameter API for JAIN SLEE applications, according to Cx/Dx intefaces based on Diameter protocol.

The Cx interface specifies a Diameter application that allows a Diameter Multimedia server and a Diameter Multimedia client:

- to exchange location information
- to authorize a user to access the IMS
- to exchange authentication information
- to download and handle changes in the user data stored in the server

Dx interface is used to locate the proper HSS in a multi-HSS environment.

Cx/Dx interface sessions are implicitly terminated, for which the server does not maintain state information.

Events represent Diameter Cx/Dx messages received by the Diameter stack. Different events types are specified for each Diameter request or answer. Events are fired either on client or server 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 Cx/Dx Resource Adaptor Type is defined by Mobicents team as part of effort to standardize RA Types.

2.1. Activities

Diameter Cx/Dx Type 2.4.0.CR1 defines the following Activities:

net.java.slee.resource.diameter.cxdx.CxDxClientSessionActivity
: This type of activity represents client side of Cx/Dx session. User-Authorization-Request (UAR), Server-Assignment-Request (SAR), Location-Info-Request (LIR) and Multimedia-Authentication-Request (MAR) messages can be created and sent in this Activity, receiving the respective Answer (or timeout) later on this Activity. Registration-Termination-Request (RTR) and Push-Profile-Request (PPR) messages are received in this Activity and respective Answers are sent from it.

  This activity type is implicitly created when receiving Requests or can be created with call to the proper `createCxDxClientSessionActivity` method of `net.java.slee.resource.diameter.cxdx.CxDxProvider` for sending Requests. It ends once underlying Cx/Dx session ends.

net.java.slee.resource.diameter.cxdx.CxDxServerSessionActivity
: This type of activity represents server side of Cx/Dx session. Registration-Termination-Request (RTR) and Push-Profile-Request (PPR) messages can be created and sent in this Activity, receiving the respective Answer (or timeout) later on this Activity. User-Authorization-Request (UAR), Server-Assignment-Request (SAR), Location-Info-Request (LIR) and Multimedia-Authentication-Request (MAR) messages are received in this Activity and respective Answers are sent from it.

  This activity type is implicitly created when receiving Requests or can be created with call to the proper `createCxDxServerSessionActivity` method of `net.java.slee.resource.diameter.cxdx.CxDxProvider` for sending Requests. It ends once underlying Cx/Dx session ends.

All activities define methods required to properly function and expose necessary information to JAIN SLEE services. Common part for both Client and Server Activity is defined as follows:

```
public String getSessionId();

public CxDxAVPFactory getCxDxAvpFactory();

public CxDxMessageFactory getCxDxMessageFactory();
```

public String getSessionId();
This method returns Session ID of underlying Diameter session.

public CxDxAVPFactory getCxDxAvpFactory();
This method returns an AVP factory capable of creating Diameter Cx/Dx related AVPs. It also provides access to Base AVP Factory.

public CxDxMessageFactory getCxDxMessageFactory();
This method returns a message factory capable of creating Diameter Cx/Dx messages. It also provides access to Base Message Factory.

Cx/Dx Client Activity is defined as follows:

```
public UserAuthorizationRequest createUserAuthorizationRequest();

public void sendUserAuthorizationRequest(
    UserAuthorizationRequest userAuthorizationRequest)
        throws IOException;

public ServerAssignmentRequest createServerAssignmentRequest();

public void sendServerAssignmentRequest(
    ServerAssignmentRequest serverAssignmentRequest)
        throws IOException;

public LocationInfoRequest createLocationInfoRequest();

public void sendLocationInfoRequest(LocationInfoRequest locationInfoRequest)
    throws IOException;

public MultimediaAuthenticationRequest createMultimediaAuthenticationRequest();

public void sendMultimediaAuthenticationRequest(
    MultimediaAuthenticationRequest multimediaAuthenticationRequest) throws IOException;

public RegistrationTerminationAnswer createRegistrationTerminationAnswer();

public void sendRegistrationTerminationAnswer(
    RegistrationTerminationAnswer registrationTerminationAnswer) throws IOException;

public PushProfileAnswer createPushProfileAnswer();
```

```
public void sendPushProfileAnswer(PushProfileAnswer pushProfileAnswer)
    throws IOException;
```

public UserAuthorizationRequest createUserAuthorizationRequest();

This method creates a User-Authorization-Request message pre-populated with the AVPs appropriate for this session.

public void sendUserAuthorizationRequest(UserAuthorizationRequest userAuthorizationRequest) throws IOException;

This method sends an event User-Authorization-Request. An event containing the answer will be fired on this activity.

public ServerAssignmentRequest createServerAssignmentRequest();

This method creates a ServerAssignmentRequest message pre-populated with the AVPs appropriate for this session.

public void sendServerAssignmentRequest(ServerAssignmentRequest serverAssignmentRequest) throws IOException;

This method sends an event Server-Assignment-Request. An event containing the answer will be fired on this activity.

public LocationInfoRequest createLocationInfoRequest();

This method creates a LocationInfoRequest message pre-populated with the AVPs appropriate for this session.

public void sendLocationInfoRequest(LocationInfoRequest locationInfoRequest) throws IOException;

This method sends an event Location-Info-Request. An event containing the answer will be fired on this activity.

public MultimediaAuthenticationRequest createMultimediaAuthenticationRequest();

This method creates a MultimediaAuthenticationRequest message pre-populated with the AVPs appropriate for this session.

public void sendMultimediaAuthenticationRequest(MultimediaAuthenticationRequest multimediaAuthenticationRequest) throws IOException;

This method sends an event Multimedia-Authentication-Request. An event containing the answer will be fired on this activity.

public RegistrationTerminationAnswer createRegistrationTerminationAnswer();

This method creates a RegistrationTerminationAnswer message pre-populated with the AVPs appropriate for this session.

public void sendRegistrationTerminationAnswer(RegistrationTerminationAnswer registrationTerminationAnswer) throws IOException;

This method sends an event Registration-Termination-Answer. An event containing the answer will be fired on this activity.

public void sendRegistrationTerminationAnswer(RegistrationTerminationAnswer registrationTerminationAnswer) throws IOException;

This method creates a Registration-Termination-Answer populated with the AVPs appropriate for this session.

public void sendPushProfileAnswer(PushProfileAnswer pushProfileAnswer) throws IOException;

This method sends an event Push-Profile-Answer in response to a Push-Profile-Request received on this activity.

Cx/Dx Server Activity is defined as follows:

```
public UserAuthorizationAnswer createUserAuthorizationAnswer();

public void sendUserAuthorizationAnswer(
    UserAuthorizationAnswer userAuthorizationAnswer)
        throws IOException;

public ServerAssignmentAnswer createServerAssignmentAnswer();

public void sendServerAssignmentAnswer(
    ServerAssignmentAnswer serverAssignmentAnswer)
        throws IOException;

public LocationInfoAnswer createLocationInfoAnswer();

public void sendLocationInfoAnswer(LocationInfoAnswer locationInfoAnswer)
    throws IOException;

public MultimediaAuthenticationAnswer createMultimediaAuthenticationAnswer();

public void sendMultimediaAuthenticationAnswer(
    MultimediaAuthenticationAnswer multimediaAuthenticationAnswer) throws IOException;

public RegistrationTerminationRequest createRegistrationTerminationRequest();

public void sendRegistrationTerminationRequest(
    RegistrationTerminationRequest registrationTerminationRequest) throws IOException;

public PushProfileRequest createPushProfileRequest();

public void sendPushProfileRequest(PushProfileRequest pushProfileRequest)
    throws IOException;
```

public UserAuthorizationAnswer createUserAuthorizationAnswer();

This method creates a User-Authorization-Answer message pre-populated with the AVPs appropriate for this session.

public void sendUserAuthorizationAnswer(UserAuthorizationAnswer userAuthorizationAnswer) throws IOException;

This method sends an event User-Authorization-Answer in response to a User-Authorization-Request received on this activity.

public ServerAssignmentAnswer createServerAssignmentAnswer();

This method creates a ServerAssignmentAnswer message pre-populated with the AVPs appropriate for this session.

public void sendServerAssignmentAnswer(ServerAssignmentAnswer serverAssignmentAnswer) throws IOException;

This method sends an event Server-Assignment-Answer in response to a Server-Assignment-Request received on this activity.

public LocationInfoAnswer createLocationInfoAnswer();

This method creates a LocationInfoAnswer message pre-populated with the AVPs appropriate for this session.

public void sendLocationInfoAnswer(LocationInfoAnswer locationInfoAnswer) throws IOException;

This method sends an event Location-Info-Answer in response to a Location-Info-Request received on this activity.

public MultimediaAuthenticationAnswer createMultimediaAuthenticationAnswer();

This method creates a MultimediaAuthenticationAnswer message pre-populated with the AVPs appropriate for this session.

public void sendMultimediaAuthenticationAnswer(MultimediaAuthenticationAnswer multimediaAuthenticationAnswer) throws IOException;

This method sends an event Multimedia-Authentication-Answer in response to a Multimedia-Authentication-Request received on this activity.

public RegistrationTerminationRequest createRegistrationTerminationRequest();

This method creates a RegistrationTerminationRequest message pre-populated with the AVPs appropriate for this session.

public void sendRegistrationTerminationRequest(RegistrationTerminationRequest registrationTerminationRequest) throws IOException;

This method sends an event Registration-Termination-Request. An event containing the answer will be fired on this activity.

public PushProfileRequest createPushProfileRequest();

This method creates a Push-Profile-Request pre-populated with the AVPs appropriate for this session.

public void sendPushProfileRequest(PushProfileRequest pushProfileRequest) throws IOException;

This method sends an event Push-Profile-Request. An event containing the answer will be fired on 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 Cx/Dx Resource Adaptor Type declares all the Diameter Cx/Dx Application specific events.

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

Table 2.1. Events received on Rf Server Activity

Name	Vendor	Version	Class
net.java.slee.resource.diameter.cxdx.events.UserAuthorizationRequest	java.net	0.8	net.java.slee.resource.diameter.cxdx.events.UserAuthorizationRequest
net.java.slee.resource.diameter.cxdx.events.ServerAssignmentRequest	java.net	0.8	net.java.slee.resource.diameter.cxdx.events.ServerAssignmentRequest
net.java.slee.resource.diameter.cxdx.events.LocationInfoRequest	java.net	0.8	net.java.slee.resource.diameter.cxdx.events.LocationInfoRequest
net.java.slee.resource.diameter.cxdx.events.MultimediaAuthenticationRequest	java.net	0.8	net.java.slee.resource.diameter.cxdx.events.MultimediaAuthenticationRequest
net.java.slee.resource.diameter.cxdx.events.RegistrationTerminationAnswer	java.net	0.8	net.java.slee.resource.diameter.cxdx.events.RegistrationTerminationAnswer
net.java.slee.resource.diameter.cxdx.events.PushProfileAnswer	java.net	0.8	net.java.slee.resource.diameter.cxdx.events.PushProfileAnswer

Table 2.2. Events received on Rf Client Activity

Name	Vendor	Version	Class
net.java.slee.resource. diameter.cxdx.events. UserAuthorizationAnswer	java.net	0.8	net.java.slee.resource. diameter.cxdx.events. UserAuthorizationAnswer
net.java.slee.resource. diameter.cxdx.events. ServerAssignmentAnswer	java.net	0.8	net.java.slee.resource. diameter.cxdx.events. ServerAssignmentAnswer
net.java.slee.resource. diameter.cxdx.events. LocationInfoAnswer	java.net	0.8	net.java.slee.resource. diameter.cxdx.events. LocationInfoAnswer
net.java.slee.resource. diameter.cxdx.events. MultimediaAuthenticationAnswer	java.net	0.8	net.java.slee.resource. diameter.cxdx.events. MultimediaAuthenticationAnswer
net.java.slee.resource. diameter.cxdx.events. RegistrationTerminationRequest	java.net	0.8	net.java.slee.resource. diameter.cxdx.events. RegistrationTerminationRequest
net.java.slee.resource. diameter.cxdx.events. PushProfileRequest	java.net	0.8	net.java.slee.resource. diameter.cxdx.events. PushProfileRequest

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 Cx/Dx Activity Context Interface Factory is defined as follows:

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

import javax.slee.ActivityContextInterface;

public interface CxDxActivityContextInterfaceFactory {

    public ActivityContextInterface getActivityContextInterface(
        CxDxClientSessionActivity cxdxcs);
```

```
    public ActivityContextInterface getActivityContextInterface(
        CxDxServerSessionActivity cxdxss);
}
```

2.4. Resource Adaptor Interface

The JBoss Communications Diameter Cx/Dx 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.cxdx;

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

public interface CxDxProvider {

    public CxDxMessageFactory getCxDxMessageFactory();

    public CxDxAVPFactory getCxDxAVPFactory();

    public CxDxClientSessionActivity createCxDxClientSessionActivity()
        throws CreateActivityException;

    public CxDxClientSessionActivity createCxDxClientSessionActivity(
        DiameterIdentity destinationHost, DiameterIdentity destinationRealm)
            throws CreateActivityException;

    public CxDxServerSessionActivity createCxDxServerSessionActivity()
        throws CreateActivityException;

    public CxDxServerSessionActivity createCxDxServerSessionActivity(
        DiameterIdentity destinationHost, DiameterIdentity destinationRealm)
            throws CreateActivityException;

    public DiameterIdentity[] getConnectedPeers();

    public int getPeerCount();
}
```

public CxDxMessageFactory getCxDxMessageFactory();

This method returns a message factory to be used to create concrete implementations of Cx/Dx messages.

public CxDxAVPFactory getCxDxAVPFactory();

This method returns a AVP factory to be used to create concrete implementations of Cx/Dx AVPs.

public CxDxClientSessionActivity createCxDxClientSessionActivity()throws CreateActivityException;

This method creates a new client session to send and receive Diameter messages. All messages sent on an activity created by this method must contain valid routing AVPs (one or both of Destination-Realm and Destination-Host as defined by RFC3588).

public CxDxClientSessionActivity createCxDxClientSessionActivity(DiameterIdentity destinationHost, DiameterIdentity destinationRealm) throws CreateActivityException;

This method creates a new client session to send and receive Diameter messages. Messages sent on an activity created by this method will automatically have the Destination-Host and Destination-Realm AVPs set to the provided values.

public CxDxServerSessionActivity createCxDxServerSessionActivity()throws CreateActivityException;

This method creates a new server session to send and receive Diameter messages. All messages sent on an activity created by this method must contain valid routing AVPs (one or both of Destination-Realm and Destination-Host as defined by RFC3588).

public CxDxServerSessionActivity createCxDxServerSessionActivity(DiameterIdentity destinationHost, DiameterIdentity destinationRealm) throws CreateActivityException;

This method creates a new server session to send and receive Diameter messages. Messages sent on an activity created by this method will automatically have the Destination-Host and Destination-Realm AVPs set to the provided values.

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

Example SBB code to handle communication with HSS look as follows:

```
public void onTimerEvent(TimerEvent event, ActivityContextInterface aci) {

    tracer.info("Diameter Cx/Dx example: Acting as client(ICSCF after REGISTER)");
    this.timerFacility.cancelTimer(event.getTimerID());

    // We are client - ICSCF, we get REGISTER from UE, instead of this timer methods
    // we should be in onRegister handler.
    DiameterIdentity destinationHost = new DiameterIdentity("aaa://" +
        getProperty("destination.ip") + ":" + getProperty("destination.port"));
    DiameterIdentity destinationRealm =
        new DiameterIdentity((String)getProperty("destination.realm"));

    CxDxClientSession clientSession;

    try {
        clientSession = this.provider.createCxDxClientSessionActivity(destinationHost,
            destinationRealm);

        ActivityContextInterface clientACI=this.acif.getActivityContextInterface(clientSession);
        clientACI.attach(this.sbbContext.getSbbLocalObject());

        UserAuthorizationRequest UAR = clientSession.createUserAuthorizationRequest();
        // < User-Authorization-Request> ::= < Diameter Header: 300, REQ, PXY, 16777216 >
        //         < Session-Id >
        //         { Vendor-Specific-Application-Id }
        //         { Auth-Session-State }
        UAR.setAuthSessionState(AuthSessionStateType.STATE_MAINTAINED);
        //         { Origin-Host }
        //         { Origin-Realm }
        //         [ Destination-Host ]
        //         { Destination-Realm }
        //         { User-Name }
        UAR.setUserName("adam.b");
        //         *[ Supported-Features ]
        //         { Public-Identity }
        UAR.setPublicIdentity("sip:adam.b@travel.contanct.com");
        //         { Visited-Network-Identifier }
        UAR.setVisitedNetworkIdentifier("visit.airport.moscow.ru");
        //         [ User-Authorization-Type ]
        UAR.setUserAuthorizationType(UserAuthorizationType.DE_REGISTRATION);
        //         [ UAR-Flags ]
```

```
            //          *[ AVP ]
            //          *[ Proxy-Info ]
            //          *[ Route-Record ]

            // Send the message
            tracer.info("Diameter Cx/Dx example: Sending UAR:\n"+UAR);
            clientSession.sendUserAuthorizationRequest(UAR);
            tracer.info("Diameter Cx/Dx example: Sent UAR:\n"+UAR);
        }
        catch (CreateActivityException e) {
            tracer.severe("Unable to create Cx/Dx Client Session Activity.", e);
        }
        catch (IOException e) {
            tracer.severe("Failed to send Cx/Dx User-Authorization-Request.", e);
        }
    }

    ...

    public void onUserAuthorizationAnswer(UserAuthorizationAnswer event,
        ActivityContextInterface aci) {
        tracer.info("Diameter Cx/Dx example: receveid UAA:\n"+event);

        if(event.getResultCode() / 1000 != 2) {
            tracer.info("Diameter Cx/Dx example: receveid UAA with wrong result code:\n" +
                event.getResultCode());
            return;
        }

          tracer.info("Diameter Cx/Dx example: Acting as SCSCF after receiving REGISTER from
ICSCF");
        // HSS responded to our call, here we would forward REGISTER to S-CSCF,
        // however lets not do that, its easier to set up it like that
        // lets act as S-CSCF, which received REGISTER and sends SAR
        try{
            //Detach, this will die
            aci.detach(this.sbbContext.getSbbLocalObject());
            DiameterIdentity destinationHost = new DiameterIdentity("aaa://" +
                getProperty("destination.ip") + ":" + getProperty("destination.port"));
            DiameterIdentity destinationRealm =
                new DiameterIdentity((String)getProperty("destination.realm"));

            //create SAR/SAA activity
            CxDxClientSession clientSession = this.provider.
```

```
            createCxDxClientSessionActivity(destinationHost, destinationRealm);
        ActivityContextInterface clientACI = this.acif.
            getActivityContextInterface(clientSession);
        clientACI.attach(this.sbbContext.getSbbLocalObject());
        ServerAssignmentRequest SAR = clientSession.createServerAssignmentRequest();

        // <Server-Assignment-Request> ::= < Diameter Header: 301, REQ, PXY, 16777216 >
        //     < Session-Id >
        //     { Vendor-Specific-Application-Id }
        //     { Auth-Session-State }
        SAR.setAuthSessionState(AuthSessionStateType.STATE_MAINTAINED);
        //     { Origin-Host }
        //     { Origin-Realm }
        //     [ Destination-Host ]
        //     { Destination-Realm }
        //     [ User-Name ]
        SAR.setUserName("adam.b");
        //     *[ Supported-Features ]
        //     *[ Public-Identity ]
        SAR.setPublicIdentity("sip:adam.b@travel.contanct.com");
        //     [ Wildcarded-PSI ]
        //     [ Wildcarded-IMPU ]
        //     { Server-Name }
        SAR.setServerName("x123.s.cscf.local.domain.org");
        //     { Server-Assignment-Type }
        SAR.setServerAssignmentType(ServerAssignmentType.USER_DEREGISTRATION);
        //     { User-Data-Already-Available }
        SAR.setUserDataAlreadyAvailable(
            UserDataAlreadyAvailable.USER_DATA_NOT_AVAILABLE);
        //     [ SCSCF-Restoration-Info ]
        //     [ Multiple-Registration-Indication ]
        //     *[ AVP ]
        //     *[ Proxy-Info ]
        //     *[ Route-Record ]

        // Send the SAR.
        tracer.info("Diameter Cx/Dx example: Sending SAR:\n"+SAR);
        clientSession.sendServerAssignmentRequest(SAR);
        tracer.info("Diameter Cx/Dx example: Sent SAR:\n"+SAR);
    }
    catch (CreateActivityException e) {
        tracer.severe("Unable to create Cx/Dx Client Session Activity.", e);
    }
    catch (IOException e) {
```

```
            tracer.severe("Failed to send Cx/Dx User-Authorization-Request.", e);
        }
    }
```

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	10415:16777216, 13019:16777216

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

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

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

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 Cx/Dx 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 2.4.0.CR1.

```
[usr]$ svn co ?/2.4.0.CR1 slee-ra-diameter-cx-dx-2.4.0.CR1
```

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-cx-dx-2.4.0.CR1
[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 Cx/Dx 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 Cx/Dx 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 Cx/Dx RA implementation does not support clustering.

Appendix A. Revision History

Revision History
Revision 1.0 Wed Feb 10 2010 AlexandreMendonça
Creation of the JBoss Communications JAIN SLEE Diameter Cx/Dx RA User Guide.

Index

F

feedback, viii