SIP Servlet Example Applications

Table of Contents

The SIP Servlet Server has a selection of examples that demonstrate particular capabilities of the server. Available Examples lists the available examples, their location, and a brief description about the functionality each example demonstrates. The examples can also provide a useful starting point for developing SIP Applications, therefore it is encouraged to experiment and adapt the base examples. Each example is available in both binary and source formats.

Table 1. Available Examples

Example	Description
Call Blocking	Demonstrates how to block calls by specifying that the INVITE SIP Extension checks the From address to see if it is specified in the block list. If the blocked SIP address matches, the Call Blocking application send a FORBIDDEN response.
Call Forwarding	Demonstrates how to forward calls by specifying that the INVITE SIP Extension checks the To address to see if it is specified in the forward list. If the SIP address matches, the application acts as a back-to-back user agent (B2BUA).
Call Controller	Call Blocking and Call Forwarding are merged to create a new service.
Speed Dial	Demonstrates how to implement speed dialing for SIP addresses. The demonstration uses a static list of speed dial numbers. The numbers are translated into a complete address based on prior configuration. The SIP addresses are proxied without record-routing, or supervised mode.
Location Service	Demonstrates a location service that performs a lookup based on the request URI, into a hard-coded list of addresses. The request is proxied to the set of destination addresses associated with that URI.

Example	Description
Composed Speed Dial and Location	Speed Dial and Location are merged to create a new service. Speed Dial proxies the speed dial number to a SIP address, then Location Service proxies the call to the actual location of the call recipient.
Click to Call	Demonstrates how SIP Servlets can be used along with HTTP servlets as a converged application to place calls from a web portal. The example is a modified version of the click to dial example from the Sailfin project, but has been reworked to comply with JSR 289.
Chat Server	Demonstrates MESSAGE SIP Extension support. This example is based on the chatroom server demonstration from the BEA dev2dev project, and has been modified to meet JSR 289 requirements.
Media JSR 309 Example	Demonstrates how the Sip Servlets Application Developers can leverage the JSR-309 API, which provides to application developers multimedia capabilities with a generic media server (MS) abstraction interface. This example is only compatible with JBoss AS5. The solution is know to work with Twinkle and linphone SIP soft-phones.
Shopping	Demonstrates integration with Seam and Java Enterprise Edition (JEE), and JSR 309 Media integration with text to speech (TTS) and dual-tone multi-frequency (DTMF) tones. The demonstration builds on the Converged Demo example, and adds support for the SIP Servlets v1.1 specification.

Example	Description
Diameter Event Charging Service	Demonstrates how the Diameter Event Charging, and the Location service, can be used to perform fixed-rated charging of calls (event charging). When a call is initiated, a debit of ten euros is applied to the A Party account. If the call is rejected by the B Party, or A Party hangs up before B Party can answer the call, the ten euro charge is credited to the A Party account.
Diameter Sh OpenIMS Integration	Demonstrates the integration between RestComm and OpenIMS, using the Diameter Sh interface to receive profile updates and SIP.
Diameter Ro/Rf IIntegration	A Diameter Ro/Rf service that performs online call charging.
Conference	Demonstrates the capabilities of the Media Server, such as endpoint composition and conferencing, as well as proving that SIP Servlets are capable of working seamlessly with any third-party web framework, without repackaging or modifying the deployment descriptors. The demonstration uses Google's GWT Ajax framework with server-push updates to provide a desktop-like user interface experience and JSR 309 for Media Control.
Alerting Application	This application was developed so that the JBoss RHQ/Jopr Enterprise Management Solution would be able to notify system administrators when a monitoring alert is fired by Jopr/RHQ.

Example	Description
SIP Presence Client Application	A Call Blocking application interoperating with the PLATFORM_NAME; SIP Presence Service (Technology Preview) to fetch the blocked contacts through XCAP.

Operating the Example Applications

Important Information

Before trying out the examples in this section, you must have installed, configured and have Restcomm for JBoss or Restcomm for Tomcat AS7 running on your system.

See the chapters below for detailed instructions.

[_getting_started_with_mss_jboss_as7]

[_getting_started_with_mss_tomcat_as7]

The Location Service

The Restcomm Location Service contains a list of mappings of request URIs to destination addresses. When the Location Service receives a request, it performs a lookup on that mapping and proxies the request simultaneously to the destination address (or addresses) associated with that URI.

The Location Service Mappings Cannot Currently Be Configured

The Location Service currently performs a lookup on a hard-coded list of addresses. This model is evolving toward the eventual use of a database.

Regardless of whether you are using the JBoss Application Server or the Tomcat Servlet Container as the Servlets Server, the application, container and Location Service perform the following steps:

- A user—let us call her Alice—makes a call to `sip:receiver@sip-servlets.com`. The `INVITE` is received by the servlet container, which then starts the Location Service.
- The Location Service, using non-SIP means, determines that the callee (i.e. the receiver) is registered at two locations, identified by the two SIP URIs, `sip:receiver@127.0.0.1:5090` and `sip:receiver@127.0.0.1:6090`.
- The Location Service proxies to those two destinations in parallel, without record-routing, and without making use of supervised mode.
- One of the destinations returns a `200 OK` status code; the second proxy is then canceled.
- The `200 OK` is forwarded to Alice, and call setup is completed as usual.

Here is the current list of hard-coded contacts and their location URIs:

- .sip:receiver@sip-servlets.com `sip:receiver@127.0.0.1:5090`
- `sip:receiver@127.0.0.1:6090`

Downloading

The Location Service is comprised of two archive files, a Web Archive (WAR) and a Default Application Router (DAR) configuration file, which you need to add to your installed SIP Servlets Server. For more information about WAR files, refer to the JBoss Application Server Administration and Development Guide. For more information about DAR files, refer to the JSR 289 spec, Appendix C.

Download the Location Service’s WAR file from here: &VERSION;&VERSION;https://oss.sonatype.org/content/groups/public/org/mobicents/servlet/sip/examples/location-service//location-service-.war.

Download the Location Service's DAR file from here: https://sipservlets.googlecode.com/git/sip-servlets-examples/location-service/locationservice-dar.properties.

Installing

Both the *location-service-.war* WAR file and the *locationservice-dar.properties* DAR file that you downloaded should be placed into different directories in your SIP Servlet Server installation hierarchy. Which directory depends on whether you are using the Location Service with Restcomm for JBoss or with Restcomm for Tomcat:

Restcomm for JBoss AS7

Place *location-service-.war* into the *$JBOSS_HOME/standalone/deployments/* directory, and *locationservice-dar.properties* into the [path]_$JBOSS_HOME/standalone/configuration/dars/ _ directory.

Restcomm for Tomcat AS7

Place *location-service-.war* into the *$CATALINA_HOME/webapps/* directory, and *locationservice-dar.properties* into the *$CATALINA_HOME/conf/dars/* directory.

Configuration

Restcomm for JBoss

Open the $JBOSS_HOME/standalone/configuration/standalone-sip.xml configuration file and find the mobicents subsystem element.

Example 1. Editing MSS for JBoss's standalone-sip.xml for the Location Service

In the $JBOSS_HOME/standalone/configuration/standalone-sip.xml file search for the line

```
 <subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/mobicents-dar.properties"
```

and replace it with the line below

```
<subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/locationservice-dar.properties"
```

Restcomm for Tomcat

Open the *$CATALINA_HOME/conf/server.xml* configuration file and find the `Service` element. Add an attribute to it called `darConfigurationFileLocation`, and set it to `conf/dars/locationservice-dar.properties`:

Example 2. Editing MSS for Tomcat's server.xml for the Location Service

In the $JBOSS_HOME/standalone/configuration/standalone-sip.xml file search for the line

```
 <subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/mobicents-dar.properties"
```

and replace it with the line below

```
<subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/locationservice-dar.properties"
```

Running

Once the WAR and DAR files have been placed in the right directories, and the JBoss Application Server or Tomcat Servlet Container knows where to find them (which you specified in the *standalone-sip.xml* and *server.xml* file), then you should go ahead and run the SIP Servlets Server.

Testing

The following procedure shows how to test the Location Service.

Procedure:

1. Start two SIP soft-phones. The first phone should be set up as `sip:receiver@sip-servlets.com` at the IP address `127.0.0.1` on port `5090`. The second phone can be set up in any way you like. Note that the SIP phones do not have to be registered.
2. Using the second phone, make a call to `sip:receiver@sip-servlets.com`. If the Location Service

has been set up correctly and is running, the first phone—as the receiver or callee—should now be ringing.

The Diameter Event-Changing Service

The Diameter Event-Changing Service is based on the Location Service, which performs call-charging at a fixed rate. Upon the initiation of a call, a debit of €10.00 occurs. In the cases of a call being rejected or the caller disconnecting (hanging up) before an answer is received, the caller's account is refunded.

Note that an Restcomm for JBoss installation is required to run this example; it will not work with Restcomm for Tomcat.

Provided here is a step-by-step description of the procedure as performed by the application and container:

Procedure: Diameter Event-Changing Service Step-By-Step

1. A user, Alice, makes a call to `sip:receiver@sip-servlets.com`. The `INVITE` is received by the servlet container, which sends a request to debit Alice's account to the Charging Server. The servlet container then invokes the location service.
2. The Location Service determines, without using the SIP protocol itself, where the callee—or receiver—is registered. The callee may be registered at two locations identified by two SIP URIs: `sip:receiver@127.0.0.1:5090` and `sip:receiver@127.0.0.1:6090`.
3. The Location Service proxies to those two destinations simultaneously, without record-routing and without using the supervised mode.
4. One of the destinations returns `200 (OK)`, and so the container cancels the other.
5. The `200 (OK)` is forwarded upstream to Alice, and the call setup is carried out as usual.
6. If none of the registered destinations accepts the call, a Diameter Accounting-Request for refund is sent to the Diameter Charging Server in order to debit the already-credited €10.00

Diameter Event-Changing Service: Installing, Configuring and Running

Preparing your Restcomm for JBoss server to run the Diameter Event-Changing example requires downloading a WAR archive, a DAR archive, the Ericsson Charging Emulator, setting an attribute in JBoss's *standalone-sip.xml* configuration file, and then running JBoss AS. Detailed instructions in the section below.

Pre-Install Requirements and Prerequisites

The following requirements must be met before installation can begin.

Software Prerequisites

One Restcomm for JBoss Installation

Before proceeding, you should follow the instructions for installing, configuring, running and testing Restcomm for JBoss from the binary distribution.

Downloading

The following procedure describes how to download the required files.

1. First, download the latest Web Application Archive () file corresponding to this example, the current version of which is named diameter-event-charging-*.war, from &VERSION;&VERSION;https://oss.sonatype.org/content/groups/public/org/mobicents/servlet/sip/examples/diameter-event-charging//diameter-event-charging-.war.
2. Secondly, download the corresponding Disk Archive () configuration file here: https://sipservlets.googlecode.com/git/sip-servlets-examples/diameter-event-charging/diametereventcharging-dar.properties.
3. Finally, you will need to download the Ericsson Charging Emulator, version 1.0, from http://mobicents.googlecode.com/files/ChargingSDK-1_0_D31E.zip.

Installing

The following procedure describes how to install the downloaded files.

1. Place the *diameter-event-charging-.war* WAR archive into the *$JBOSS_HOME/standalone/deployments/* directory.
2. Place the *diametereventcharging-dar.properties* DAR file in your $JBOSS_HOME/standalone/configuration/dars/ directory.
3. Finally, open the terminal, move into the directory to which you downloaded the Ericsson Charging SDK (for the sake of this example, we will call this directory *charging_sdk*), and then unzip the downloaded zip file (you can use Java's `jar -xvf` command for this:

```
~]$ cd charging_sdk
charging_sdk]$ jar -xvf ChargingSDK-1_0_D31E.zip
```

Alternatively, you can use Linux's `unzip` command to do the dirty work:

```
charging_sdk]$ unzip ChargingSDK-1_0_D31E.zip
```

Configuration

Restcomm for JBoss

Open the $JBOSS_HOME/standalone/configuration/standalone-sip.xml configuration file and find the mobicents subsystem element.

Example 3. Editing the standalone-sip.xml for the Diameter Event-Changer Service

In the $JBOSS_HOME/standalone/configuration/standalone-sip.xml file search for the line

```
 <subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/mobicents-dar.properties"
```

and replace it with the line below

```
<subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/diametereventcharging-dar.properties"
```

Running

The following procedure describes how to run the Diameter Event-Changing Service.

Procedure: Diameter Event-Changing Service

1. Then, run the Ericsson Charging Emulator. Open a terminal, change the working directory to the location of the unzipped Charging Emulator files (in *ChargingSDK-1_0_D31E* or a similarly-named directory), and run it with the `java -jar PPSDiamEmul.jar` command:

   ```
   ~]$ java -jar PPSDiamEmul.jar
   ```

Using

Using the Event-Changing service means, firstly, inserting some parameters into the Charging Emulator, and then, by using two SIP (soft)phones, calling one with the other. The following sequential instructions show you how.

SIP (Soft)Phone? Which?

The Restcomm team recommends one of the following SIP phones, and has found that they work well: the 3CX Phone, the SJ Phone or the WengoPhone.

Procedure: Using the Diameter Event-Changing Service

1. Configure the Ericsson SDK Charging Emulator

   Once you have started the Charging Emulator, you should configure it exactly as portrayed in [_figure_mss_chargingemulatorconfig].

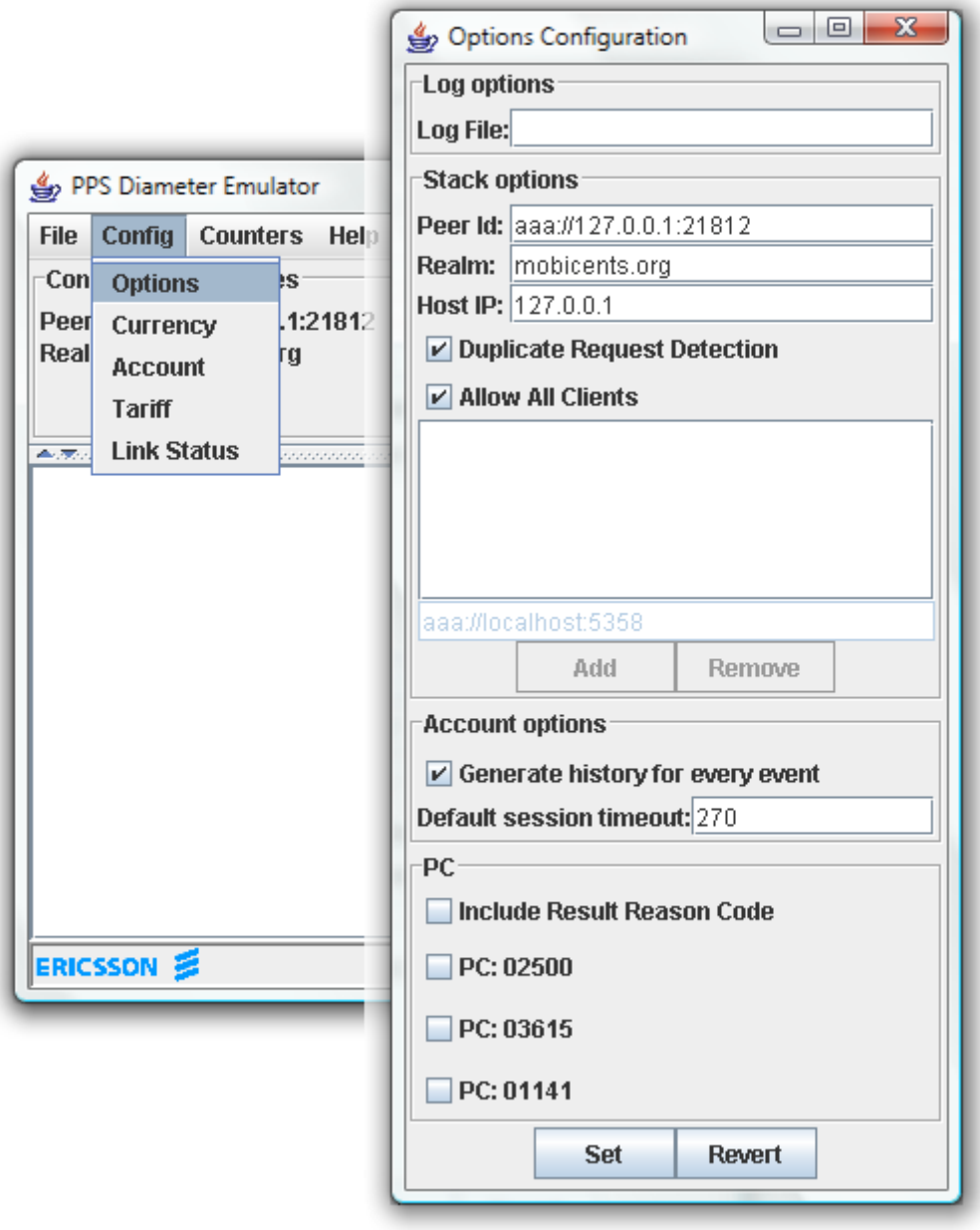

Figure 1. Configuring the Charging Emulator

2. Set the `Peer ID` to: `aaa://127.0.0.1:21812`
3. Set the `Realm` to: `mobicents.org`
4. Set the `Host IP` to: `127.0.0.1`
5. Start two SIP (soft)phones. You should set the first phone up with the following parameters: `sip:receiver@sip-servlets` on IP address `127.0.0.1` on port `5090`. The other phone can be set up any way you like.
6. Before making a call, open the menu:Config[Options] dialog window, as shown in the image.

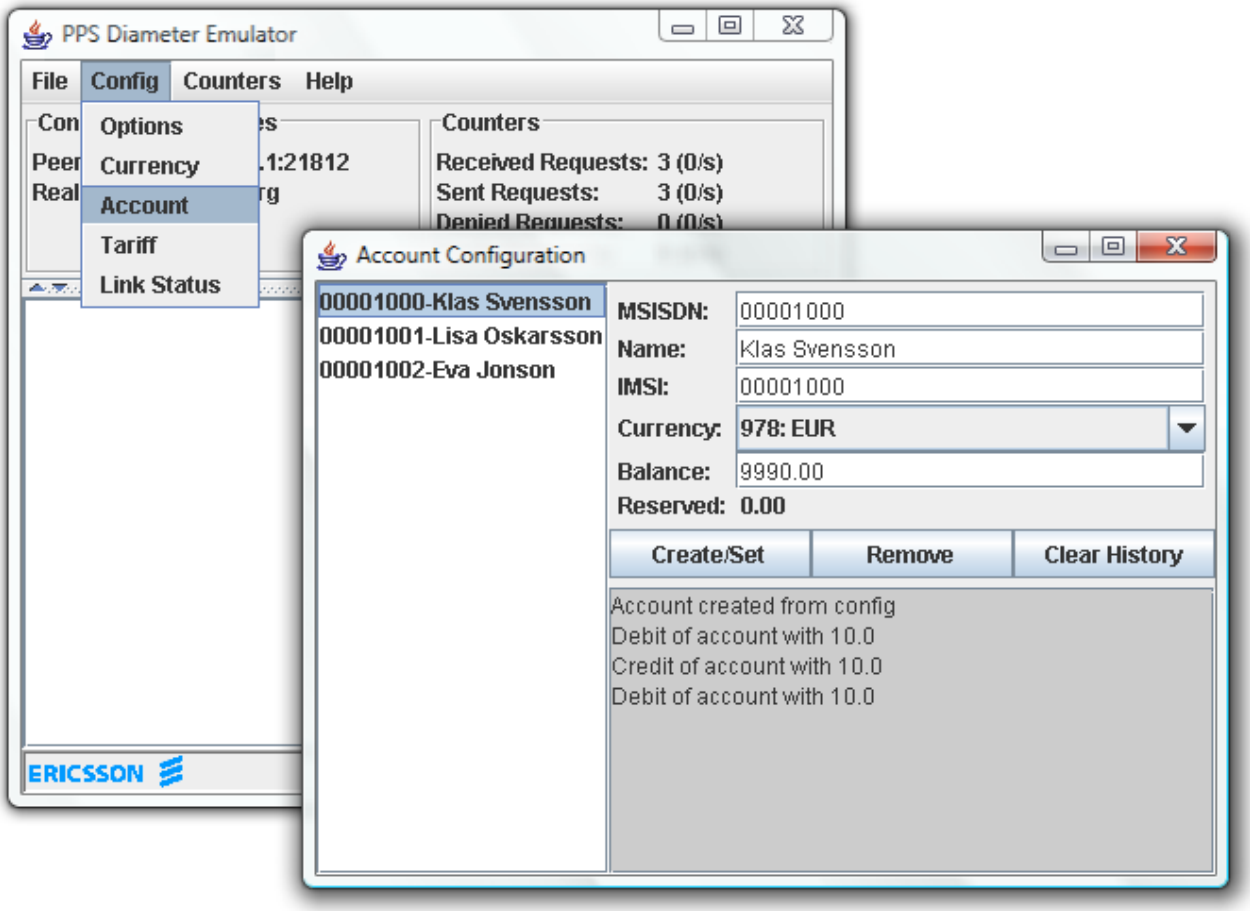

Figure 2. Configuring Accounts in the Charging Emulator

In the Account Configuration window of the Charging Emulator, you can see the user's balances. Select a user to watch the balance. You can also stretch the window lengthwise to view the user's transaction history.

7. Time to call! From the second, "any-configuration" phone, make a call to `sip:receiver@sip-servlets.com`. Upon doing so, the other phone should ring or signal that it is being contacted .
8. You should be able to see a request—immediately following the invite and before the other party (i.e. you) accepts or rejects the call—sent to the Charging Emulator. That is when the debit of the user's account is made. In the case that the call is rejected, or the caller gives up, a second, new Diameter request is sent to refund the initial amount charged by the call. On the other hand, if the call is accepted, nothing else related to Diameter happens, and no second request takes place.

   Please note that this is not the correct way to do charging, as Diameter provides other means, such as unit reservation. However, for the purpose of a demonstration it is sufficient to show the debit and follow-up credit working. Also, this is a fixed-price call, regardless of the duration. Charging can, of course, be configured so that it is time-based.

The Call-Blocking Service

The Restcomm Call-Blocking Service, upon receiving an `INVITE` request, checks to see whether the sender's address is a blocked contact. If so, it returns a `FORBIDDEN` reply; otherwise, call setup proceeds as normal.

Blocked Contacts Cannot Currently Be Configured

Blocked contacts are currently hard-coded addresses. This model is evolving towards the eventual use of a database.

Here is the current hard-coded list of blocked contacts:

- `sip:blocked-sender@sip-servlets.com`
- `sip:blocked-sender@127.0.0.1`

The Call-Blocking Service: Installing, Configuring and Running

Software Prerequisites

Either an Restcomm for JBoss or an Restcomm for Tomcat Installation

The Call-Blocking Service requires either an Restcomm for JBoss or an Restcomm for Tomcat binary installation.

Downloading

The Call-Blocking Service is comprised of two archive files, a Web Archive (WAR) and a Default Application Router (DAR) configuration file, which you need to add to your installed SIP Servlets Server. For more information about WAR files, refer to the JBoss Application Server Administration and Development Guide. For more information about DAR files, refer to the JSR 289 spec, Appendix C.

Download the Call-Blocking Service’s WAR file from here: &VERSION;&VERSION;https://oss.sonatype.org/content/groups/public/org/mobicents/servlet/sip/examples/call-blocking//call-blocking-.war.

Download the Call-Blocking Service's DAR file from here: https://sipservlets.googlecode.com/git/sip-servlets-examples/call-blocking/call-blocking-servlet-dar.properties.

Installing

Both the *call-blocking-.war* WAR file and the *call-blocking-servlet-dar.properties* DAR file that you downloaded should be placed into different directories in your SIP Servlet Server installation hierarchy. Which directory depends on whether you are using the Call-Blocking Service with Restcomm for JBoss or with Restcomm for Tomcat:

Restcomm for JBoss

Place *call-blocking-.war* into the *$JBOSS_HOME/standalone/deployments/* directory, and *call-blocking-servlet-dar.properties* into the *$JBOSS_HOME/standalone/configuration/dars/* directory.

Restcomm for Tomcat

Place *call-blocking-servlet-dar.properties* into the *$CATALINA_HOME/webapps/* directory, and *call-blocking-servlet-dar.properties* into the *$CATALINA_HOME/conf/dars/* directory.

Configuring

Restcomm for JBoss

Open the $JBOSS_HOME/standalone/configuration/standalone-sip.xml configuration file and find the mobicents subsystem element.

Example 4. Editing MSS for JBoss's standalone-sip.xml for the Location Service

In the $JBOSS_HOME/standalone/configuration/standalone-sip.xml file search for the line

```
 <subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/mobicents-dar.properties"
```

and replace it with the line below

```
<subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/call-blocking-servlet-dar.properties"
```

Restcomm for Tomcat

Open the *$CATALINA_HOME/conf/server.xml* configuration file and find the `Service` element. Add an attribute to it called `darConfigurationFileLocation`, and set it to `conf/dars/call-blocking-servlet-dar.properties`:

Example 5. Editing MSS for Tomcat's server.xml for the Location Service

In the $JBOSS_HOME/standalone/configuration/standalone-sip.xml file search for the line

```
 <subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/mobicents-dar.properties"
```

and replace it with the line below

```
<subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/call-blocking-servlet-dar.properties"
```

Running

Once the WAR and DAR files have been placed in the right directories, and the JBoss Application Server or Tomcat Servlet Container knows where to find them (which you specified in a *server.xml* and the *standalone-sip.xml* files), then you should go ahead and run the SIP Servlets Server.

Testing

The following procedure shows how to test the Call-Blocking Service.

Procedure: Testing the Call Blocking Service

1. Start a SIP softphone of your choice. The account name should be `blocked-sender`. The `From Header` should list one of the following addresses: `sip:blocked-sender@sip-servlets.com` or `sip:blocked-sender@127.0.0.1`. The SIP softphone does not need to be registered.
2. Make a call to any address, and you should receive a `FORBIDDEN` response.

The Call-Forwarding Service

The Restcomm Call-Forwarding Service, upon receiving an `INVITE` request, checks to see whether the sender's address is among those in a list of addresses which need to be forwarded. If so, then the Call-Forwarding Service acts as a Back-to-Back User Agent (B2BUA), and creates a new call leg to the destination. When the response is received from the new call leg, it sends it an acknowledgment (`ACK`) and then responds to the original caller. If, on the other hand, the server does not receive an `ACK`, then it tears down the new call leg with a `BYE`. Once the `BYE` is received, then it answers `OK` directly and sends the `BYE` to the new call leg.

Contacts to Forward Cannot Currently Be Configured

Contacts to forward are currently hard-coded addresses. This model is evolving toward the eventual use of a database.

Here is the current hard-coded list of contacts to forward:

- `sip:receiver@sip-servlets.com`
- `sip:receiver@127.0.0.1`

The Call-Forwarding Service: Installing, Configuring and Running

Pre-Install Requirements and Prerequisites

The following requirements must be met before installation can begin.

Downloading

The Call-Forwarding Service is comprised of two archive files, a Web Archive (WAR) and a Data Archive (DAR), which you need to add to your installed SIP Servlets Server. For more information about WAR and DAR files, refer to the JBoss Application Server Administration and Development Guide.

Download the Call-Forwarding Service’s WAR file from here: &VERSION;&VERSION;https://oss.sonatype.org/content/groups/public/org/mobicents/servlet/sip/examples/call-forwarding//call-forwarding-.war.

Download the Call-Forwarding Service's DAR file from here: https://sipservlets.googlecode.com/git/sip-servlets-examples/call-forwarding/call-forwarding-b2bua-servlet-dar.properties.

Installing

Both the *call-forwarding-.war* WAR file and the *call-forwarding-servlet-dar.properties* DAR file that you downloaded should be placed into different directories in your SIP Servlet Server installation hierarchy. Which directory depends on whether you are using the Call-Forwarding Service with Restcomm for JBoss or with Restcomm for Tomcat:

Restcomm for JBoss

Place *call-forwarding-.war* into the *$JBOSS_HOME/standalone/deployments/* directory, and *call-forwarding-servlet-dar.properties* into the *$JBOSS_HOME/standalone/configuration/dars/* directory.

Restcomm for Tomcat

Place *call-forwarding-.war* into the *$CATALINA_HOME/webapps/* directory, and *call-forwarding-servlet-dar.properties* into the *$CATALINA_HOME/conf/dars/* directory.

Configuring

Restcomm for JBoss

Open the $JBOSS_HOME/standalone/configuration/standalone-sip.xml configuration file and find the mobicents subsystem element.

Example 6. Editing MSS for JBoss's standalone-sip.xml for the Location Service

In the $JBOSS_HOME/standalone/configuration/standalone-sip.xml file search for the line

```
 <subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/mobicents-dar.properties"
```

and replace it with the line below

```
<subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/call-forwarding-b2bua-servlet.properties"
```

Restcomm for Tomcat

Open the *$CATALINA_HOME/conf/server.xml* configuration file and find the `Service` element. Add an attribute to it called `darConfigurationFileLocation`, and set it to `conf/dars/call-forwarding-b2bua-servlet-dar.properties`:

Example 7. Editing MSS for Tomcat's server.xml for the Location Service

In the $JBOSS_HOME/standalone/configuration/standalone-sip.xml file search for the line

```
 <subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/mobicents-dar.properties"
```

and replace it with the line below

```
<subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/call-forwarding-b2bua-servlet-dar.properties"
```

Running

Once the WAR and DAR files have been placed in the right directories, and the JBoss Application Server or Tomcat Servlet Container knows where to find them (which you specified in a *standalone-sip.xml* and *server.xml* files), then you should go ahead and run the SIP Servlets Server.

Testing

The following procedure shows how to test the Call-Forwarding Service.

Procedure:

1. Start two SIP soft-phones of your choice. Set the account settings of the first SIP softphone to:
   - Account name: `forward-receiver`
   - IP address: `127.0.0.1`
   - Port: `5090`

   Neither of the SIP soft-phones needs to be registered.

2. From the second phone, make a call to `sip:receiver@sip-servlets.com`. The first phone, "forward-receiver", should now be ringing.

The Call-Controller Service

The Call-Controller service is a composition of two other services: Call-Blocking and Call-Forwarding. Essentially, it performs the services of both call-forwarding and call-blocking.

- To learn about how the Call-Blocking service works, refer to The Call-Blocking Service.
- To learn about how the Call-Forwarding service works, refer to The Call-Forwarding Service.

Blocked Contacts and Contacts to Forward Cannot Currently Be Configured

Both the list of blocked contacts and the list of contacts to forward are currently both hard-coded. However, both of those models are evolving toward the eventual use of databases.

The Call-Controller Service: Installing, Configuring and Running

The Call-Controller service requires the two WAR files for the Call-Blocking and Call-Forwarding services to be placed in the correct directory inside your Restcomm SIP Servlets Server binary installation. However, the Call-Controller service does *not* require their corresponding DAR files: you need only to download and install a DAR file customized for the Call-Controller service. The instructions below show you how to do precisely this; there is no need, therefore, to first install either the Call-Blocking or the Call-Forwarding services, though it is helpful to at least be familiar with them.

Pre-Install Requirements and Prerequisites

The following requirements must be met before installation can begin.

Downloading

The Call-Controller Service is comprised of two WAR files, one for the Call-Forwarding service and one for Call-Blocking, and a customized Call-Controller DAR file. You do not need to install the DAR files for the Call-Forwarding or the Call-Blocking services. For more information about WAR files, refer to the JBoss Application Server Administration and Development Guide. For more information about DAR files, refer to the JSR 289 spec, Appendix C

Download the Call-Blocking Service’s WAR file from here: &VERSION;&VERSION;https://oss.sonatype.org/content/groups/public/org/mobicents/servlet/sip/examples/call-blocking//call-blocking-.war.

Download the Call-Forwarding Service’s WAR file from here: &VERSION;&VERSION;https://oss.sonatype.org/content/groups/public/org/mobicents/servlet/sip/examples/call-forwarding//call-forwarding-.war.

Download the Call-Controller Service's DAR file from here: https://sipservlets.googlecode.com/git/sip-servlets-examples/call-blocking/call-controller-servlet-dar.properties.

Installing

The *call-blocking-.war*, *call-forwarding-.war* and *call-controller-servlet-dar.properties* archive files that you downloaded should be placed into different directories in your SIP Servlet Server installation hierarchy. Which directory depends on whether you are using the Call-Controller Service with Restcomm for JBoss or with Restcomm for Tomcat:

Restcomm for JBoss

Place *call-blocking-.war* and *call-forwarding-.war* into the *$JBOSS_HOME/standalone/deployments/* directory, and *call-controller-servlet-dar.properties* into the *$JBOSS_HOME/standalone/configuration/dars/* directory.

Restcomm for Tomcat

Place *call-blocking-.war* and *call-forwarding-.war* into the *$CATALINA_HOME/webapps/* directory,

and *call-controller-servlet-dar.properties* into the *$CATALINA_HOME/conf/dars/* directory.

Configuring

RRestcomm for JBoss

Open the $JBOSS_HOME/standalone/configuration/standalone-sip.xml configuration file and find the mobicents subsystem element.

Example 8. Editing MSS for JBoss's standalone-sip.xml for the Location Service

In the $JBOSS_HOME/standalone/configuration/standalone-sip.xml file search for the line

```
 <subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/mobicents-dar.properties"
```

and replace it with the line below

```
<subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/call-forwarding-b2bua-servlet.properties"
```

Restcomm for Tomcat

Open the *$CATALINA_HOME/conf/server.xml* configuration file and find the `Service` element. Add an attribute to it called `darConfigurationFileLocation`, and set it to `conf/dars/call-controller-servlet-dar.properties`:

Example 9. Editing MSS for Tomcat's server.xml for the Location Service

In the $JBOSS_HOME/standalone/configuration/standalone-sip.xml file search for the line

```
 <subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/mobicents-dar.properties"
```

and replace it with the line below

```
<subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0"
application-router="dars/call-controller-servlet-dar.properties"
```

Running

Once the WAR and DAR files have been placed in the right directories, and the JBoss Application Server or Tomcat Servlet Container knows where to find them (which you specified in a *server.xml* file), then you should go ahead and run the SIP Servlets Server.

Testing

Two use-cases can be distinguished for the Call-Controller service: one in which a call is blocked,

and another in which a call is forwarded. Therefore, we have two cases for which we can test the Call-Controller.

Procedure: Blocking a Call with Call-Controller

1. Start two SIP soft-phones of your choice. Set the account settings of the SIP soft-phones to:
   - .Relevant First Softphone SettingsAccount name: `forward-receiver`
   - IP address: `127.0.0.1`
   - Port: `5090`
   - .Relevant Second Softphone SettingsAccount name: `blocked-sender`

   Neither of the SIP soft-phones needs to be registered.

2. From the second phone, `blocked-sender`, make a call to `sip:receiver@sip-servlets.com`. You should receive a `FORBIDDEN` response.

Procedure: Forwarding a Call with Call-Controller

1. Start two SIP soft-phones of your choice. Set the account settings of the SIP soft-phones to:
   - .Relevant First Softphone SettingsAccount name: `forward-receiver`
   - IP address: `127.0.0.1`
   - Port: `5090`
   - .Relevant Second Softphone SettingsAccount name: `forward-sender`

   Neither of the SIP soft-phones needs to be registered.

2. From the second softphone, `forward-sender`, make a call to `sip:receiver@sip-servlets.com`. The first phone, `forward-receiver`, should now be ringing.

[_ssea_sip_servlet_example_applications] provides more information about other service examples available.

Click To Call