Configuring and Running Restcomm JAIN SLEE

Table of Contents

Server Profiles

Restcomm JAIN SLEE reuses &JEE.PLATFORM;server profiles to expose different configurations for different needs:

Default Profile

> The `default` profile is proper for standalone or pure high availability. It provides the best performance per cluster node, with linear scaling, but there is no state replication in the cluster, which means that there is no support for failover, neither there is any kind of state gravitation (one node sending state so another node continues its work.

All Profile

> The `all` profile is proper for more flexible high availability and failover support. Performance per node decreases but the cluster does state replication or gravitation.

> Profiles can be selected when starting the server, see Running Restcomm JAIN SLEE for detailed instructions.

Running Restcomm JAIN SLEE

Starting or stopping Restcomm JAIN SLEE is no different than starting or stopping &JEE.PLATFORM;

Starting

Once installed, you can run server(s) by executing the run.sh (Unix) or run.bat (Microsoft Windows) startup scripts in the *<install_directory>/bin* directory (on Unix or Windows).

Starting Parameters

Server Profile

> To specify the server profile use `-c profile_name`, for instance, to use the `all` profile then start the server with `-c all`

> ❗ | If not specified the default profile is used.

IP / Host

> To specify the IP/Host which the server binds, use `-b IP`, for instance, to use the 192.168.0.1 IP then start the server with `-b 192.168.0.1`

> ❗ | If not specified then 127.0.0.1 is used.

Stopping

You can shut down the server(s) by executing the `shutdown.sh -s` (Unix) or `shutdown.bat -s` (Microsoft Windows) scripts in the *<install_directory>/bin* directory (on Unix or Windows). Note that if you properly stop the server, you will see the following three lines as the last output in the

Unix terminal or Command Prompt:

```
[Server] Shutdown complete
Shutdown complete
Halting VM
```

Configuring Restcomm JAIN SLEE

JAIN SLEE is configured through an XML descriptor for each Server Profiles. The XML file is named *jboss-beans.xml* and is located at *$JBOSS_HOME/server/profile_name/deploy/restcomm-slee/META-INF*, where `profile_name` is the server profile name.

> ⚠️ This configuration greatly affects performance or correctness of the container behavior. This is for advanced users that know the internals of the container.

EventContext Factory Configuration

The EventContext Factory is responsible for managing all EventContexts in the SLEE Container, and its behavior is configurable.

The EventContext Factory configuration can be changed through an XML file and a JMX MBean.

EventContext Factory Persistent Configuration

Configuration is done through an XML descriptor for each Restcomm Cluster. The XML file is named *jboss-beans.xml* and is located at *{JBOSS_HOME}/server/{profile_name}/deploy/restcomm-slee/META-INF*

The configuration is exposed a JBoss Microcontainer Bean:

```xml
<bean name="Mobicents.JAINSLEE.EventContextFactoryConfiguration"

class="org.mobicents.slee.container.management.jmx.EventContextFactoryConfiguration">
    <annotation>@org.jboss.aop.microcontainer.aspects.jmx.JMX(name=
        "org.mobicents.slee:name=EventContextFactoryConfiguration",exposedInterface=

org.mobicents.slee.container.management.jmx.EventContextFactoryConfigurationMBean.class,
        registerDirectly=true)</annotation>
    <property name="defaultEventContextSuspensionTimeout">60000</property>
</bean>
```

Table 1. EventContext Factory Bean Configuration

Property Name	Property Type	Description
defaultEventContextSuspension Timeout	int	defines the default timeout applied when suspending delivery of an EventContext

EventContext Factory JMX Configuration

Through JMX the EventContext Factory module configuration can be changed with the container running. Note that such configuration changes are not persisted.

The JMX MBean which can be used to change the EventContext Factory configuration is named `org.mobicents.slee:name=EventContextFactoryConfiguration`, and provides getters and setters to change each property defined in the persistent configuration. The JMX Console can be used to use this MBean, see [_management_jmx_console].

Event Router Statistics and Configuration

The JAIN SLEE Event Router is the module responsible for creating new service instances and delivering events to all interested parties. It is capable of doing the routing of several events in parallel, through the usage of multiple executors, each bound to a different thread.

The Event Router is also able to account performance and load statistics, indicating the number of activities being assigned or several timings regarding event routing, globally or for each individual executor/thread. Statistics are turned on by default and may be retrieved through the JMX MBean `org.mobicents.slee:name=EventRouterStatistics`.

An important sub-module of the Event Router is the Executor Mapper, which is responsible for assigning activities to the available executors. JAIN SLEE includes two different Executor Mappers. The default one takes into account the hashcode of the activity handle when distributing, while the alternative uses a round robin algorithm.

> In the case of advanced performance tuning, it is advised to try the different implementations available, or even provide a custom one.

The Executor Mapper is nothing more than an interface: `org.mobicents.slee.container.eventrouter.EventRouterExecutorMapper`. To deploy a custom implementation, drop the implementation class or classes, packed in a jar file, in the server profile /deploy directory.

The whole Event Router is a critical component with respect to the container's performance. Its configuration can be tuned, through an XML file and a JMX MBean.

Event Router Persistent Configuration

Configuration is done through an XML descriptor for each Server Profiles. The XML file is named jboss-beans.xml and is located at $JBOSS_HOME/server/profile_name/deploy/restcomm-slee/META-

INF, where `profile_name` is the server profile name.

The configuration is exposed a JBoss Microcontainer Bean:

```
<bean name="Mobicents.JAINSLEE.EventRouterConfiguration"
    class="org.mobicents.slee.container.management.jmx.EventRouterConfiguration">
    <annotation>@org.jboss.aop.microcontainer.aspects.jmx.JMX(name=
        "org.mobicents.slee:name=EventRouterConfiguration", exposedInterface=

org.mobicents.slee.container.management.jmx.EventRouterConfigurationMBean.class,
        registerDirectly=true)</annotation>
    <property name="eventRouterThreads">8</property>
    <property name="collectStats">true</property>
    <property name="executorMapperClassName">

org.mobicents.slee.runtime.eventrouter.mapping.ActivityHashingEventRouterExecutorMappe
r
    </property>
</bean>
```

Table 2. JAIN SLEE Event Router Bean Configuration

Property Name	Property Type	Description
eventRouterThreads	int	defines how many executors should be used by the Event Router, each bounds to a different thread
collectStats	boolean	defines if performance and load statistics should be collected, turning this feature off will increase performance
confirmSbbEntityAttachement	boolean	defines if the event router should reconfirm that sbb entities are attached to activity context, before delivering event, this will avoid that a sbb entity handles concurrent events after it detachs, turning this feature off will increase performance

Property Name	Property Type	Description
executorMapperClassName	Class	This property defines the implementation class of Executor Mapper used by the Event Router, the one above and default uses the activity handle hashcode to do the mapping, an alternative is org.mobicents.slee.runtime.eventrouter .mapping.RoundRobinEvent RouterExecutorMapper, which uses Round Robin algorithm.

Event Router JMX Configuration

Through JMX, the Event Router module configuration can be changed while the container is running. These configuration changes are not persisted.

The JMX MBean that can be used to change the Event Router configuration is named `org.mobicents.slee:name=EventRouterConfiguration`, and provides getters and setters to change each property defined in the persistent configuration. See [_management_jmx_console] for how the JMX Console can be used to use this MBean.

Timer Facility Configuration

The JAIN SLEE Timer Facility is the module responsible for managing SLEE timers, and the number of threads it uses is configurable.

The Timer Facility configuration can be changed through an XML file and a JMX MBean.

Timer Facility Persistent Configuration

Configuration is done through an XML descriptor for each Restcomm Cluster. The XML file is named *jboss-beans.xml* and is located at *{JBOSS_HOME}/server/{profile_name}/deploy/restcomm-slee/META-INF*

The configuration is exposed a JBoss Microcontainer Bean:

```
<bean name="Mobicents.JAINSLEE.TimerFacilityConfiguration"
    class="org.mobicents.slee.container.management.jmx.TimerFacilityConfiguration">
    <annotation>@org.jboss.aop.microcontainer.aspects.jmx.JMX(name=
        "org.mobicents.slee:name=TimerFacilityConfiguration",exposedInterface=

org.mobicents.slee.container.management.jmx.TimerFacilityConfigurationMBean.class,
        registerDirectly=true)</annotation>
    <property name="timerThreads">4</property>
</bean>
```

Table 3. JAIN SLEE Timer Facility Bean Configuration

Property Name	Property Type	Description
timerThreads	int	defines how many threads should be used by the Timer Facility
purgePeriod	int	defines the period (in minutes) of purging canceled tasks from the Timer Facility. Use 0 for no purge at all.

Timer Facility JMX Configuration

Through JMX the Timer Facility module configuration can be changed with the container running. Note that such configuration changes are not persisted.

The JMX MBean which can be used to change the Timer Facility configuration is named org.mobicents.slee:name=TimerFacilityConfiguration, and provides getters and setters to change each property defined in the persistent configuration. The JMX Console can be used to use this MBean, see [_management_jmx_console].

Configuring JAIN SLEE Profiles

JAIN SLEE Profiles is a component used to store data, usually related with a user and/or service profile. JAIN SLEE maps JAIN SLEE Profiles to a Java Persistence API () Datasource, through Hibernate.

There are two configurations for JAIN SLEE Profiles provided as JBoss Microcontainer Beans:

```xml
<bean name="Mobicents.JAINSLEE.Profiles.JPA.HSQLDBConfig"
    class="org.mobicents.slee.container.deployment.profile.jpa.Configuration">
    <property name="persistProfiles">true</property>
    <property name="clusteredProfiles">false</property>
    <property name="hibernateDatasource">java:/DefaultDS</property>
    <property name="hibernateDialect">org.hibernate.dialect.HSQLDialect</property>
    <depends>jboss.jca:service=DataSourceBinding,name=DefaultDS</depends>
</bean>
<bean name="Mobicents.JAINSLEE.Profiles.JPA.PostgreSQLConfig"
    class="org.mobicents.slee.container.deployment.profile.jpa.Configuration">
    <property name="persistProfiles">true</property>
    <property name="clusteredProfiles">true</property>
    <property name="hibernateDatasource">java:/PostgresDS</property>
    <property name="hibernateDialect">
org.hibernate.dialect.PostgreSQLDialect</property>
</bean>
```

❗ Configurations can be changed, or new ones can be added. For new ones, ensure that the name attribute of the bean element is unique.

Table 4. JAIN SLEE Profiles Bean Configuration

Property Name	Property Type	Description
persistProfiles	boolean	If true, profile changes are persisted into the data source.
clusteredProfiles	boolean	If true, then the container is aware there is a shared data source and that updates may be done by other nodes (for example, deletion of a JAIN SLEE profile table).
hibernateDatasource	String	The name of the Java Datasource deployed in the JBoss Application Server.
hibernateDialect	String	The java class name of the hibernate dialect to use, related with the selected datasource.

To switch the active configuration simply change the parameter injected in the bean named `Mobicents.JAINSLEE.Container`.

Other Configurations

Other JAIN SLEE runtime configuration is done through the following JBoss Microcontainer Bean:

```
<bean name="Mobicents.JAINSLEE.MobicentsManagement"
    class="org.mobicents.slee.container.management.jmx.MobicentsManagement">
    <annotation>@org.jboss.aop.microcontainer.aspects.jmx.JMX(
        name="org.mobicents.slee:service=MobicentsManagement",
        exposedInterface=org.mobicents.slee.container.management.
        jmx.MobicentsManagementMBean.class,
        registerDirectly=true)</annotation>
    <property name="entitiesRemovalDelay">1</property>
    <property name="timerThreads">8</property>
    <property name="loadClassesFirstFromAS">true</property>
</bean>
```

Table 5. Other JAIN SLEE Configurations

Property Name	Property Type	Description
entitiesRemovalDelay	int	The number of minutes before the container forces the ending of SBB entities from a service being deactivated.
timerThreads	int	The number of threads used by the timer facility.

This configuration can be changed with the container running with JMX. Note that such configuration changes are not persisted.

To change the configuraton, use the JMX MBean named org.mobicents.slee:service=MobicentsManagement, which provides getters and setters to change each property defined in the persistent configuration that is configurable with the container running. The JMX Console can be used to use this MBean, as described in [_management_jmx_console].

Logging Configuration

Logging configuration is documented in section [_global_logging_config]

Congestion Control Configuration

Congesture Control feature configuration is documented in section [_congestion_control_configuration]