Concurrency and Congestion Control

Table of Contents

Concurrency and Congestion control refer to settings that define the way in which messages are processed under heavy load. The way Restcomm SIP Servlets Server processes messages in a production environment is crucial to ensure quality of service for customers.

Concurrency control mode tuning affects the way in which the SIP Servlets Server processes messages, whereas Congestion Control tuning affects the point at which the server begins rejecting new requests. Both of these parameters can be set using the following methods:

- Through the SIP Servlets Management Console.
- Editing the server's *server.xml* or *standalone-sip.xml* configuration file.
- From the `dispatcher` MBean.
- From the Embedded Jopr integrated management platform.

Concurrency Control

The JSR 289 expert group does not specify how concurrency control should be implemented.

Restcomm SIP Servlets for JBoss and Restcomm SIP Servlets for Tomcat have concurrency control implemented as a configurable mode, which defines the way in which the SIP Servlets Server processes messages.

It has to be noted that this concurrency control mechanism is not cluster aware and will work per node only, it is not a cluster wide lock.

The following modes are provided, and cater for the particular setup required in an implementation:

None

All SIP messages are processed as soon as possible in a thread from the global thread pool.

Transaction

Bypass the SIP Servlets request/response executors, and utilize the JAIN SIP built-in Transaction serialization to manage race conditions on the same transaction.

SipSession

SIP messages are processed as soon as possible except for messages originating from the same `SipSession`. These messages are excluded from any simultaneous processing.

SipApplicationSession

SIP messages are processed as soon as possible, with the guarantee that no two messages from the same `SipSession` *or* from the same `SipApplicationSession` will ever be processed simultaneously. Of all the available methods, this mode is the best choice for guaranteed thread-safety.

Congestion Control

Restcomm Sip Servlets currently provides the following congestion control mechanisms:

Changing Congestion Control Settings

All the settings and congifurations starting with gov.nist.javax.sip are located in the $JBOSS-HOME/standalone/configuration/mss-sip-stack.properties file. The section below will provide further details.

- Congestion control is largely application-specific and it is implemented in a pipeline. First messages arrive in the JAIN SIP message queue where they will wait on the locks needed before they can be processed. To avoid keeping too many messages in the queue and potentially running out of memory, older messages are discarded without any error indication. This prevents spam and flood DoS attacks to accumulate large backlog and render the server unresponsive. It also guranatees flood recovery time of 20 seconds or less, in the mean time retransmissions are already queuing so that normal SIP calls can continue without dropping them. After the request has passed the first queue it enters the SIP transaction layer where there is a customizable optional congestion control logic. There is one packaged congestion control algorithm which can be enabled by setting the following property `gov.nist.javax.sip.SIP_MESSAGE_VALVE=gov.nist.javax.sip.stack.CongestionControlMessageValve`. For this algorithm you can set the limit value by the following property `gov.nist.javax.sip.MAX_SERVER_TRANSACTIONS=2000`. You can also implement your own algorithm and change the class name in `gov.nist.javax.sip.SIP_MESSAGE_VALVE` to activate it.

  There is also another optional legacy congestion control stage with another queue where messages can be discarded based on dynamic parameters such as available JVM heap memory or number of messages in the queue. This method will be deprecated and is not recommended. All SIP messages which cannot be processed immediately are put into a queue, and wait for either a free thread or for the lock on their session to be released. The size of the SIP message queue is a tunable parameter, which defaults to `1500`.

- If the SIP Message queue becomes full, the container immediately begins rejecting new SIP requests until the queue clears. This is achieved by using one of the following methods:
  - Sending a `503` SIP error code to the originating application.
  - Dropping incoming messages (according to the specified congestion control policy).
- If the container exceeds the configurable memory threshold (90% by default), new SIP requests are rejected until the memory usage falls below the specified memory threshold. This is achieved by using one of the following methods:
  - Sending a `503` SIP error code to the originating application.
  - Dropping incoming messages (according to the specified congestion control policy).

A background task gathers information about the current server congestion. The data collection interval can be adjusted, and congestion control deactivated, by setting the interval to 0 or a negative value.

The congestion control policy defines how an incoming message is handled when the server is overloaded. The following parameters are configurable:

- DropMessage - drop any incoming message

- ErrorResponse - send a 503 - Service Unavailable response to any incoming request (Default).

Configuring the Concurrency and Congestion Control Settings

The concurrency and congestion control settings can be configured through the SIP Servlets Management Console, using the following methods:

- Through the SIP Servlets Management Console.
- Editing the server's *server.xml* or the *standalone-sip.xml* configuration file.
- From the `dispatcher` MBean.
- From the Embedded Jopr integrated management platform.

Tuning Parameters with the SIP Servlets Management Console

The easiest way to configure the SIP Message Queue Size and Concurrency Control Mode tunable parameters is to open the `SIP Servlets Management Console` in your browser (by going to http://localhost:8080/sip-servlets-management), making your changes, and then clicking button **Apply**.

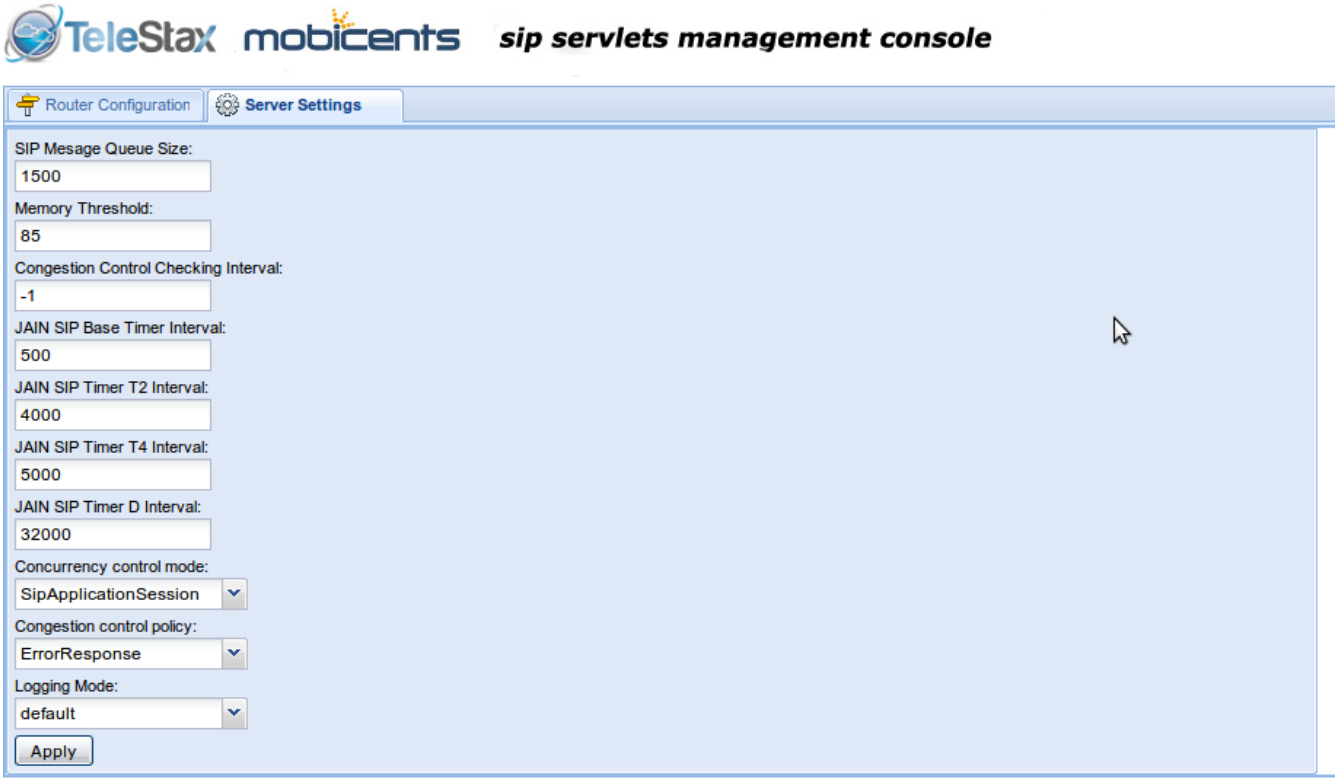

Figure 1. SIP Servlets Management Console Concurrency and Congestion Control Tuning Parameters

Concurrency and congestion control settings altered through the SIP Servlets Management Console are not saved to the server.xml on Tomcat, only on JBoss AS7 through the standalone-sip.xmlconfiguration file. To make settings persistent, append the settings to the server.xml file directly.

Making your changes permanent in standalone-sip.xml or server.xml by manual editing

Alternatively, you can edit your server's *standalone-sip.xml* or *server.xml* configuration file, which has the benefit of making your chosen settings changes permanent for Tomcat.

Instructions follow, grouped by the SIP Servlets Server you are running:

Procedure: Tuning RestComm SIP Servlets for JBoss Server Settings for Concurrency and Congestion Control

1. Open standalone-sip.xml File

   Open the $JBOSS_HOME/standalone/configuration/standalone-sip.xml configuration file in a text editor.

2. Extract from stanalone-sip.xml file with conccurency configuration

```
<subsystem xmlns="urn:org.mobicents:sip-servlets-as7:1.0" application-
router="dars/mobicents-dar.properties" stack-properties="mss-sip-stack.properties"
path-name="gov.nist" app-dispatcher-
class="org.mobicents.servlet.sip.core.SipApplicationDispatcherImpl" concurrency-
control-mode="SipApplicationSession" congestion-control-interval="-1">
    <connector name="sip-udp" protocol="SIP/2.0" scheme="sip" socket-binding="sip-
udp"/>
    <connector name="sip-tcp" protocol="SIP/2.0" scheme="sip" socket-binding="sip-
tcp"/>
    <connector name="sip-tls" protocol="SIP/2.0" scheme="sip" socket-binding="sip-
tls"/>
    <connector name="sip-tls" protocol="SIP/2.0" scheme="sip" socket-binding="sip-
ws"/>
    <connector name="sip-tls" protocol="SIP/2.0" scheme="sip" socket-binding="sip-
wss"/>
</subsystem>
```

Procedure: Tuning RestComm SIP Servlets for Tomcat Server Settings for Concurrency and Congestion Control

1. Open server.xml File

   Open the $CATALINA_HOME/conf/server.xml configuration file in your text editor.

2. Add Parameters to <service> Element

   Locate the <service> element, and add the concurrencyControlMode and/or sipMessageQueueSize attributes.

   Possible values for the concurrencyControlMode attribute include: None, SipSession or SipApplicationSession. SipSession is the value of this attribute when it is not present—and overridden—in server.xml.

3. Define the Correct Attribute Values

   The following default values for the concurrency and congestion control parameters are used regardless of whether the attributes are defined in the server.xml file:

   - sipMessageQueueSize="1500"

- backToNormalSipMessageQueueSize="1300"
- congestionControlCheckingInterval="30000" (30 seconds, in milliseconds)
- memoryThreshold="95" (in percentage)
- backToNormalMemoryThreshold="90" (in percentage)
- congestionControlPolicy="ErrorResponse"

Experimentation is required for these tuning parameters depending on the operating system and server.

Tuning Parameters from the dispatcher MBean

Navigate to the `dispatcher` MBean from Restcomm SIP Servlets for JBoss's JMX console.

All changes performed at run time are effective immediately, but do not persist across reboots for Tomcat, only on JBoss AS7. the server.xml must be appended with the settings in order to make the configuration persistent.

When editing the dispatcher MBean from RestComm SIP Servlets for JBoss's JMX console, values allowed for the concurrency control mode are **None**, **SipSession** or **SipApplicationSession**.