

Monitoring

# Table of Contents

- View SMSC Statistics ..... 1
  - Using CLI ..... 1
  - Using GUI ..... 2
- CDR Log..... 3

# View SMSC Statistics

## Using CLI

You can view the current state of SMSC using the command `smsc stat get` with appropriate parameters as described below:

### Name

`smsc stat get`

### SYNOPSIS

`smsc stat get`

### DESCRIPTION

This command is used to view the details of the current state of the SMSC and monitor the SMSC. The output prints the following parameters:

**Time** - Current time. By obtaining this statistic twice, this value can be used to calculate the time interval between statistic time.

**MessageInProgress** - Number of messages currently being processed for delivery in GSM (MT messages) and SMPP (messages that are routed to ESME) and SIP (messages that are routed to ESME).

**MessageId** - This is the last assigned Message Id. This indicates the number of messages that have come into the SMSC from GSM (Mobile Originated messages), from ESMEs or from SIP and stored in the Cassandra database. The MessageId counter is initiated from the time of the installation of the SMSC.

**MessageScheduledTotal** - The number of messages put into the delivering process in both GSM, SMPP or SIP since the SMSC was started.

**DueSlotProcessingLag** - The time (in seconds) between "in time" due\_slot and "in process" due\_slot. "In time" means the current actual time, "in process" means that SMSC GW is processing messages that was scheduled for that time (in the past).

If this value is equal to 0 or 1 or 2, it means that the SMSC is not highly loaded and all the messages are being processed on time.

If this value is high, say for example 300, then it means the SMSC is overloaded and is now processing messages, that are scheduled for processing 300 seconds before the current time.

If this value is progressively increasing, then the SMSC is heavily overloaded and the incoming messages count at SMPP (and MO) are more than what the SMSC can deliver.

If the incoming messages are not many, this value will decrease and will reach 0 when there are no messages.

Normally this value will return to 0 except for few peaks. If it does not, then you must reduce the load for SMSC.

`DueSlotProcessingTime` - This field shows the time for which SMSC GW is processing / fetching stored in cassandra database messages ("in process" `due_slot`). If this time is far before the current actual time, then this means:

- either SMSC GW is overloaded and can not send messages in time
- or SMSC GW was turned off for much time and now is checking for messages for the time when it was off

If you want to skip unsent messages that was scheduled for the time in the past (and then shift `DueSlotProcessingTime` to to current time) - you can use the command [the reference to CLI / GUI command "smc skip-unsent-messages" - see the Chapter "Skipping of scheduled for the past and not yet sent messages ("In process" `due_slot` shifting)]."

`Param1` - Ignore.

`Param2` - Ignore.

`SmscStartTime` - The time when the SMSC was started.

#### EXAMPLES

```
smc stat get
```

```
Stat: Time: Mon Jan 13 18:40:42 CET 2014, MessageInProgress: 515, MessageId: 10212,  
MessageScheduledTotal: 8992, DueSlotProcessingLag: 0, Param1: 0, Param2: 0,  
SmscStartTime: Mon Jan 13 18:39:30 CET 2014
```

## Using GUI

### *Procedure: View SMSC Statistics using the GUI*

1. In the GUI Management Console for SMSC Gateway, click on 'Stats' in the left panel.
2. The main panel will display the current statistics of the SMSC and display the details of the parameters 'START TIME', 'CURRENT TIME', 'TOTAL MESSAGES SCHEDULED', 'MESSAGES IN PROCESS', 'CURRENT MESSAGE ID' and 'DUE SLOT PROCESSING LAG'. For more details of these parameters please refer to the description of the CLI command in the preceding section.

This page gets auto-refreshed every 2.5 seconds and therefore the statistics get refreshed automatically.

# CDR Log

Restcomm SMSC is configured to generate CDR in a plain text file located at *restcomm-smscgateway-<version>/jboss-5.1.0.GA/server/<profile>/log/cdr.log*. The CDR generated in the text file is of the below format:

```
SIMBIT_DATE,ADDR_SRC_DIGITS,ADDR_SRC_TON,ADDR_SRC_NPI,ADDR_DST_DIGITS,ADDR_DST_TON,ADDR_DST_NPI,Message_Delivery_Status,ORIG_SYSTEM_ID,MESSAGE_ID,NNN_DIGITS,IMSI,CORR_ID,First 20 characters of SMS, Reason_For_Failure
```

where *ADDR\_SRC\_DIGITS*, *ADDR\_SRC\_TON*, *ADDR\_SRC\_NPI*, *ADDR\_DST\_DIGITS*, *ADDR\_DST\_TON*, *ADDR\_DST\_NPI*, *ORIG\_SYSTEM\_ID*, *MESSAGE\_ID*, *NNN\_DIGITS*, *IMSI* and *CORR\_ID* are as explained in [\[slot\\_messages\\_table\\_yyyy-mm-dd\]](#); *SIMBIT\_DATE*, *Message\_Delivery\_Status* and *Reason\_For\_Failure* are described below in this section.



*NNN\_DIGITS* and *IMSI* fields are present only in the case of SS7 terminated messages when there is a SRI positive response. *CORR\_ID* is present only if a message has come to the SMSC Gateway via "home-routing" procedure.

## *SIMBIT\_DATE*

Time when the message reached the SMSC Gateway.

## *Message\_Delivery\_Status*

The CDR text file contains a special field, *Message\_Delivery\_Status*, that specifies the message delivery status. The possible values are described below:

### *Message\_Delivery\_Status if delivering to GSM network:*

#### *partial*

Delivered a part of a multi-part message but not the last part.

#### *success*

Delivered the last part of a multi-part message or a single message.

#### *temp\_failed*

Failed delivering a part of a multi-part message or a single message. It does not indicate if a resend will be attempted or not.

#### *failed*

Failed delivering a message and the SMSC will now attempt to resend the message or part of the message.

#### *failed\_imsi*

Delivery process was broken by a mproc rule applying at the step when a successful SRI response has been received from HLR.

### *Message\_Delivery\_Status if delivering to ESME:*

#### *partial\_esme*

Delivered a part of a multi-part message but not the last part.

#### *success\_esme*

Delivered the last part of a multi-part message or a single message.

#### *temp\_failed\_esme*

Failed delivering a part of a multi-part message or a single message.

#### *failed\_esme*

Failed delivering a message and the SMSC will now attempt to resend the message or part of the message.

#### *Message\_Delivery\_Status if delivering to SIP:*

##### *partial\_sip*

Delivered a part of a multi-part message but not the last part.

##### *success\_sip*

Delivered the last part of a multi-part message or a single message.

##### *temp\_failed\_sip*

Failed delivering a part of a multi-part message or a single message.

##### *failed\_sip*

Failed delivering a message and the SMSC will now attempt to resend the message or part of the message.

#### *Message\_Delivery\_Status if the message has been rejected by the OCS Server (Diameter Server):*

##### *ocs\_rejected*

OCS Server rejected an incoming message.

#### *Message\_Delivery\_Status if the message has been rejected by a mproc rule applying at the step when a message has been arrived to SMSC GW:*

##### *mproc\_rejected*

A mproc rule rejected an incoming message (and reject response was sent to a message originator).

##### *mproc\_dropped*

A mproc rule dropped an incoming message (and accept response was sent to a message originator).

. The last field in the CDR generated is **Reason\_For\_Failure**, which records the reason for delivery failure and is empty if the delivery is successful. The possible delivery failure cases are explained below.

#### *Reasons\_For\_Failure*

##### *XXX response from HLR*

A MAP error message is received from HLR after SRI request; XXX: `AbsentSubscriber`, `AbsentSubscriberSM`, `CallBarred`, `FacilityNotSupported`, `SystemFailure`, `UnknownSubscriber`, `DataMissing`, `UnexpectedDataValue`, `TeleserviceNotProvisioned`.

*Error response from HLR: xxx*

Another MAP error message is received from HLR after SRI request.

*Error XXX after `MtForwardSM` Request*

A MAP error message is received from MSC/VLR after `MtForwardSM` request; XXX: `subscriberBusyForMtSms`, `absentSubscriber`, `absentSubscriberSM`, `smDeliveryFailure`, `systemFailure`, `facilityNotSup`, `dataMissing`, `unexpectedDataValue`, `facilityNotSupported`, `unidentifiedSubscriber`, `illegalSubscriber`.

*Error after `MtForwardSM` Request: xxx*

Another MAP error message is received from MSC/VLR after `MtForwardSM` request.

*DialogClose after `MtRequest`*

No `MtForwardSM` response and no error message received after `MtForwardSM` request.

*`onDialogProviderAbort` after `MtForwardSM` Request*

MAP `DialogProviderAbort` is received after `MtForwardSM` request.

*`onDialogProviderAbort` after SRI Request*

MAP `DialogProviderAbort` is received after SRI request.

*Error condition when invoking `sendMtSms()` from `onDialogReject()`*

After a `MtForwardSM` request MAP version conflict, MAP message negotiation was processed but this process failed, or other fundamental MAP error occurred.

*`onDialogReject` after SRI Request*

After a SRI request MAP version conflict, MAP message negotiation was processed but this process failed, or other fundamental MAP error occurred.

*`onDialogTimeout` after `MtForwardSM` Request*

Dialog timeout occurred after `MtForwardSM` Request. The reason may be GSM network connection failure or SMSC overload.

*`onDialogTimeout` after SRI Request*

Dialog timeout occurred after SRI Request. The reason may be GSM network connection failure or SMSC overload.

*`onDialogUserAbort` after `MtForwardSM` Request*

`DialogUserAbort` message is received from a peer or sent to a peer. The reason may be GSM fundamental failure or SMSC overload.

*`onDialogUserAbort` after SRI Request*

`DialogUserAbort` message is received from a peer or sent to a peer. The reason may be GSM fundamental failure or SMSC overload.

**onRejectComponent** *after MtForwardSM Request*

Reject component was received from a peer or sent to a peer. This is an abnormal case and implies MAP incompatibility.

**onRejectComponent** *after SRI Request*

Reject component was received from a peer or sent to a peer. This is an abnormal case and implies MAP incompatibility.

*Other*

Any other message that usually indicates some internal failure.