

## OsmoMSC - Bug #2895

### OsmoMSC excessively uses LU Reject Cause 22 (congestion)

01/28/2018 12:18 AM - laforge

<b>Status:</b>	New	<b>Start date:</b>	01/28/2018
<b>Priority:</b>	Normal	<b>Due date:</b>	
<b>Assignee:</b>	osmith	<b>% Done:</b>	0%
<b>Category:</b>			
<b>Target version:</b>			
<b>Resolution:</b>		<b>Spec Reference:</b>	
<b>Description</b>			
<p>For some reason OsmoMSC almost always claims that a LU Reject is due to congestion, which is simply not true. Like, let's say there is no intersection between the network-permitted and MS-supported encryption algorithms. <b>clearly</b> we are not rejecting due to congestion in such a case.</p> <p>Aside being confusing when looking at protocol traces, there are strong reasons for proper reject causes. Congestion is a temporary condition, i.e. the MS will continue to re-try to register. The lack of matching encryption algorithms is a permanent condition, and hence a permanent LU reject cause should be used to prevent the MS from any retries.</p>			

#### History

#1 - 02/08/2018 10:12 AM - googlegravity

#2 - 05/17/2018 01:58 PM - laforge

- Assignee changed from sysmocom to stsp

#3 - 12/11/2018 11:07 AM - stsp

- Status changed from New to In Progress

I could identify the following cases where osmo-msc sends a LU reject with cause CONGESTION:

Case 1: A location update is already in progress while another one arrives:

```
930 static int assoc_lfp_with_sub(struct osmo_fsm_inst *fi, struct vlr_subscr *vsub)
931 {
932     struct lu_fsm_priv *lfp = lu_fsm_priv(fi);
933     struct vlr_instance *vlr = lfp->vlr;
934
935     if (vsub->lu_fsm) {
936         LOGPFSML(fi, LOGL_ERROR,
937                 "A Location Updating process is already pending for"
938                 " this subscriber. Aborting.\n");
939         /* Also get rid of the other pending LU attempt? */
940         /*lu_fsm_failure(vsub->lu_fsm, GSM48_REJECT_CONGESTION);*/
941         lu_fsm_failure(fi, GSM48_REJECT_CONGESTION);
942         return -EINVAL;
943     }
944
945 static void lu_fsm_failure(struct osmo_fsm_inst *fi, enum gsm48_reject_value rej_cause)
946 {
947     struct lu_fsm_priv *lfp = lu_fsm_priv(fi);
948     lfp->vlr->ops.tx_lu_rej(lfp->msc_conn_ref, rej_cause ? : GSM48_REJECT_NETWORK_FAILURE);
949 }
```

Case 2: When the VLR receives an IMSI detach while a location update is being processed:

```
1153 /* See TS 23.012 version 9.10.0 4.3.2.1 "Process Detach_IMSI_VLR" */
1154 int vlr_subscr_rx_imsi_detach(struct vlr_subscr *vsub)
1155 {
1156     /* paranoia: should any LU or PARQ FSMs still be running, stop them. */
1157     vlr_subscr_cancel_attach_fsm(vsub, OSMO_FSM_TERM_ERROR, GSM48_REJECT_CONGESTION);
1158
1159 void vlr_subscr_cancel_attach_fsm(struct vlr_subscr *vsub,
1160                                 enum osmo_fsm_term_cause fsm_cause,
1161                                 uint8_t gsm48_cause)
1162 }
```

```
278 {
279     if (!vsub)
280         return;
281
282     vlr_subscr_get(vsub);
283     if (vsub->lu_fsm)
284         vlr_loc_update_cancel(vsub->lu_fsm, fsm_cause, gsm48_cause);
```

```
1466 void vlr_loc_update_cancel(struct osmo_fsm_inst *fi,
1467                             enum osmo_fsm_term_cause fsm_cause,
1468                             uint8_t gsm48_cause)
1469 {
1470     [...]
1471     if (fi->state != VLR_ULA_S_DONE)
1472         lu_fsm_failure(fi, gsm48_cause);
```

Case 3: When the MSC connection times out while an LU is in progress:

```
1173 void vlr_ran_conn_timeout(struct vlr_subscr *vsub)
1174 {
1175     vlr_subscr_cancel_attach_fsm(vsub, OSMO_FSM_TERM_TIMEOUT, GSM48_REJECT_CONGESTION);
1176 }
```

In all cases, some code with "EBUSY" semantics would seem more appropriate, wouldn't it?  
I will try to find a reason code we could use instead.

Perhaps we don't even need to transmit a LU reject message in some of these cases?

**#4 - 07/18/2019 05:41 AM - laforge**

- Assignee changed from stsp to osmith

**#5 - 07/18/2019 05:41 AM - laforge**

- Status changed from In Progress to New