OsmoMSC - Bug #4337

osmo-msc: Don't send LU-Reject after LU-Accept if no TMSI Realloc Complete is received

12/17/2019 09:09 PM - pespin

Status:	New	**Start date:**	12/17/2019
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Mobility Management		
Target version:			
Resolution:		**Spec Reference:**	

Description

```
pespin> neels, hi! I'm looking at some osmo-gsm-tester tests. I see osmo-msc answering a LU Reques
t with an LU Accept, and 4 seconds later answer it with a LU Reject (cause: congestion 22)
<pespin> the reject seems to be triggered by fsm.c:322 msc_a(IMSI-901700000015253:MSISDN-7770:TMSI
new-0x0E475C08:GERAN-A-1:LU)[0x612000010420]{MSC_A_ST_AUTH_CIPH}: Timeout of X1
<pespin> is that expected?
<pespin> is it because a TMSI Reallocation Complete is expected from uplink?
<pespin> I'm not really sure if it's correct sending a LU Reject after already having sent an LU a
ccept...
<pespin> neels, I introduced a TTCN3 that reproduces the scenario: https://gerrit.osmocom.org/c/os
mo-ttcn3-hacks/+/16648
<neels> pespin, so in a setup without TMSI, the LU Accept is the end of it. However, with TMSI ena
bled, the LU Accept sends the newly assigned TMSI for future conns
<neels> so then we are only successful when the MS responds with "TMSI Reallocation Complete"
<neels>I'm not 100% sure whether we should still regard the LU as successful, but the code as it i
s now completely rejects when the TMSI reallocation didn't work, on purpose
<pespin>30 yes, been learning that while looking at it later. Does it make sense to send a reject
after already having sent an Accept thought?
<neels>IIRC I took that behavior from the osmo-nitb code base, we'd need to ride the specs to find
 out for sure
<pespin>30 neels, any hint on where to look at?
<neels>(sorry for not answering sooner, I missed the highlight)
<pespin>30 np
<neels> TS 24.008 or 48.008 maybe?
<LaF0rge> neels: it most be in 04.08 or its successors, 24.008 is a good guess as it describes L3
MM+CC.  48.008/08.08 sounds rather unlikely
<LaF0rge> 44.018 also not likely, as that's the RR part of 04.08.
<neels> yep, 48.008 is about BSSAP...
<pespin> neels, LaF0rge TS 24.008 4.3.1.5 "If the RR connection is lost before the TMSI REALLOCATI
ON COMPLETE message is received, the network shall release all MM connections, if any. Furthermore
, the network should consider both the old and the new TMSI as occupied for a certain recovery tim
e."
<pespin> and during that period (old and new TMSI as occupied afaiu): the network can use the IMSI
 or if the MS sneds something then the TMSI realloc procedure is restarted. It can even consider t
he new TMSI as valid if the MS is using it afterwards.
<pespin> so afaiu that implies the LU is one part, and TMSI realloc is another stage afterwards wh
ich happens to go together in the same message in LU Accept for performance timing reasons
<pespin> so LU reject shouldn't be sent at that point.. or is that the meaning for "release all MM
 connections" ?
<neels> I interpret "release all connections" as a "Clear Command", which we normally do after a L
U.
<neels> so it seems we should see it as an accepted LU
<neels> (but in practice it's not something that typically happens)
<neels> (so if we fix the behavior, we're not likely to see any practical impact at all.)
```

Related test: TTCN3 MSC_Tests.TC_lu_imsi_timeout_tmsi_realloc

Related issues:

Related to OsmoMSC - Feature #4336: Convert vlr_lu_fsm.c to use osmo_tdef (an...	**New**	**12/17/2019**

History

#1 - 12/17/2019 09:10 PM - pespin

- Related to Feature #4336: Convert vlr_lu_fsm.c to use osmo_tdef (and drop vlr_timer()) added

#2 - 01/07/2021 05:58 PM - laforge

I'm seeing the following on osmo-msc 1.6.1 which could be related:

```
<000f> ../../../git/src/libmsc/sccp_ran.c:84 (UTRAN-Iu-3 from RI=SSN_PC,PC=0.23.5,SSN=RANAP) sccp_ran_sap_up(N
-CONNECT.indication)
<000f> ../../../git/src/libmsc/ran_msg_iu.c:270 msc_a(unknown:UTRAN-Iu-3:NONE)[0x81f1948]{MSC_A_ST_VALIDATE_L3
}: RAN decode: RANAP: dir=1 proc=19
<000f> ../../../git/src/libmsc/msc_a.c:1547 msc_a(unknown:UTRAN-Iu-3:NONE)[0x81f1948]{MSC_A_ST_VALIDATE_L3}: R
AN decode: RANAP InitialUE RAN PDU
<0002> ../../../git/src/libmsc/gsm_04_08.c:347 msc_a(IMSI-001010123456063:UTRAN-Iu-3:LU)[0x81f1948]{MSC_A_ST_V
ALIDATE_L3}: LOCATION UPDATING REQUEST: MI=IMSI-001010123456063 LU-type=NORMAL
<0002> ../../../git/src/libmsc/gsm_04_08.c:390 msc_a(IMSI-001010123456063:UTRAN-Iu-3:LU)[0x81f1948]{MSC_A_ST_V
ALIDATE_L3}: USIM: old LAI: 001-01-65534
<0002> ../../../git/src/libmsc/gsm_04_08.c:625 Tx AUTH REQ (rand = 5c4557c7f471a34f08e58497c999966a)
<0002> ../../../git/src/libmsc/gsm_04_08.c:627    AUTH REQ (autn = c4f074a5480000005c4455c4f074a548)
<000f> ../../../git/src/libmsc/msc_a.c:1614 msc_a(IMSI-001010123456063:UTRAN-Iu-3:LU)[0x81f1948]{MSC_A_ST_AUTH
_CIPH}: Sending DTAP: MM GSM48_MT_MM_AUTH_REQ
<000f> ../../../git/src/libmsc/ran_msg_iu.c:402 msc_a(IMSI-001010123456063:UTRAN-Iu-3:LU)[0x81f1948]{MSC_A_ST_
AUTH_CIPH}: RAN encode: RANAP: DirectTransfer
<000f> ../../../git/src/libmsc/sccp_ran.c:108 (UTRAN-Iu-3) sccp_ran_sap_up(N-DATA.indication)
<000f> ../../../git/src/libmsc/ran_msg_iu.c:270 msc_i(IMSI-001010123456063:UTRAN-Iu-3:LU)[0x81f17a0]{READY}: R
AN decode: RANAP: dir=1 proc=20
<000f> ../../../git/src/libmsc/ran_msg_iu.c:270 msc_a(IMSI-001010123456063:UTRAN-Iu-3:LU)[0x81f1948]{MSC_A_ST_
AUTH_CIPH}: RAN decode: RANAP: dir=1 proc=20
<000f> ../../../git/src/libmsc/msc_a.c:1547 msc_a(IMSI-001010123456063:UTRAN-Iu-3:LU)[0x81f1948]{MSC_A_ST_AUTH
_CIPH}: RAN decode: RANAP DirectTransfer RAN PDU
<0002> ../../../git/src/libmsc/gsm_04_08.c:1007 msc_a(IMSI-001010123456063:UTRAN-Iu-3:LU)[0x81f1948]{MSC_A_ST_
AUTH_CIPH}: MM UMTS AUTHENTICATION RESPONSE (res = 5c4455c4f074a54800ec8e9cc5949865)
<000f> ../../../git/src/libmsc/ran_msg_iu.c:408 msc_a(IMSI-001010123456063:UTRAN-Iu-3:LU)[0x81f1948]{MSC_A_ST_
AUTH_CIPH}: RAN encode: RANAP: SecurityModeCommand
<000f> ../../../git/src/libmsc/ran_msg_iu.c:381 msc_a(IMSI-001010123456063:UTRAN-Iu-3:LU)[0x81f1948]{MSC_A_ST_
AUTH_CIPH}: RAN encode: RANAP: Tx RANAP SECURITY MODE COMMAND to RNC, ik 55c4f074a54800ec8e9cc59498655c44
<000f> ../../../git/src/libmsc/sccp_ran.c:108 (UTRAN-Iu-3) sccp_ran_sap_up(N-DATA.indication)
<000f> ../../../git/src/libmsc/ran_msg_iu.c:270 msc_i(IMSI-001010123456063:UTRAN-Iu-3:LU)[0x81f17a0]{READY}: R
AN decode: RANAP: dir=2 proc=6
<000f> ../../../git/src/libmsc/ran_msg_iu.c:270 msc_a(IMSI-001010123456063:UTRAN-Iu-3:LU)[0x81f1948]{MSC_A_ST_
AUTH_CIPH}: RAN decode: RANAP: dir=2 proc=6
<000f> ../../../git/src/libmsc/msc_a.c:1547 msc_a(IMSI-001010123456063:UTRAN-Iu-3:LU)[0x81f1948]{MSC_A_ST_AUTH
_CIPH}: RAN decode: RANAP SecurityModeControl successfulOutcome
<000f> ../../../git/src/libmsc/ran_msg_iu.c:416 msc_a(IMSI-001010123456063:UTRAN-Iu-3:LU)[0x81f1948]{MSC_A_ST_
AUTH_CIPH}: RAN encode: RANAP: CommonId
<000e> ../../../git/src/libmsc/gsm_04_08.c:1392 SUBSCR(IMSI-001010123456063) VLR: update for IMSI=001010123456
063 (MSISDN=)
<000e> ../../../git/src/libmsc/gsm_04_08.c:1392 SUBSCR(IMSI-001010123456063:TMSInew-0x052CC223) VLR: update fo
r IMSI=001010123456063 (MSISDN=)
<0002> ../../../git/src/libmsc/gsm_04_08.c:153 -> IMSI-001010123456063:TMSInew-0x052CC223 LOCATION UPDATE ACCE
PT (TMSI = 0x052cc223)
<000f> ../../../git/src/libmsc/msc_a.c:1614 msc_a(IMSI-001010123456063:TMSInew-0x052CC223:UTRAN-Iu-3:LU)[0x81f
1948]{MSC_A_ST_AUTH_CIPH}: Sending DTAP: MM GSM48_MT_MM_LOC_UPD_ACCEPT
<000f> ../../../git/src/libmsc/ran_msg_iu.c:402 msc_a(IMSI-001010123456063:TMSInew-0x052CC223:UTRAN-Iu-3:LU)[0
x81f1948]{MSC_A_ST_AUTH_CIPH}: RAN encode: RANAP: DirectTransfer
<000f> ../../../git/src/libmsc/msc_a.c:738 msc_a(IMSI-001010123456063:TMSInew-0x052CC223:UTRAN-Iu-3:LU)[0x81f1
948]{MSC_A_ST_RELEASING}: Releasing: msc_a use is 0 (-)
<0002> ../../../git/src/libmsc/gsm_04_08.c:108 msc_a(IMSI-001010123456063:TMSInew-0x052CC223:UTRAN-Iu-3:LU)[0x
81f1948]{MSC_A_ST_RELEASING}: LOCATION UPDATING REJECT
<000f> ../../../git/src/libmsc/msc_a.c:1614 msc_a(IMSI-001010123456063:TMSInew-0x052CC223:UTRAN-Iu-3:LU)[0x81f
1948]{MSC_A_ST_RELEASING}: Sending DTAP: MM GSM48_MT_MM_LOC_UPD_REJECT
<000f> ../../../git/src/libmsc/ran_msg_iu.c:402 msc_a(IMSI-001010123456063:TMSInew-0x052CC223:UTRAN-Iu-3:LU)[0
x81f1948]{MSC_A_ST_RELEASING}: RAN encode: RANAP: DirectTransfer
<0006> ../../../git/src/libvlr/vlr_lu_fsm.c:733 msc_a(IMSI-001010123456063:TMSInew-0x052CC223:UTRAN-Iu-3:LU)[0
x81f1948]{MSC_A_ST_RELEASING}: Event MSC_A_EV_CN_CLOSE not permitted
<000f> ../../../git/src/libmsc/ran_msg_iu.c:420 msc_a(IMSI-001010123456063:TMSInew-0x052CC223:UTRAN-Iu-3:LU)[0
x81f1948]{MSC_A_ST_RELEASING}: RAN encode: RANAP: Iu Release
<000f> ../../../git/src/libmsc/sccp_ran.c:116 (UTRAN-Iu-3) sccp_ran_sap_up(N-DISCONNECT.indication)
<000f> ../../../git/src/libmsc/ran_msg_iu.c:270 msc_i(IMSI-001010123456063:TMSInew-0x052CC223:UTRAN-Iu-3:LU)[0
x81f17a0]{READY}: RAN decode: RANAP: dir=2 proc=1
<000f> ../../../git/src/libmsc/ran_msg_iu.c:270 msc_a(IMSI-001010123456063:TMSInew-0x052CC223:UTRAN-Iu-3:LU)[0
x81f1948]{MSC_A_ST_RELEASING}: RAN decode: RANAP: dir=2 proc=1
```

```
<000f> ../../../git/src/libmsc/msc_a.c:1547 msc_a(IMSI-001010123456063:TMSInew-0x052CC223:UTRAN-Iu-3:LU)[0x81f
1948]{MSC_A_ST_RELEASING}: RAN decode: RANAP Iu Release successfulOutcome
<000f> ../../../git/src/libmsc/msc_a.c:828 msc_a(IMSI-001010123456063:TMSInew-0x052CC223:UTRAN-Iu-3:LU)[0x81f1
948]{MSC_A_ST_RELEASED}: Released: msc_a use is 1 (msc_a_ran_dec)
<000f> ../../../git/src/libmsc/ran_conn.c:142 ran_peer(UTRAN-Iu:RI-SSN_PC:PC-0-23-5:SSN-RANAP)[0x81f12c8]{READ
Y}: Closing UTRAN-Iu-3
```

#3 - 01/07/2021 06:09 PM - laforge

Clearly in this setup there is no TMSI REALLOC COMPLET from the UE. It's a Samsung Galaxy S5 connected via a nano3G S8.

Regarding the spec quote:

> If the RR connection is lost before the TMSI REALLOCATION COMPLETE message is received, the network shall release all MM connections, if any. Furthermore, the network should consider both the old and the new TMSI as occupied for a certain recovery time.

This applies *if the RR connection is lost*. Whereas in this example, over UMTS, the SCCP connection (and hence the RR connection) is very much alive until we as the MSC decide to kill it at the 4s timeout after sending the LU ACCEPT.

#4 - 01/07/2021 06:22 PM - laforge

Ok, in this situation it apparently was a mismatch between LAI being broadcast on the radio interface and LAI confirmed in the LAU ACCEPT. This made the phone ignore the LU ACCEPT and not send a TMSI REALLOC COMPLETE.