OsmoMSC - Bug #4191

vlr.c:762 Trying to dispatch event 1 to non-existent FSM instance!

09/06/2019 08:43 AM - MPoslusny

Status:	Resolved	**Start date:**	09/06/2019
Priority:	Normal	**Due date:**	
Assignee:	neels	**% Done:**	100%
Category:			
Target version:			
Resolution:		**Spec Reference:**	

Description

Hello,
I test sysmoBTS 1002 with osmo CN against 35 stations. On the stations is a script that registers and unregisters them into the network in the cycle (and switches between EDGE and LTE technologies).
After about two minutes, this error will appear in the MSC log.

BR
Marek

```
apt-cache showpkg osmo-msc
Package: osmo-msc
Versions:
1.5.0 (/var/lib/apt/lists/download.opensuse.org_repositories_network:_osmocom:_latest_xUbuntu%5f18
.04_._Packages) (/var/lib/dpkg/status)
 Description Language:
                 File: /var/lib/apt/lists/download.opensuse.org_repositories_network:_osmocom:_lat
est_xUbuntu%5f18.04_._Packages
                  MD5: 56b9ef74e5f59bb90af4a5ea954cc999
 Description Language:
                 File: /var/lib/apt/lists/download.opensuse.org_repositories_network:_osmocom:_lat
est_xUbuntu%5f18.04_._Packages
                  MD5: 56b9ef74e5f59bb90af4a5ea954cc999

Reverse Depends:
  osmo-msc-dbg,osmo-msc 1.5.0
  osmo-msc:i386,osmo-msc
Dependencies:
1.5.0 - osmocom-latest (0 (null)) libasn1c1 (0 (null)) libc6 (2 2.14) libdbi1 (2 0.9.0) libosmo-gs
up-client0 (0 (null)) libosmo-mgcp-client6 (0 (null)) libosmo-ranap2 (0 (null)) libosmo-sigtran3 (
0 (null)) libosmocore12 (0 (null)) libosmoctrl0 (0 (null)) libosmogsm13 (0 (null)) libosmonetif8 (
0 (null)) libosmovty4 (0 (null)) libsctp1 (2 1.0.10+dfsg) libsmpp1 (0 (null)) libtalloc2 (2 2.0.4~
git20101213) libdbd-sqlite3 (0 (null)) osmo-msc:i386 (32 (null))
Provides:
1.5.0 - osmo-msc:i386 (= 1.5.0)
Reverse Provides:
osmo-msc:i386 1.5.0 (= 1.5.0)

Fri Sep  6 09:44:46 2019 DMSC <0006> msc_a.c:361 msc_a(IMSI-901700000025369:MSISDN-0600000229:TMSI
-0x00D6C8C0:GERAN-A-50:LU)[0x5555fd915b80]{MSC_A_ST_RELEASING}: transition to state MSC_A_ST_RELEA
SING not permitted!
Fri Sep  6 09:44:49 2019 DLGLOBAL <0013> vlr.c:762 Trying to dispatch event 1 to non-existent FSM
instance!
Fri Sep  6 09:44:49 2019 DLGLOBAL <0013> backtrace.c:47 backtrace() returned 10 addresses
Fri Sep  6 09:44:49 2019 DLGLOBAL <0013> backtrace.c:57         /usr/lib/x86_64-linux-gnu/libosmoc
ore.so.12(_osmo_fsm_inst_dispatch+0x2e7) [0x7fd80c52f6a7]
Fri Sep  6 09:44:49 2019 DLGLOBAL <0013> backtrace.c:57         osmo-msc(+0x58ded) [0x5555fbb76ded
]
Fri Sep  6 09:44:49 2019 DLGLOBAL <0013> backtrace.c:57         osmo-msc(+0x213b3) [0x5555fbb3f3b3
]
Fri Sep  6 09:44:49 2019 DLGLOBAL <0013> backtrace.c:57         /usr/lib/x86_64-linux-gnu/libosmo-
gsup-client.so.0(+0x121d) [0x7fd80b2a121d]
Fri Sep  6 09:44:49 2019 DLGLOBAL <0013> backtrace.c:57         /usr/lib/x86_64-linux-gnu/libosmoa
```

```
bis.so.6(+0x96c4) [0x7fd80a2e96c4]
Fri Sep  6 09:44:49 2019 DLGLOBAL <0013> backtrace.c:57         /usr/lib/x86_64-linux-gnu/libosmoc
ore.so.12(osmo_select_main+0x227) [0x7fd80c52aef7]
Fri Sep  6 09:44:49 2019 DLGLOBAL <0013> backtrace.c:57         osmo-msc(+0x1141a) [0x5555fbb2f41a
]
Fri Sep  6 09:44:49 2019 DLGLOBAL <0013> backtrace.c:57         /lib/x86_64-linux-gnu/libc.so.6(__
libc_start_main+0xe7) [0x7fd80aabdb97]
Fri Sep  6 09:44:49 2019 DLGLOBAL <0013> backtrace.c:57         osmo-msc(+0x11ada) [0x5555fbb2fada
]
Fri Sep  6 09:44:54 2019 DMSC <0006> vlr_lu_fsm.c:733 msc_a(TMSI-0xB0FA53F3:GERAN-A-51:LU)[0x5555f
d916d00]{MSC_A_ST_RELEASING}: Event MSC_A_EV_CN_CLOSE not permitted
Fri Sep  6 09:44:54 2019 DMSC <0006> msc_a.c:361 msc_a(TMSI-0xB0FA53F3:GERAN-A-51:LU)[0x5555fd916d
00]{MSC_A_ST_RELEASING}: transition to state MSC_A_ST_RELEASING not permitted!
Fri Sep  6 09:44:56 2019 DLGLOBAL <0013> vlr.c:762 Trying to dispatch event 1 to non-existent FSM
instance!
Fri Sep  6 09:44:56 2019 DLGLOBAL <0013> backtrace.c:47 backtrace() returned 10 addresses
Fri Sep  6 09:44:56 2019 DLGLOBAL <0013> backtrace.c:57         /usr/lib/x86_64-linux-gnu/libosmoc
ore.so.12(_osmo_fsm_inst_dispatch+0x2e7) [0x7fd80c52f6a7]
Fri Sep  6 09:44:56 2019 DLGLOBAL <0013> backtrace.c:57         osmo-msc(+0x58ded) [0x5555fbb76ded
]
Fri Sep  6 09:44:56 2019 DLGLOBAL <0013> backtrace.c:57         osmo-msc(+0x213b3) [0x5555fbb3f3b3
]
Fri Sep  6 09:44:56 2019 DLGLOBAL <0013> backtrace.c:57         /usr/lib/x86_64-linux-gnu/libosmo-
gsup-client.so.0(+0x121d) [0x7fd80b2a121d]
Fri Sep  6 09:44:56 2019 DLGLOBAL <0013> backtrace.c:57         /usr/lib/x86_64-linux-gnu/libosmoa
bis.so.6(+0x96c4) [0x7fd80a2e96c4]
Fri Sep  6 09:44:56 2019 DLGLOBAL <0013> backtrace.c:57         /usr/lib/x86_64-linux-gnu/libosmoc
ore.so.12(osmo_select_main+0x227) [0x7fd80c52aef7]
Fri Sep  6 09:44:56 2019 DLGLOBAL <0013> backtrace.c:57         osmo-msc(+0x1141a) [0x5555fbb2f41a
]
Fri Sep  6 09:44:56 2019 DLGLOBAL <0013> backtrace.c:57         /lib/x86_64-linux-gnu/libc.so.6(__
libc_start_main+0xe7) [0x7fd80aabdb97]
Fri Sep  6 09:44:56 2019 DLGLOBAL <0013> backtrace.c:57         osmo-msc(+0x11ada) [0x5555fbb2fada
]
Fri Sep  6 09:45:00 2019 DLGLOBAL <0013> vlr.c:762 Trying to dispatch event 1 to non-existent FSM
instance!
Fri Sep  6 09:45:00 2019 DLGLOBAL <0013> backtrace.c:47 backtrace() returned 10 addresses
Fri Sep  6 09:45:00 2019 DLGLOBAL <0013> backtrace.c:57         /usr/lib/x86_64-linux-gnu/libosmoc
ore.so.12(_osmo_fsm_inst_dispatch+0x2e7) [0x7fd80c52f6a7]
Fri Sep  6 09:45:00 2019 DLGLOBAL <0013> backtrace.c:57         osmo-msc(+0x58ded) [0x5555fbb76ded
]
Fri Sep  6 09:45:00 2019 DLGLOBAL <0013> backtrace.c:57         osmo-msc(+0x213b3) [0x5555fbb3f3b3
]
Fri Sep  6 09:45:00 2019 DLGLOBAL <0013> backtrace.c:57         /usr/lib/x86_64-linux-gnu/libosmo-
gsup-client.so.0(+0x121d) [0x7fd80b2a121d]
Fri Sep  6 09:45:00 2019 DLGLOBAL <0013> backtrace.c:57         /usr/lib/x86_64-linux-gnu/libosmoa
bis.so.6(+0x96c4) [0x7fd80a2e96c4]
Fri Sep  6 09:45:00 2019 DLGLOBAL <0013> backtrace.c:57         /usr/lib/x86_64-linux-gnu/libosmoc
ore.so.12(osmo_select_main+0x227) [0x7fd80c52aef7]
Fri Sep  6 09:45:00 2019 DLGLOBAL <0013> backtrace.c:57         osmo-msc(+0x1141a) [0x5555fbb2f41a
]
Fri Sep  6 09:45:00 2019 DLGLOBAL <0013> backtrace.c:57         /lib/x86_64-linux-gnu/libc.so.6(__
libc_start_main+0xe7) [0x7fd80aabdb97]
Fri Sep  6 09:45:00 2019 DLGLOBAL <0013> backtrace.c:57         osmo-msc(+0x11ada) [0x5555fbb2fada
]
```

Related issues:

Related to OsmoMSC - Bug #4721: osmo-msc creates evil-twin entries in the VLR...	**Stalled**	**08/19/2020**

History

#1 - 09/16/2019 04:04 PM - neels

Thanks for your report!

Your log transcript starts with "transition to state MSC_A_ST_RELEASING not permitted", but most interesting would be: why did that Release happen?

The non-existing FSM the crash complains about is the auth_fsm. The code path is receiving apparently a GSUP AUTH_INFO_RESULT for a conn that has no auth FSM (anymore?).
Certainly we can easily add a condition that catches this error. But what I really would like to know is: how did this situation come about?

In our tests and "normal" situations, we never encounter this error.
So, to more thoroughly understand why this error happens, and to be able to create a regression test for this:

@MPoslusny , can you reproduce this issue and provide a more complete error log, best even from the startup of osmo-msc up to the error?
If possible, can you also provide a network trace (pcap) of the error?
Thanks!

#2 - 09/16/2019 04:11 PM - neels

- Status changed from New to In Progress

- Assignee changed from sysmocom to neels

- % Done changed from 0 to 50

https://gerrit.osmocom.org/c/osmo-msc/+/15544

#3 - 09/19/2019 08:22 AM - MPoslusny

Hello,

Because logs may contain sensitive information, I do not want to publish them on the web without login at the moment.
But I'm ready to send them to the individual developers working on it.

@Neels, I sent you the logs (17/9/2019) to an e-mail in your profile, but I'm not sure they arrived.

BR
Marek

#4 - 09/23/2019 02:04 AM - neels

Thanks, have received a tar with an osmo-msc and a pcap which I can open.
However, the pcap doesn't seem to contain a useful trace. I see some GTP negotiation, but no BSSAP or Abis.
Anyhow, the log reveals this, individual lines picked with lots left out in-between:

```
Tue Sep 17 13:06:58 2019 DMM &lt;0002&gt; gsm_04_08.c:324 msc_a(TMSI-0xDCC69AEC:GERAN-A-5:LU)[0x55dc513114c0]{
MSC_A_ST_VALIDATE_L3}: LOCATION UPDATING REQUEST: MI=TMSI-0xDCC69AEC LU-type=IMSI-ATTACH
  Tue Sep 17 13:06:59 2019 DVLR &lt;000e&gt; fsm.c:423 VLR_Authenticate(TMSI-0xDCC69AEC:GERAN-A-5:LU)[0x55dc51
314c80]{VLR_SUB_AS_NEEDS_AUTH}: Allocated
  Tue Sep 17 13:07:00 2019 DVLR &lt;000e&gt; vlr_auth_fsm.c:136 SUBSCR(IMSI-901700000025366:TMSI-0xDCC69AEC) A
UTH on GERAN received SRES/RES: bffff6d1cf5e6932 (8 bytes)
  Tue Sep 17 13:07:00 2019 DVLR &lt;000e&gt; vlr_auth_fsm.c:244 VLR_Authenticate(TMSI-0xDCC69AEC:GERAN-A-5:LU)
[0x55dc51314c80]{VLR_SUB_AS_WAIT_RESP}: Authentication terminating with result PASSED
  Tue Sep 17 13:07:00 2019 DMM &lt;0002&gt; gsm_04_08.c:153 -> IMSI-901700000025366:MSISDN-0600000226:TMSI-0xD
CC69AEC:TMSInew-0x0F4AAC5D LOCATION UPDATE ACCEPT (TMSI = 0x0f4aac5d)
  Tue Sep 17 13:07:00 2019 DRLL &lt;0000&gt; msc_a.c:1166 msc_a(IMSI-901700000025366:MSISDN-0600000226:TMSI-0x
DCC69AEC:TMSInew-0x0F4AAC5D:GERAN-A-5:LU)[0x55dc513114c0]{MSC_A_ST_AUTH_CIPH}: Dispatching 04.08 message: MM G
SM48_MT_MM_TMSI_REALL_COMPL
  Tue Sep 17 13:07:00 2019 DMM &lt;0002&gt; gsm_04_08.c:1071 msc_a(IMSI-901700000025366:MSISDN-0600000226:TMSI
-0xDCC69AEC:TMSInew-0x0F4AAC5D:GERAN-A-5:LU)[0x55dc513114c0]{MSC_A_ST_AUTH_CIPH}: TMSI Reallocation Completed
  Tue Sep 17 13:07:00 2019 DVLR &lt;000e&gt; vlr_lu_fsm.c:381 vlr_lu_fsm(IMSI-901700000025366:MSISDN-060000022
6:TMSI-0x0F4AAC5D:GERAN-A-5:LU)[0x55dc51311390]{VLR_ULA_S_WAIT_LU_COMPL}: Received Event VLR_ULA_E_LU_COMPL_SU
CCESS
```

At first the conn has an older TMSI == DCC69AEC (let's call it "DCC"), and assigns new TMSI 0F4AAC5D ("0F4"), in a successful Location Updating complete with auth. The MS should now be attached.
Then exactly a minute later we see another "IMSI-ATTACH" LU with TMSI 0F4:

```
Tue Sep 17 13:08:01 2019 DMM &lt;0002&gt; gsm_04_08.c:324 msc_a(TMSI-0x0F4AAC5D:GERAN-A-14:LU)[0x55dc5131d000]
{MSC_A_ST_VALIDATE_L3}: LOCATION UPDATING REQUEST: MI=TMSI-0x0F4AAC5D LU-type=IMSI-ATTACH
  Tue Sep 17 13:08:01 2019 DMM &lt;0002&gt; gsm_04_08.c:602 Tx AUTH REQ (rand = 74afe94e7991e012cd5cdcb187a400
75)
  Tue Sep 17 13:08:01 2019 DBSSAP &lt;0010&gt; ran_msg_a.c:1179 msc_a(IMSI-901700000025366:MSISDN-0600000226:T
MSI-0x0F4AAC5D:GERAN-A-14:LU)[0x55dc5131d000]{MSC_A_ST_AUTH_CIPH}: RAN encode: BSSMAP: DTAP
```

but get a clear request from the RAN right in the middle, before getting an auth result:

```
Tue Sep 17 13:08:02 2019 DBSSAP &lt;0010&gt; ran_msg_a.c:787 msc_i(IMSI-901700000025366:MSISDN-0600000226:TMSI
-0x0F4AAC5D:GERAN-A-14:LU)[0x55dc5130cb60]{READY}: RAN decode: BSSMAP: Rx BSSMAP DT1 CLEAR REQUEST
```

```
  Tue Sep 17 13:08:02 2019 DMSC &lt;0006&gt; fsm.c:530 msc_a(IMSI-901700000025366:MSISDN-0600000226:TMSI-0x0F4
AAC5D:GERAN-A-14:LU)[0x55dc5131d000]{MSC_A_ST_RELEASED}: Deallocated, including all deferred deallocations
```

That's weird enough:

- why should the MS attach again?
- why should the MS abort in the middle of the auth?

Putting aside weird MS behavior, osmo-msc should be able to deal with that.
The actual problem comes up later:
Jump to five minutes later, when we see, even weirder, **another** LU from the **old** TMSI "DCC":

```
Tue Sep 17 13:13:08 2019 DMM &lt;0002&gt; gsm_04_08.c:324 msc_a(TMSI-0xDCC69AEC:GERAN-A-70:LU)[0x55dc51317ce0]
{MSC_A_ST_VALIDATE_L3}: LOCATION UPDATING REQUEST: MI=TMSI-0xDCC69AEC LU-type=IMSI-ATTACH
  Tue Sep 17 13:13:09 2019 DRLL &lt;0000&gt; msc_a.c:1166 msc_a(TMSI-0xDCC69AEC:GERAN-A-70:LU)[0x55dc51317ce0]
{MSC_A_ST_AUTH_CIPH}: Dispatching 04.08 message: MM GSM48_MT_MM_ID_RESP
  Tue Sep 17 13:13:09 2019 DMM &lt;0002&gt; gsm_04_08.c:193 IDENTITY RESPONSE: MI=IMSI-901700000025366
  Tue Sep 17 13:13:09 2019 DVLR &lt;000e&gt; fsm.c:423 VLR_Authenticate(TMSI-0xDCC69AEC:GERAN-A-70:LU)[0x55dc5
131d670]{VLR_SUB_AS_NEEDS_AUTH}: Allocated
```

and then when it asks the HLR for the auth tuples, the response is somehow associated with the newer "0F4" TMSI,
which also should long have been a thing of the past from the prematurely Cleared LU attempt:

```
Tue Sep 17 13:13:09 2019 DVLR &lt;000e&gt; vlr.c:762 SUBSCR(IMSI-901700000025366:MSISDN-0600000226:TMSI-0x0F4A
AC5D) Received GSUP OSMO_GSUP_MSGT_SEND_AUTH_INFO_RESULT, but there is no auth_fsm
  Tue Sep 17 13:13:13 2019 DMSC &lt;0006&gt; fsm.c:284 msc_a(TMSI-0xDCC69AEC:GERAN-A-70:LU)[0x55dc51317ce0]{MS
C_A_ST_AUTH_CIPH}: Timeout of X1
```

(BTW, the log shows that the merged patch successfully avoids the reported crash.
The remainder of this issue is figuring out the cause for two different TMSIs / two connections for the same subscriber.)

What the heck is going on here.

The dance the MS is doing is quite weird:

- Successful LU with TMSI DCC successfully reallocates TMSI 0F4. MS acks 0F4.
- MS does a second LU with TMSI 0F4.
- The second LU gets aborted.
- The MS magically resurrects the old TMSI DCC (that was surpassed by the acked TMSI reallocation, and we even saw a LU with that new TMSI)

But also osmo-msc seems to keep the aborted auth_fsm for TMSI 0F4 alive for over 5 minutes, which shouldn't happen.
I guess that is because the vlr_subscr is already listed as attached and hence doesn't get cleared out on unsuccessful LU.
We need to ensure that even an already attached subscriber cleans out the auth_fsm / LU state if a LU gets aborted.

But how does the MS resurrect old TMSIs? @MPoslusny , are you using two SIM cards with the same IMSI??
Even that shouldn't be able to resurrect an old, already reallocated TMSI...

With this analysis we can try to reproduce the "missing auth_fsm" problem by replaying the odd MS behavior in a ttcn3 test.
Clarifying why the MS is behaving this way is optional from osmo-msc perspective.

If it's not much effort, I would still like to get some more detail on the logging, since some parts seem to be missing.
If you can find the time, would you configure osmo-msc as follows and reproduce the problem again?

osmo-msc.cfg

```
log stderr
 logging filter all 1
 logging color 1
 logging print level 1
 logging print category 1
 logging print category-hex 0
 logging print file basename last
 logging print extended-timestamp 1
 logging level set-all debug
 logging level linp notice
 logging level lss7 notice
 logging level lsccp notice
 logging level lsua notice
 logging level lm3ua notice

log gsmtap 127.0.0.1
 logging filter all 1
 logging level set-all debug
```

```
 logging level linp notice
 logging level lss7 notice
 logging level lsccp notice
 logging level lsua notice
 logging level lm3ua notice
```

And configure osmo-hlr with:

osmo-hlr.cfg

```
log stderr
 logging filter all 1
 logging color 1
 logging print level 1
 logging print category 1
 logging print category-hex 0
 logging print file basename last
 logging print extended-timestamp 1
 logging level set-all debug

log gsmtap 127.0.0.1
 logging filter all 1
 logging level set-all debug
```

Also, a pcap containing the actual negotiation would be nice.
There should be interesting info shown with each of these wireshark filters:

```
gsm_abis_rsl || bssap || gsmtap_log
```

But again, only if it is convenient and does not take much time.
Thanks!

#5 - 09/23/2019 12:27 PM - MPoslusny

Hello
It is unlikely that there are more SIM cards with the same IMSI in my network.
I might speculate that cellular modules are trying to establish Circuit switched and Packet switched connections simultaneously, but I'm not sure whether it's possible.

It seems that cellular modules PLS8-E do not cause this error or I have not noticed yet. While the MC7304, EC25 and PLS8-X modules do this frequently.

@Neels, I sent you new logs (23/9/2019) to the e-mail.

Thanks
Marek

#6 - 09/25/2019 09:46 AM - MPoslusny

I experimented with a single station (using EC25 module) capable to connect (direct connection using cabling into a system of diplexers, splitters, and attenuators combining 2G and 4G system).
It seems that the essential condition for the error occurs there is a technology change (module is disabled before reconfiguring) from 4G (a successful LTE connection is necessary) to 2G.
The module uses the TMSI from 4G (0x9cee59d5) for connection into 2G. The error occurs when TMSI changes into 0xBC0766FF (from previous 2G connection).
I can get the qdiag log from the module if useful, but I can't analyze it.

```
20190925103449602 DMSC DEBUG msub(IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF) 1 MSC-I still active
 (msub.c:78)
20190925103449602 DMSC DEBUG msub_fsm[0x5603b1e42750]{terminating}: Terminating in cascade, depth 2 (cause = O
SMO_FSM_TERM_REGULAR, caused by: msc_a(IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF:GERAN-A-9:LU)[0x
5603b1e49670]) (msub.c:112)
20190925103449602 DMSC DEBUG msc_i(IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF:GERAN-A-9:LU)[0x5603
b1e49970]{READY}: Terminating in cascade, depth 3 (cause = OSMO_FSM_TERM_PARENT, caused by: msc_a(IMSI-9017000
00025368:MSISDN-0600000228:TMSI-0xBC0766FF:GERAN-A-9:LU)[0x5603b1e49670]) (msub.c:112)
20190925103449602 DMSC DEBUG msc_i(IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF:GERAN-A-9:LU)[0x5603
b1e49970]{READY}: Removing from parent msub_fsm[0x5603b1e42750] (msub.c:112)
20190925103449602 DMSC DEBUG msc_i(IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF:GERAN-A-9:LU)[0x5603
b1e49970]{READY}: Deferring: will deallocate with msc_a(IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF
:GERAN-A-9:LU)[0x5603b1e49670] (fsm.c:514)
20190925103449602 DMSC DEBUG msub(IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF) Free (msub.c:118)
```

```
20190925103449602 DREF DEBUG VLR subscr IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF - active-conn:
now used by 1 (attached) (msub.c:374)
20190925103449602 DMSC DEBUG msub_fsm[0x5603b1e42750]{terminating}: Deferring: will deallocate with msc_a(IMSI
-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF:GERAN-A-9:LU)[0x5603b1e49670] (fsm.c:514)
20190925103449602 DMSC DEBUG msc_a(IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF:GERAN-A-9:LU)[0x5603
b1e49670]{MSC_A_ST_RELEASED}: Deallocated, including all deferred deallocations (fsm.c:530)

20190925104017963 DMM DEBUG IDENTITY RESPONSE: MI=IMSI-901700000025368 (gsm_04_08.c:193)
20190925104017967 DREF DEBUG VLR subscr IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF + vlr_gsup_rx:
now used by 2 (attached,vlr_gsup_rx) (vlr.c:1083)
20190925104017967 DVLR ERROR SUBSCR(IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF) Received GSUP OSMO
_GSUP_MSGT_SEND_AUTH_INFO_RESULT, but there is no auth_fsm (vlr.c:762)
20190925104017967 DREF DEBUG VLR subscr IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF - vlr_gsup_rx:
now used by 1 (attached) (vlr.c:1129)

20190925104022494 DREF DEBUG VLR subscr IMSI-901700000025368:TMSI-0x9CEE59D5 + msc_a_fsm_releasing_onenter: no
w used by 2 (active-conn,msc_a_fsm_releasing_onenter) (msc_a.c:725)
20190925104022494 DREF DEBUG VLR subscr IMSI-901700000025368:TMSI-0x9CEE59D5 + vlr_subscr_cancel_attach_fsm: n
ow used by 3 (active-conn,msc_a_fsm_releasing_onenter,vlr_subscr_cancel_attach_fsm) (vlr.c:293)
20190925104022494 DREF DEBUG VLR subscr IMSI-901700000025368:TMSI-0x9CEE59D5 - vlr_subscr_cancel_attach_fsm: n
ow used by 2 (active-conn,msc_a_fsm_releasing_onenter) (vlr.c:298)
20190925104022494 DREF DEBUG VLR subscr IMSI-901700000025368:TMSI-0x9CEE59D5 - msc_a_fsm_releasing_onenter: no
w used by 1 (active-conn) (msc_a.c:771)
20190925104022495 DMSC DEBUG msub(IMSI-901700000025368:TMSI-0x9CEE59D5) MSC-A terminated (msub.c:260)
20190925104022495 DMSC DEBUG msub(IMSI-901700000025368:TMSI-0x9CEE59D5) 1 MSC-I still active (msub.c:78)
20190925104022495 DMSC DEBUG msub(IMSI-901700000025368:TMSI-0x9CEE59D5) Free (msub.c:118)
20190925104022495 DREF INFO VLR subscr IMSI-901700000025368:TMSI-0x9CEE59D5 - active-conn: now used by 0 (-) (
msub.c:374)
20190925104022495 DVLR DEBUG freeing VLR subscr IMSI-901700000025368:TMSI-0x9CEE59D5 (max total use count was
3) (vlr.c:306)
20190925104039148 DMM DEBUG IDENTITY RESPONSE: MI=IMSI-901700000025368 (gsm_04_08.c:193)

01909251040391 51 DREF DEBUG VLR subscr IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF + vlr_gsup_rx: n
ow used by 2 (attached,vlr_gsup_rx) (vlr.c:1083)
20190925104039151 DVLR ERROR SUBSCR(IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF) Received GSUP OSMO
_GSUP_MSGT_SEND_AUTH_INFO_RESULT, but there is no auth_fsm (vlr.c:762)
20190925104039151 DREF DEBUG VLR subscr IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF - vlr_gsup_rx:
now used by 1 (attached) (vlr.c:1129)

20190925104043679 DREF DEBUG VLR subscr IMSI-901700000025368:TMSI-0x9CEE59D5 + msc_a_fsm_releasing_onenter: no
w used by 2 (active-conn,msc_a_fsm_releasing_onenter) (msc_a.c:725)
20190925104043679 DREF DEBUG VLR subscr IMSI-901700000025368:TMSI-0x9CEE59D5 + vlr_subscr_cancel_attach_fsm: n
ow used by 3 (active-conn,msc_a_fsm_releasing_onenter,vlr_subscr_cancel_attach_fsm) (vlr.c:293)
20190925104043679 DREF DEBUG VLR subscr IMSI-901700000025368:TMSI-0x9CEE59D5 - vlr_subscr_cancel_attach_fsm: n
ow used by 2 (active-conn,msc_a_fsm_releasing_onenter) (vlr.c:298)
20190925104043679 DREF DEBUG VLR subscr IMSI-901700000025368:TMSI-0x9CEE59D5 - msc_a_fsm_releasing_onenter: no
w used by 1 (active-conn) (msc_a.c:771)
20190925104043680 DMSC DEBUG msub(IMSI-901700000025368:TMSI-0x9CEE59D5) MSC-A terminated (msub.c:260)
20190925104043680 DMSC DEBUG msub(IMSI-901700000025368:TMSI-0x9CEE59D5) 1 MSC-I still active (msub.c:78)
20190925104043680 DMSC DEBUG msub(IMSI-901700000025368:TMSI-0x9CEE59D5) Free (msub.c:118)
20190925104043680 DREF INFO VLR subscr IMSI-901700000025368:TMSI-0x9CEE59D5 - active-conn: now used by 0 (-) (
msub.c:374)
20190925104043680 DVLR DEBUG freeing VLR subscr IMSI-901700000025368:TMSI-0x9CEE59D5 (max total use count was
3) (vlr.c:306)

20190925104100333 DMM DEBUG IDENTITY RESPONSE: MI=IMSI-901700000025368 (gsm_04_08.c:193)
20190925104100336 DREF DEBUG VLR subscr IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF + vlr_gsup_rx:
now used by 2 (attached,vlr_gsup_rx) (vlr.c:1083)
20190925104100336 DVLR ERROR SUBSCR(IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF) Received GSUP OSMO
_GSUP_MSGT_SEND_AUTH_INFO_RESULT, but there is no auth_fsm (vlr.c:762)
20190925104100336 DREF DEBUG VLR subscr IMSI-901700000025368:MSISDN-0600000228:TMSI-0xBC0766FF - vlr_gsup_rx:
now used by 1 (attached) (vlr.c:1129)
```

#7 - 09/26/2019 10:47 PM - neels

- Status changed from In Progress to Feedback

- Assignee changed from neels to laforge

MPoslusny wrote:

> It seems that the essential condition for the error occurs there is a technology change (module is disabled before reconfiguring) from 4G (a successful LTE connection is necessary) to 2G.

This is very interesting information and probably explains why this has not been reported before.
Personally, I have no 4G set up for tests.
Maybe someone with 4G hardware/experience could reproduce this or translate to a test case?

#8 - 09/26/2019 11:23 PM - neels

- Status changed from Feedback to In Progress

- Assignee changed from laforge to neels

Taking another short look, it seems to me that this is actually a fault in osmo-msc in handling LU with an unknown TMSI, in a case where the IMSI already exists in the VLR:

- Upon receiving a LU with TMSI, we create a vlr_subscr without IMSI.
- We send an Identity Request (IMSI)
- When the MS replies and we know the IMSI, we fail to check that this same subscriber already may exist in the VLR, and end up with **two** vlr_subscr on the same IMSI.
- Instead we set the new vlr_subscr's IMSI, creating an evil twin in the VLR.
- Subsequently, a GSUP Rx looks up the vlr_subscr by IMSI and invariably finds the earlier twin; the auth_fsm though was put in the later twin.

So this seems unrelated to 4G as such, and might also appear if a 2G subscriber is re-attaching with an unknown TMSI.
This is a near impossible situation when only 2G/3G is available with exactly one osmo-msc running,
nevertheless osmo-msc should be able to resolve such situations without creating evil twins in the VLR.

IIUC, the only difference that 4G makes here is that the SGs interface may establish a new TMSI, which the phone subsequently uses towards 2G.
I would actually still expect us to somehow know TMSIs established on 4G, but maybe we only know about those in the case of CSFB?? Not sure there.

#9 - 09/27/2019 01:25 AM - neels

I tried to come up with a quick fix for a duplicate IMSI in the VLR, but have hit these complications:

- If an MS attaches and we have an old record for that same IMSI, we should drop the old record.

- However, if someone attaches with the same IMSI, that someone must first be authenticated before we discard another conn.
  Otherwise anyone coming along claiming to own a given IMSI would be able to detach any other subscriber without having to authenticate.
  So, we must allow more than one vlr_subscr records for the same IMSI to stick around in the VLR until we have actually authenticated the new conn.
  Only when authentication is through can we be sure to clean up old records for this IMSI.

- But, to authenticate a subscriber, we need to talk GSUP to osmo-hlr.
  The way we find the target for any GSUP response is: looking it up by IMSI.
  From above considerations, an IMSI may sit in the VLR more than once (awaiting authentication).
  It is not enough to dispatch to the first vlr_subscr found, because new attach attempts would never succeed.
  We cannot route GSUP only to subscribers that are already authenticated, because we need the GSUP response in order to authenticate in the first place.
  If a GSUP response comes in for an IMSI, we must actually attempt to dispatch it for all VLR entries that match this IMSI: There is no way to tell which one is the source of the GSUP request.

So, we must allow an IMSI to exist more than once in the VLR (for the duration of an attach attempt),
and we need to dispatch GSUP to **all** vlr_subscr with a given IMSI.

However, we must also prevent an unauthenticated subscriber to benefit from the GSUP responses for an attached subscriber with the same IMSI.
So we need to filter GSUP message dispatch by msg types: Only very specific GSUP messages should be received by an unattached vlr_subscr.

#10 - 09/27/2019 01:50 AM - neels

- Status changed from In Progress to Stalled

I think I might be spinning off here a little. There must rather be a way to keep the IMSI unique in the VLR.
I should compare to what happens if a Location Updating comes in with an IMSI that we already know,
and LU for an unknown TMSI should plug into that same code path as soon as the IMSI becomes known
(replace the vlr_subscr for the new conn, could be hard enough but that's probably the way to do it).
All considerations about whether an unauthenticated conn may violate a valid one just by claiming the same IMSI should be handled above those considerations.

I am stopping on this topic for now.
I thought it would be quick and easy to avoid evil twin IMSIs in the VLR, but it is taking too long to resolve at this time.

#11 - 09/27/2019 06:56 AM - MPoslusny

Hello

> I would actually still expect us to somehow know TMSIs established on 4G, but maybe we only know about those in the case of CSFB?? Not sure there.

The 4G system does not know about the 2G system and vice versa. There is no signaling link between these systems. Both systems have a separate HLR.

@Neels, It seems your fix works in my case. Thank you for your efforts!

BR
Marek

#12 - 09/27/2019 08:12 PM - laforge

On Thu, Sep 26, 2019 at 11:23:34PM +0000, neels [REDMINE] wrote:

> Taking another short look, it seems to me that this is actually a fault in osmo-msc in handling LU with an unknown TMSI, in a case where the IMSI already exists in the VLR:

Isn't this a relatively frequent event in any non-trivial network (i.e.
a network with multiple MSCs)?

In the absence of supporting an interface between the MSCs where the
'new' MSC can obtain the subscriber information based on the TMSI
allocated by the old MSC, we frequently encounter UEs which perform a LU
by TMSI but we don't know the UE.

I believe MAP has a provision for such an (optional) query between MSCs,
where the old MSC is looked up by a local state table based on the 'Old
LAI' information in the LU REQ.

Once a UE starts moving back and forth (e.g. at coverage boundary area
bettween MSCs), we may encounteer such situations.

> So this seems unrelated to 4G as such, and might also appear if a 2G subscriber is re-attaching with an unknown TMSI.
> This is a near impossible situation when only 2G/3G is available with exactly one osmo-msc running,

correct. But we do support inter-MSC-HO these days, so networks with
multiple (osmo or non-osmo) MSCs are realistic deployment scenarios.

> nevertheless osmo-msc should be able to resolve such situations without creating evil twins in the VLR.

ACK.

> IIUC, the only difference that 4G makes here is that the SGs interface may establish a new TMSI, which the phone subsequently uses towards 2G.

I don't think the original reporter indicated anything about SGs. Also,
SGs is entirely optional. It's legal to operate a 4G + 2G/3G network
which doesn't have an SGs interface. If you don't need CSFB nor
SMS-over-SGs (which both are entirely optional), you wouldn't need it.

#13 - 09/30/2019 03:51 PM - neels

MPoslusny wrote:

> @Neels, It seems your fix works in my case. Thank you for your efforts!

Could you elaborate which fix?
I have a patch on a branch (neels/vlr_evil_twin3) but that is lacking a substantial part.

This patch is telling the MSC to use the existing VLR record instead of the newly created twin,
but the twin should already have Location Updating FSM instances running which I guess we would have to move to the new VLR record.
So if it is this patch that is fixing the problem, I would be a bit surprised.

Hmm, maybe it could already work by failing a LU and working out the next time?

Anyway, are you talking about that branch, @MPoslusny?

#14 - 09/30/2019 04:09 PM - neels

neels wrote:

> but the twin should already have Location Updating FSM instances running which I guess we would have to move to the new VLR record.

s/new/old

#15 - 10/01/2019 05:22 AM - MPoslusny

I mean this fix
https://gerrit.osmocom.org/c/osmo-msc/+/15544/2/src/libvlr/vlr.c

The ugly backtrace hid but MS / UE can still connect.

#16 - 10/28/2020 08:28 PM - neels

- Related to Bug #4721: osmo-msc creates evil-twin entries in the VLR when an already attached IMSI does a LU by an unknown TMSI (was: MSC_Tests.TC_lu_by_tmsi_noauth_unknown fails sporadically locally) added

#17 - 10/28/2020 08:30 PM - neels

- Status changed from Stalled to Resolved

- % Done changed from 50 to 100

The root cause for the failure is the evil twin problem described in #4721 (which should still be fixed), track that bug there.
Since the failure to attach has been solved by a "workaround" (retry by the MS succeeds), closing this issue.