OsmoBTS - Bug #3506

FACCH LAPDm sequence error (state LAPD_STATE_MF_EST)

08/28/2018 10:24 AM - fixeria

Status:	Feedback	**Start date:**	08/28/2018
Priority:	High	**Due date:**	
Assignee:	fixeria	**% Done:**	0%
Category:	osmo-bts-trx		
Target version:			
Spec Reference:			

Description

During a voice call (either TCH/F, or TCH/H), any FACCH activity (DTMF, call hold/retrieve) causes the following messages:

```
<0011> lapd_core.c:1556 N(S) sequence error: N(S)=7, V(R)=0 (dl=0x7fa23038d478 state LAPD_STATE_MF
_EST) # Key press
<0011> lapd_core.c:1556 N(S) sequence error: N(S)=0, V(R)=1 (dl=0x7fa23038d478 state LAPD_STATE_MF
_EST) # Key release
...
<0011> lapd_core.c:1556 N(S) sequence error: N(S)=3, V(R)=4 (dl=0x7fa23038d478 state LAPD_STATE_MF
_EST) # Call hold
<0004> measurement.c:563 (bts=0,trx=0,ts=2,ss=0) No measurements for SUB!!!
<0004> measurement.c:563 (bts=0,trx=0,ts=2,ss=0) No measurements for SUB!!!
<0004> measurement.c:563 (bts=0,trx=0,ts=2,ss=0) No measurements for SUB!!!
<0004> measurement.c:563 (bts=0,trx=0,ts=2,ss=0) No measurements for SUB!!!
<0011> lapd_core.c:1556 N(S) sequence error: N(S)=4, V(R)=5 (dl=0x7fa23038d478 state LAPD_STATE_MF
_EST) # Call retrieve
...
```

osmo-bts-trx 08062e6dcc8a0d1f869a1ce0b88238c8218546c3
Observed with both OsmocomBB and a regular phone.

Please see A-bis/RSL traffic capture attached to this report.

Related issues:

Related to OsmoBTS - Feature #3906: Automatically adjust LAPDm timer values w...	**Resolved**	**04/06/2019**

History

#1 - 04/04/2019 10:33 AM - fixeria

- File lapdm_seq_error.pcapng.gz added

- Priority changed from Normal to High

I faced this issue again, while trying to send a MO SMS larger than 100 7-bit encoded characters:

```
DL1C NOTICE scheduler.c:623 Activating SDCCH/8(0) on trx=0 ts=1
DL1C NOTICE scheduler.c:623 Activating SACCH/8(0) on trx=0 ts=1
DL1C NOTICE scheduler.c:672 Set mode 3, 0, handover 0 on SDCCH/8(0) of trx=0 ts=1
DL1C NOTICE scheduler.c:737 Set a5/0 uplink for SDCCH/8(0) on trx=0 ts=1
DL1C NOTICE scheduler.c:737 Set a5/0 uplink for SACCH/8(0) on trx=0 ts=1
DL1C NOTICE scheduler.c:737 Set a5/0 downlink for SDCCH/8(0) on trx=0 ts=1
DL1C NOTICE scheduler.c:737 Set a5/0 downlink for SACCH/8(0) on trx=0 ts=1
DRSL NOTICE rsl.c:745 (bts=0,trx=0,ts=1,pchan=SDCCH8) (ss=0) SDCCH Tx CHAN ACT ACK
DLLAPD NOTICE lapd_core.c:920 Store content res. (dl=0x7f4aaa0eada8)
DLLAPD NOTICE lapd_core.c:1556 N(S) sequence error: N(S)=0, V(R)=1 (dl=0x7f4aaa0eada8 state LAPD_STATE_MF_EST)
DLLAPD NOTICE lapd_core.c:1556 N(S) sequence error: N(S)=0, V(R)=1 (dl=0x7f4aaa0eada8 state LAPD_STATE_MF_EST)
DLOOP INFO loops.c:199 (bts=0,trx=0,ts=1,ss=0) TOA is correct (94), keeping current TA of 0
DLLAPD NOTICE lapd_core.c:1556 N(S) sequence error: N(S)=0, V(R)=1 (dl=0x7f4aaa0eaf30 state LAPD_STATE_MF_EST)
DLLAPD NOTICE lapd_core.c:1556 N(S) sequence error: N(S)=1, V(R)=2 (dl=0x7f4aaa0eaf30 state LAPD_STATE_MF_EST)
DLOOP INFO loops.c:199 (bts=0,trx=0,ts=1,ss=0) TOA is correct (96), keeping current TA of 0
DLLAPD NOTICE lapd_core.c:1556 N(S) sequence error: N(S)=2, V(R)=3 (dl=0x7f4aaa0eaf30 state LAPD_STATE_MF_EST)
DLLAPD NOTICE lapd_core.c:1556 N(S) sequence error: N(S)=3, V(R)=4 (dl=0x7f4aaa0eaf30 state LAPD_STATE_MF_EST)
DLLAPD NOTICE lapd_core.c:1556 N(S) sequence error: N(S)=4, V(R)=5 (dl=0x7f4aaa0eaf30 state LAPD_STATE_MF_EST)
DL1C NOTICE scheduler.c:623 Deactivating SACCH/8(0) on trx=0 ts=1
```

```
DLLAPD NOTICE lapd_core.c:1284 S frame ignored in this state (dl=0x7f4aaa0eaf30)
DL1C NOTICE scheduler.c:623 Deactivating SDCCH/8(0) on trx=0 ts=1
DRSL NOTICE rsl.c:724 (bts=0,trx=0,ts=1,pchan=SDCCH8) (ss=0) SDCCH Tx CHAN REL ACK
```

Unlike DTMF, CP-DATA / RP-DATA from the MS doesn't even reach OsmoMSC, so it closes the connection due to inactivity.
Same problem can be observed during multi-part MO SMS submission.

Please see the attached capture of LAPDm (from the BTS), RSL and BSSAP.
Unfortunately, I know almost nothing about LAPDm internals.

laforge do you have any ideas?

#2 - 04/04/2019 01:17 PM - laforge

- Assignee set to laforge

#3 - 04/04/2019 04:38 PM - laforge

Initial analysis:

```
Frame 44: UL SAPI0 I: 16 bytes of L3 payload, 3 bytes of LAPDm header, 4 bytes padding
 * N(S)=0: UL transmit sequence number "0" (first frame after SABM/UA)
 * N(R)=1; acknowledge Rx of all frames up to number "0" in DL (none so far)

Frame 45: DL SAPI0 RR (Receiver Ready)
 * N(R)=1: BTS indicates it has received frames up to "0" in UL

Frame 46: UL SAPI0 retransmission of Frame 44 N(S)=0

Frame 48: DL SAPI0 REJ: Request retransmission of I frames starting with N(R)=1

[MS has no frames to re-transmit, as 1 is higher than 0, the last sent frame]

Frame 49: UL SAPI0 RR: MS states it has not sent any fames with seq > 0.
```

The odd part here is that wireshark somehow claims this is a fragment and hence it won't decode the actual payload, which supposedly is a CM SERVICE REQ.

SAPI3 / SMS

But let's look at the important part:

```
Frame 52: UL SAPI3 SABM: Establish SMS SAPI data link connection
Frame 53: RSL RLL EST IND (SAPI3) to BSC
Frame 55: DL SAPI3 UA: Acknowledge SMS SAPI data link connection
Frame 56: UL SAPI3 SABM: Retransmission of Frame 52
Frame 57: Another RSL RLL EST IND (SAPI3) to BSC (is that correct?)
Frame 58: DL SAPI3 UA: Acknowledge SMS SAPI data link connection

Frame 59: UL SAPI3 I: 20 bytes of L3 payload, 3 bytes of LAPDm header, 0 bytes padding
Frame 61: DL SAPI3 RR: Acknowledge Frame 59
Frame 62: UL SAPI3 I: Retransmission of Frame 59
Frame 64: DL SAPI3 REJ:

Frame 68: UL SAPI3 I: N(S)=1
Frame 69: DL SAPI3 RR: ACK of Frame 68
Frame 70: UL SAPI3 I: Retransmission of Frame 68
Frame 72: DL SAPI3 REJ:

Frame 75: UL SAPI3 I: N(S)=2
Frame 78: DL SAPI3 RR: ACK of frame 75
Frame 79: UL SAPI3 I: Retransmission of Frame 78
Frame 80: DL SAPI3 REJ:

Frame 84: UL SAPI3 I: N(S)=3
Frame 85: DL SAPI3 RR: ACK of Frame 84
Frame 86: UL SAPI3 I: Retransmission of Frame 85
Frame 89: DL SAPI3 REJ:
```

So what we can see is that somehow the latency/delay/queueing between the LAPDm entity on the MS and the LAPDm entity in the BTS is so high that the first ACK never arrives in time and every I frame is re-transmitted. This then takes so long that apparently the MSC decides to close the connection after 5 seconds.

Normally the 5s should be sufficient. However, if due to high radio link error ratio or due to implementation bugs or wrongly configured timers everything has to be transmitted twice, it is not long enough.

So there's at least two separate bugs:

- MSC 5s timeout to close the channel might be a bit short
- something is broken about this setup in terms of latency/queueing and the LAPDm timers. You either need to increase the related timers, or try to reduce latency.

What kind of physical setup are you running here?

#4 - 04/04/2019 04:38 PM - laforge

- Status changed from New to Feedback

- Assignee changed from laforge to fixeria

#5 - 04/04/2019 06:35 PM - fixeria

- File lapdm_seq_error_facch.pcapng.gz added

Hi Harald,

thanks for detailed analysis!

> So what we can see is that somehow the latency/delay/queueing between the LAPDm entity on the MS and the LAPDm entity in the BTS is so high that the first ACK never arrives in time and every I frame is re-transmitted. This then takes so long that apparently the MSC decides to close the connection after 5 seconds.

You're right. I just checked the logs of OsmoMSC again:

```
DMM DEBUG <0002> fsm.c:320 RAN_conn[0x6120000780a0]{RAN_CONN_S_NEW}: Allocated
DRLL DEBUG <0000> gsm_04_08.c:1473 Dispatching 04.08 message GSM48_MT_MM_CM_SERV_REQ (0x5:0x24)
DMM DEBUG <0002> ran_conn.c:735 RAN_conn(TMSI-0x22195893:GERAN-A-10:CM_SERVICE_REQ)[0x6120000780a0]{RAN_CONN_S
_NEW}: Updated ID
DMM DEBUG <0002> gsm_04_08.c:780 RAN_conn(TMSI-0x22195893:GERAN-A-10:CM_SERVICE_REQ)[0x6120000780a0]{RAN_CONN_
S_NEW}: Rx CM SERVICE REQUEST cm_service_type=0x04
DMM DEBUG <0002> ran_conn.c:735 RAN_conn(IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893:GERAN-A-10:CM_SERV
ICE_REQ)[0x6120000780a0]{RAN_CONN_S_NEW}: Updated ID
DMM DEBUG <0002> msc_ifaces.c:101 -> CM SERVICE ACCEPT IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893
DMM DEBUG <0002> vlr_access_req_fsm.c:150 RAN_conn(IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893:GERAN-A-
10:CM_SERVICE_REQ)[0x6120000780a0]{RAN_CONN_S_NEW}: Received Event RAN_CONN_E_ACCEPTED
DMM DEBUG <0002> ran_conn.c:123 RAN_conn(IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893:GERAN-A-10:CM_SERV
ICE_REQ)[0x6120000780a0]{RAN_CONN_S_NEW}: state_chg to RAN_CONN_S_ACCEPTED
DMM DEBUG <0002> ran_conn.c:239 RAN_conn(IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893:GERAN-A-10:CM_SERV
ICE_REQ)[0x6120000780a0]{RAN_CONN_S_ACCEPTED}: ran_conn_fsm_has_active_transactions: still awaiting first requ
est after a CM Service Request
DMM DEBUG <0002> ran_conn.c:568 RAN_conn(IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893:GERAN-A-10:CM_SERV
ICE_REQ)[0x6120000780a0]{RAN_CONN_S_ACCEPTED}: Received Event RAN_CONN_E_COMPLETE_LAYER_3

[here we stay for ~5 seconds]

DMM DEBUG <0002> fsm.c:210 RAN_conn(IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893:GERAN-A-10:CM_SERVICE_R
EQ)[0x6120000780a0]{RAN_CONN_S_ACCEPTED}: Timeout of T0
DMM DEBUG <0002> ran_conn.c:333 RAN_conn(IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893:GERAN-A-10:CM_SERV
ICE_REQ)[0x6120000780a0]{RAN_CONN_S_ACCEPTED}: Received Event RAN_CONN_E_CN_CLOSE
DMM NOTICE <0002> ran_conn.c:111 RAN_conn(IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893:GERAN-A-10:CM_SER
VICE_REQ)[0x6120000780a0]{RAN_CONN_S_ACCEPTED}: Close event, cause: CONGESTION
DMM DEBUG <0002> ran_conn.c:297 RAN_conn(IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893:GERAN-A-10:CM_SERV
ICE_REQ)[0x6120000780a0]{RAN_CONN_S_ACCEPTED}: state_chg to RAN_CONN_S_RELEASING
DMM DEBUG <0002> ran_conn.c:906 RAN_conn(IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893:GERAN-A-10:CM_SERV
ICE_REQ)[0x6120000780a0]{RAN_CONN_S_RELEASING}: Received Event RAN_CONN_E_UNUSED
DMM DEBUG <0002> ran_conn.c:408 RAN_conn(IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893:GERAN-A-10:CM_SERV
ICE_REQ)[0x6120000780a0]{RAN_CONN_S_RELEASING}: state_chg to RAN_CONN_S_RELEASED
DMM DEBUG <0002> ran_conn.c:415 RAN_conn(IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893:GERAN-A-10:CM_SERV
ICE_REQ)[0x6120000780a0]{RAN_CONN_S_RELEASED}: Terminating (cause = OSMO_FSM_TERM_REGULAR)
DRLL DEBUG <0000> ran_conn.c:552 IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893: Freeing RAN connection
DMM DEBUG <0002> ran_conn.c:415 RAN_conn(IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893:GERAN-A-10:CM_SERV
ICE_REQ)[0x6120000780a0]{RAN_CONN_S_RELEASED}: Freeing instance
DMM DEBUG <0002> fsm.c:402 RAN_conn(IMSI-262073993158656:MSISDN-123456:TMSI-0x22195893:GERAN-A-10:CM_SERVICE_R
EQ)[0x6120000780a0]{RAN_CONN_S_RELEASED}: Deallocated
```

> What kind of physical setup are you running here?

The MS is Nokia 1280, the BTS PHY is USRP B210. I have been running the latest master versions of osmo-trx-uhd, osmo-bts-trx. Both MS and BTS were placed on a small distance, so I didn't notice any xCCH decoding errors on the BTS side.

I've additionally tested a virtual setup using OsmocomBB (mobile, trxcon, fake_trx.py), and I didn't notice any LAPDm sequence errors (like we see with the physical setup)! However, the connection was also closed due to RAN_CONN_TIMEOUT.

Increasing RAN_CONN_TIMEOUT from 5 to 15 (just to be sure) helped to get SMS delivered in both OsmocomBB and Nokia 1280 cases. Meanwhile, I cannot reproduce those "sequence errors" with the Nokia phone anymore :/

In any case, sending DTMF over FACCH/F or FACCH/H still causes the following:

```
DLOOP INFO loops.c:199 (bts=0,trx=0,ts=2,ss=0) TOA is correct (126), keeping current TA of 0
DLOOP INFO loops.c:199 (bts=0,trx=0,ts=2,ss=0) TOA is correct (127), keeping current TA of 0
DLOOP INFO loops.c:199 (bts=0,trx=0,ts=2,ss=0) TOA is correct (128), keeping current TA of 0
DLLAPD NOTICE lapd_core.c:1556 N(S) sequence error: N(S)=2, V(R)=3 (dl=0x7fc2e0840478 state LAPD_STATE_MF_EST)
DLLAPD NOTICE lapd_core.c:1556 N(S) sequence error: N(S)=3, V(R)=4 (dl=0x7fc2e0840478 state LAPD_STATE_MF_EST)
DLOOP INFO loops.c:199 (bts=0,trx=0,ts=2,ss=0) TOA is correct (128), keeping current TA of 0
DLOOP INFO loops.c:199 (bts=0,trx=0,ts=2,ss=0) TOA is correct (129), keeping current TA of 0
DLOOP INFO loops.c:199 (bts=0,trx=0,ts=2,ss=0) TOA is correct (129), keeping current TA of 0
DLLAPD NOTICE lapd_core.c:1556 N(S) sequence error: N(S)=4, V(R)=5 (dl=0x7fc2e0840478 state LAPD_STATE_MF_EST)
DLLAPD NOTICE lapd_core.c:1556 N(S) sequence error: N(S)=5, V(R)=6 (dl=0x7fc2e0840478 state LAPD_STATE_MF_EST)
DLOOP INFO loops.c:199 (bts=0,trx=0,ts=2,ss=0) TOA is correct (130), keeping current TA of 0
DLOOP INFO loops.c:199 (bts=0,trx=0,ts=2,ss=0) TOA is correct (131), keeping current TA of 0
DLOOP INFO loops.c:199 (bts=0,trx=0,ts=2,ss=0) TOA is correct (131), keeping current TA of 0
DLLAPD NOTICE lapd_core.c:1556 N(S) sequence error: N(S)=6, V(R)=7 (dl=0x7fc2e0840478 state LAPD_STATE_MF_EST)
DLLAPD NOTICE lapd_core.c:1556 N(S) sequence error: N(S)=7, V(R)=0 (dl=0x7fc2e0840478 state LAPD_STATE_MF_EST)
DLOOP INFO loops.c:199 (bts=0,trx=0,ts=2,ss=0) TOA is correct (132), keeping current TA of 0
DLOOP INFO loops.c:199 (bts=0,trx=0,ts=2,ss=0) TOA is correct (132), keeping current TA of 0
DLOOP INFO loops.c:199 (bts=0,trx=0,ts=2,ss=0) TOA is correct (133), keeping current TA of 0
```

The corresponding capture file is attached.

#6 - 04/06/2019 06:19 PM - fixeria

- Related to Feature #3906: Automatically adjust LAPDm timer values with 'fn-advance' parameter added

Files

facch.pcapng.gz	3.41 KB	08/28/2018	fixeria
lapdm_seq_error.pcapng.gz	3.83 KB	04/04/2019	fixeria
lapdm_seq_error_facch.pcapng.gz	2.41 KB	04/04/2019	fixeria