OsmoBTS - Bug #4066

osmo-bts: sending BSC an ERROR Indication with "unsolicited UA response" in "sms:sysmo" osmo-gsm-tester test

06/20/2019 09:02 AM - pespin

Status:	Resolved	**Start date:**	06/20/2019
Priority:	Urgent	**Due date:**	
Assignee:	Hoernchen	**% Done:**	0%
Category:			
Target version:			
Spec Reference:			

Description

sms is broken according to osmo-gsm-tester:

```
10:49:03.831545 tst                          /sierra_1: sending sms to MSISDN 1022 {sms='message nr.
 1, from /sierra_1, to /sierra_2, from 1021, to 1022'}
10:54:04.203551 tst                   mo_mt_sms.py:41: ERR: Error: Wait timeout {condition=<bound m
ethod Modem.sms_was_received of /sierra_2>, timeout=300, timestep=1}  [trial↪sms:sysmo↪mo_mt_sms.p
y:41]
10:54:04.207759 tst                   mo_mt_sms.py:41: TRACEBACK: Traceback (most recent call last)
:
  File "/home/pespin/osmo-gsm-tester/src/osmo_gsm_tester/test.py", line 61, in run
    self.path)
  File "/home/pespin/osmo-gsm-tester/src/osmo_gsm_tester/util.py", line 350, in run_python_file
    spec.loader.exec_module( importlib.util.module_from_spec(spec) )
  File "<frozen importlib._bootstrap_external>", line 673, in exec_module
  File "<frozen importlib._bootstrap>", line 222, in _call_with_frames_removed
  File "/home/pespin/osmo-gsm-tester/suites/sms/mo_mt_sms.py", line 41, in <module>
    wait(ms_mt.sms_was_received, sms)
  File "/home/pespin/osmo-gsm-tester/src/osmo_gsm_tester/testenv.py", line 48, in <lambda>
    wait = lambda *args, **kwargs: event_module.wait(suite_run, *args, **kwargs)
  File "/home/pespin/osmo-gsm-tester/src/osmo_gsm_tester/event_loop.py", line 111, in wait
    raise log.Error('Wait timeout', condition=condition, timeout=timeout, timestep=timestep)
osmo_gsm_tester.log.Error: Wait timeout {condition=<bound method Modem.sms_was_received of /sierra
_2>, timeout=300, timestep=1}
  [trial↪sms:sysmo↪mo_mt_sms.py:41]
```

It seems to be related to LAPDm. BTS sending BSC an ERROR Indication with "unsolicited UA response" when sending the sms network->ms_mt.

See full_pcap_any.pcapng frame 1534.

Related issues:

Related to libosmocore - Bug #4074: LAPD timers completely broken	**New**	**06/21/2019**

History

#1 - 06/20/2019 09:36 AM - laforge

first of all, the GSMTAP trace is likely incomplete as there appears to be no GSMTAP listener, resulting in ICMP dest unreachable packets, which in turn I've seen many times make the kernel network stack fail to transmit some packets to those unreachable ports :(

This is why the libosmocore GSMTAP code typically creates both a transmitter and a receiver socket. That obviously only works if they are on localhost.

#2 - 06/20/2019 10:03 AM - laforge

unsolicited supervisory frame:

- Location Updates look rather normal and are complete by frame 1262
  - only oddity is some REJ frames (1219 + 1226)

- MO SABM for SAPI3 in frame 1317
- MT UA for SAPI3 in frame 1324
- I frame fragment in 1345
- RR for that I frame in 1336
- while SAPI3 is established, there are again some strange REJ frames on SAPI0 visible (1325 + 1337)
- the two ERR IND in 1338 + 1340 are related to SAPI0 and have "unsolicited supervisory response"

unsolicited UA

- SAPI0
  - 1499 is MO SABM with paging response
  - 1501 is RSL EST IND to BSC
  - we don't see a UA response from the BTS to the MS, but tha may be due to ICMP unreachable, see above
  - 1515 is a restransmission of the MO SABM
  - 1520 is a (retransmitted? UA back to the MS

- meanwhile, MT SAPI3 establishment triggered by paging response:
  - 1510 is the RSL EST REQ for SAPI3
  - 1513 the resulting SABM on Um
  - 1524 is the UA in response to the SABM
  - 1525 confirms the establishment to BSC on RSL
  - 1527 sends the first CP-DATA/RP-DATA from BSC to MS
  - 1531 is a retransmitted SABM from the MS, as the UA apparently didn't arrive [in time]
  - 1533 is the UA retransmission
  - 1534 is the ERR IND (unsolicited UA), **which is wrong, as the UA is simply a retransmission**

All of the above is - interestingly happening in 0.7 seconds. So I would guess somehow the timing (T200) is fucked up after recent changes

#3 - 06/20/2019 10:05 AM - laforge

- Assignee changed from laforge to pespin

- Priority changed from Normal to Urgent

I would suggest to try reverting 9f1fcc595c633e43869b16fea83dd4bfaa5a7892 and (if needed) also 5f85ed1a6383cb61b841dcedcc0666f50c95840c and f537b88d6193e10ee985ad25037b89e06fb100ee and give this test a shot again.

#4 - 06/20/2019 11:37 AM - laforge

osmo-bts change IDs If8babda9e3e5e219908911ddd9c0756b5ea0bca4, Ic1268ae2d769b12da6cdd4ac8375e4bc033a9e3e and I0e814fbae13e0feddd148c47255dcc38cb718f48, respectively.

#5 - 06/20/2019 01:14 PM - pespin

- Assignee changed from pespin to laforge

This seems to be the commit causing the regression:
"8ea9ba6af8317d4240d86d3f5dfb6c176b22a9b6 [correctly] use the LAPDm T200 values received via OML"

I had to revert the other 2 commits afterwards due to conflicts, but removing them alone doesn't fix the regression:
1ff0a2addd04de5bfe1601e84b013c65e500faf0
46d62b984a10b3966d8a0d0a435d4efab0e9c500

#6 - 06/20/2019 01:20 PM - pespin

- File successful_full_pcap_any.pcapng added

- File successful_run.tgz added

Attached successful run after revert for comparison.

#7 - 06/20/2019 03:27 PM - laforge

Then the T200 timer values are too short. This could have the following causes:

- they are simply configured too short in the BSC, or
- they somehow get corrupted during transmission/conversion over RSL (the Abis RSL encoding has different units like "steps of 5ms" or "steps of 20ms" depending on the channel, AFAIR), or
- the "fn_advance_us" is computed wrongly. This dynamically updated counter is supposed to determine the amount of time (in units of one GSM frame number) that passes between the L1SAP sending a downlink frame and that frame actually appearing on the radio interface. It is computed in code added in I8f739fdb808a614f080afbc4654641ec3df19eb2 which also added vty output for human introspection like

```
+       vty_out(vty, "  PH-RTS.ind FN advance average: %d, min: %d, max: %d%s",
+               bts_get_avg_fn_advance(bts), bts->fn_stats.min, bts->fn_stats.max, VTY_NEWLINE);
```

I'm not going to be able to spend time on this now or during holidays, sorry. Would be great if you could look into it. Reverting the change is the safe choice, but it also means that we're again using timeouts that are often something like an order of magnitude too large, resulting in very slow LAPDm performance in case of errors/retransmissiosn over real-world links.

One way to go about it is to play with the timeout values, run tests and experimentially determine from which values onwards the bug starts to appear. However, I think the results are not neccessarily useful, at least not directly. The hard-coded T200 values as sent by OsmoBSC right now work on the nanoBTS. So I espect that somehow we're converting them wrongly inside the BTS, or we're computing that fn_advance_us somehow wrongly.

Feele free to consult with others in the team like tnt, fixeria or @Hoernchen if you need help related to SDR or LAPDm questions.

#8 - 06/21/2019 10:32 AM - pespin

Looking at osmo-gsm-tester test trends, it seems "sysmo" and "oc2g" started failing with that commit, but "trx-uhd", "trx-umtrx", "trx-sysmocell5000" and "nanobts" are fine, so it's a bug specific to "sysmo" and "oc2g" then afaiu.

Interestingly though, nitb_sms:sysmo is working too, which means using same bts with NITB works fine...

#9 - 06/21/2019 11:47 AM - pespin

Those are the T200 values being set by BSC to sysmobts in the failing scenario:

```
20190428065409423 DOML <0001> oml.c:632 OC=BTS(01) INST=(00,ff,ff): T200[0]: OML=30, mult=5 => 150 ms
20190428065409424 DOML <0001> oml.c:632 OC=BTS(01) INST=(00,ff,ff): T200[1]: OML=36, mult=5 => 180 ms
20190428065409424 DOML <0001> oml.c:632 OC=BTS(01) INST=(00,ff,ff): T200[2]: OML=36, mult=5 => 180 ms
20190428065409424 DOML <0001> oml.c:632 OC=BTS(01) INST=(00,ff,ff): T200[3]: OML=168, mult=10 => 1680 ms
20190428065409424 DOML <0001> oml.c:632 OC=BTS(01) INST=(00,ff,ff): T200[4]: OML=52, mult=10 => 520 ms
20190428065409425 DOML <0001> oml.c:632 OC=BTS(01) INST=(00,ff,ff): T200[5]: OML=33, mult=5 => 165 ms
20190428065409425 DOML <0001> oml.c:632 OC=BTS(01) INST=(00,ff,ff): T200[6]: OML=168, mult=10 => 1680 ms
```

Same values are being sent by NITB to sysmobts during test "nitb_sms:sysmo", which actually succeeds.
And both are of course using the same sysmobts code build, so not sure where's the difference then making one fail and the other one succeed...
"sms:trx-b200" also shows same timer value, and that one is also working fine.

#10 - 06/21/2019 01:00 PM - laforge

Hi,

On Fri, Jun 21, 2019 at 10:32:08AM +0000, pespin [REDMINE] wrote:

> Looking at osmo-gsm-tester test trends, it seems "sysmo" and "oc2g" started failing with that commit, but "trx-uhd", "trx-umtrx", "trx-sysmocell5000" and "nanobts" are fine, so it's a bug specific to "sysmo" and "oc2g" then afaiu.

this would hint that somehow the "frame number advance" computation is working on OsmoTRX but failing on other BTS models, as it is the only part that is backend specific.

> Interestingly though, nitb_sms:sysmo is working too, which means using same bts with NITB works fine...

It might set different T200 values via OML, or it might deliver SMS over
different radio channel (TCH/FACCH vs. SDCCH)?

#11 - 06/21/2019 01:34 PM - pespin

Enabling debug output, on the failing setup:

```
20190429111656788 DLLAPD <0011> osmo-bts/src/common/bts.c:421 (bts=0,trx=0,ts=0,ss=4) Setting T200 D0=1028672,
 D3=2, S0=520, S3=520 (all in ms)
```

Values for DCCH look weird. SAPI0 is 1028 seconds, which looks extremly high. SAPI3 is 2 ms, which looks extremely low.

However, next values look better (that log line above is the first time the line is printed):

```
20190429111713489 DLLAPD <0011> osmo-bts/src/common/bts.c:421 (bts=0,trx=0,ts=0,ss=0) Setting T200 D0=150, D3=
165, S0=520, S3=520 (all in ms)
...
20190429111714500 DLLAPD <0011> osmo-bts/src/common/bts.c:421 (bts=0,trx=0,ts=0,ss=1) Setting T200 D0=150, D3=
165, S0=520, S3=520 (all in ms)
...
20190429111723356 DLLAPD <0011> osmo-bts/src/common/bts.c:421 (bts=0,trx=0,ts=0,ss=0) Setting T200 D0=150, D3=
165, S0=520, S3=520 (all in ms)
...
20190429111729698 DLLAPD <0011> osmo-bts/src/common/bts.c:421 (bts=0,trx=0,ts=0,ss=1) Setting T200 D0=150, D3=
165, S0=520, S3=520 (all in ms)
```

However, I see timeouts occurring quite quickly. That matches with the D0=150:

```
20190429 11:17 31.246 DLLAPD <0011> lapd_core.c:206 start T200 (dl=0xb6d75a50)
20190429 11:17 31.246 DLLAPD <0011> lapd_core.c:1861 k frames outstanding, not sending more (k=1 V(S)=2 V(A)=1
) (dl=0xb6d75a50)
20190429 11:17 31.396 DLLAPD <0011> lapd_core.c:562 Timeout T200 state=LAPD_STATE_MF_EST (dl=0xb6d75a50)
```

I also added this patch to log the value used for the timeout, and indeed it's using 150ms:
https://gerrit.osmocom.org/c/libosmocore/+/14566

```
20190429113354364 DLLAPD <0011> libosmocore/src/gsm/lapd_core.c:207 start T200 (dl=0xb6dd3498, timeout=0.15000
0s)
20190429113354364 DLLAPD <0011> libosmocore/src/gsm/lapd_core.c:1862 k frames outstanding, not sending more (k
=1 V(S)=1 V(A)=0) (dl=0xb6dd3498)
20190429113354514 DLLAPD <0011> libosmocore/src/gsm/lapd_core.c:563 Timeout T200 state=LAPD_STATE_MF_EST (dl=0
xb6dd3498)
```

#12 - 06/21/2019 01:52 PM - pespin

pespin wrote:

> Enabling debug output, on the failing setup:
> 20190429111656788 DLLAPD <0011> osmo-bts/src/common/bts.c:421 (bts=0,trx=0,ts=0,ss=4) Setting T200 D0=1028672, D3=2, S0=520, S3=520 (all in ms)
>
> Values for DCCH look weird. SAPI0 is 1028 seconds, which looks extremly high. SAPI3 is 2 ms, which looks extremely low.

After debugging further: That's for channel GSM_LCHAN_CCCH, which has no case in the switch statement in t200_by_lchan(), so the values for t200_ms_dcch are actually not set. Not sure if that's expected.

#13 - 06/21/2019 02:40 PM - laforge

Hi Pau,

as I indicated, the **adjustment/advancement** value that is added inside the BTS to those
values to compensate for the delays between L1SAP and RF is likely wrong for non-TRX models,
explaining the difference.

> Same values are being sent by NITB to sysmobts during test "nitb_sms:sysmo", which actually succeeds.

the question is "how does it succeed". If the T200 values are the same, you should see the ssme
degree of unintended retransmissions on the Um/GSMTAP side, i.e. "REJ frames, retransmitted SABM/UA frames
,... - it may just be that the NITB is waiting longer before giving up on a higher layer.

"The SMS arrived after way too many retransmissions which wastes radio resources" is also a failure, whether or not the specific high-level test passes or not.

#14 - 06/21/2019 04:16 PM - pespin

- File stdout added

Added osmo-bts-sysmo output (stdout file) which shows with extra logging that in sysmobts the fn_advance calculation is always 0 due to delta being zero because both values used to calculate it are always up-to-date with each other. Grep for "updated fn_stats".

#15 - 06/21/2019 05:27 PM - laforge

On Fri, Jun 21, 2019 at 04:16:52PM +0000, pespin [REDMINE] wrote:

Added osmo-bts-sysmo output (stdout file) which shows with extra logging that in sysmobts the fn_advance calculation is always 0 due to delta being zero because both values used to calculate it are always up-to-date with each other.

The assumption was that bts->gsm_time reflects the time of the last
Ph-DATA.ind, i.e. time received with data. PH-RTS.ind contains the
"advanced time" to proactively generate downlink data in advance.

The only place where bts->gsm_time is written is l1sap_info_time_ind()
which happens whenever the bts-specific part sends a PRIM_INFO_TIME.
This is called:

- for osmo-trx, from trx_sched_fn(), right before advancing the FN and generating the PH-RTS.ind
- for osmo-bts-{sysmo,lc15,oc2g}, whenever a GsmL1_PrimId_MphTimeInd is received
- for osmo-bts-octphy, whenever a mOCTVC1_GSM_MSG_TRX_TIME_INDICATION_EVT_SWAP is received

So it boils down to the semantics of those 'TIME IND' from the other BTS
models. I guess it makes sense to introduce a new a new
'bts->last_data_ind_time' which is updated in the common part above
l1sap every time we receive a PH-DATA.ind ?

#16 - 06/21/2019 05:36 PM - laforge

- Related to Bug #4074: LAPD timers completely broken added

#17 - 06/21/2019 05:42 PM - laforge

- Checklist item [ ] make osmo-bts ignore T200 from OML again added
Checklist item [ ] increase T200 default values again added

- Assignee changed from laforge to Hoernchen

assigning to Hoernchen who will build a patch to re-introduce the old loooong T200 values (to work around #4074) and ignore the values sent from the BSC via OML as an interim measure.

Independent of the above, we also need to fix the fn_advance_ms logic (see checklist).

#18 - 06/21/2019 05:43 PM - Hoernchen

- Checklist item [x] make osmo-bts ignore T200 from OML again set to Done
Checklist item [x] increase T200 default values again set to Done

#19 - 06/21/2019 05:44 PM - Hoernchen

Timeout reverted in https://gerrit.osmocom.org/c/osmo-bts/+/14567

#20 - 06/21/2019 05:45 PM - laforge

- Checklist item [ ] make osmo-bts ignore T200 from OML again set to Not done
Checklist item [ ] increase T200 default values again set to Not done

- File 20190620-t200.gnumeric added

attaching a gnumeric spreadsheet for the T200 values encountered for BS-11 and nanoBTS in osmo-bsc

#21 - 06/24/2019 12:50 PM - Hoernchen

- Checklist item [x] make osmo-bts ignore T200 from OML again set to Done
Checklist item [x] increase T200 default values again set to Done

#22 - 06/25/2019 08:57 AM - pespin

- Status changed from New to Resolved

"sms:sysmo" is passing again, and we have other tasks to improve the general situation with regards to T200, so closing this ticket.

Files

full_pcap_any.pcapng	624 KB	06/20/2019	pespin
run.tgz	205 KB	06/20/2019	pespin
successful_full_pcap_any.pcapng	404 KB	06/20/2019	pespin
successful_run.tgz	136 KB	06/20/2019	pespin

stdout	13 MB	06/21/2019	pespin
20190620-t200.gnumeric	2.21 KB	06/21/2019	laforge