OsmoBSC - Bug #3041

osmo-bsc initiates BSSMAP Reset after every fourth BSSMAP Clear

03/08/2018 12:06 AM - neels

Status:	Rejected	**Start date:**	03/07/2018
Priority:	High	**Due date:**	
Assignee:	dexter	**% Done:**	10%
Category:			
Target version:			
Spec Reference:			

Description

dexter**, can you take a look and clarify whether the FSM is wrong or the error is elsewhere?**

Testing with current master of osmo-bsc (f93970b167aba2805cc67e1326591f31fbe93ada) I get spurious BSSMAP Reset messages coming from OsmoBSC.
They seem to happen exactly two seconds after a BSSMAP Clear Complete message; though not after *every* BSSMAP Clear, "feels like" every fourth Clear or so.
The log always states "Timeout of T1234" shortly before the Reset. The Resets do not re-occur at regular intervals by themselves, only after a BSSMAP Clear Complete.

See attached pcap, get an overview first by filtering 'bssap', then pick a 'BSSMAP Reset' packet and clear the filter to also see the OsmoBSC gsmtap-log around it.
I filtered RTP and OsmoMGW logging from the pcap to reduce the size (the voice calls worked fine).
In the pcap, two phones subscribe, call each other, do a USSD, detach, re-attach and so on. Every so often an out-of-place BSSMAP Reset occurs.

Related issues:

Related to OsmoBSC - Feature #3102: clean up and use a_reset.c in osmo-bsc	**Resolved**	**03/23/2018**
Related to OsmoBSC - Bug #4188: BSC sends COMPLETE L3 before RESET	**Rejected**	**09/03/2019**

History

#1 - 03/08/2018 12:52 AM - neels

T1234 is

```
#define RESET_RESEND_TIMER_NO 1234      /* FIXME: dig out the real timer number */
```

from osmo-bsc/src/osmo-bsc/osmo_bsc_reset.c

and the two seconds almost certainly come from

```
#define RESET_RESEND_INTERVAL 2 /* sec */
```

which are triggered at three osmo_fsm_inst_state_chg(fi, ST_DISC, RESET_RESEND_INTERVAL, RESET_RESEND_TIMER_NO); in osmo_bsc_reset.c.

The cause for starting T1234 in this instance is EV_N_DISCONNECT sent to the A-CONNECTION FSM (osmo_bsc_reset.c), dispatched from a_reset.c:183 a_reset_conn_fail(), see packet 7640.
7639 says

```
ERROR    OsmoBSC            sccp connection (id=3) not cleared (gsm subscriber connection still active) -- for
cefully clearing it now!
```

from osmo_bsc_sigtran.c:364 osmo_bsc_sigtran_del_conn().

The above came from osmo_bsc_sigtran.c:226 in sccp_sap_up(), as logging from line 223 indicates:

```
N-DISCONNECT.ind(3, cause=768)
```

Looking from further back,

- the OsmoBSC forwards a CC Release Complete to MSC (packet 7446)
- the MSC receives it, the Subscr_Conn FSM terminates (packet 7481) and MSC sends a BSSMAP Clear Command (packet 7503)
- BSC receives it (7536) and sends a) a DLCX to the MGW (7544) and b) an RR Channel Release towards BTS (7550).
- MGW replies with DLCX OK, which triggers a Tx Clear Complete to the MSC (7555).
- MSC thus releases the SCCP connection (7597): hands an N-DISCONNECT.request to its SCCP conn FSM.
- the BSC receives a RLSD (SCCP message "Released") from the MSC (7624)
- the BSC though decides that "sccp connection (id=3) not cleared (gsm subscriber connection still active) -- forcefully clearing it now!" (7639) (osmo_bsc_sigtran.c:364)
  - [edit] it actually seems that this is a normal sequence of events, osmo_bsc_sigtran_del_conn() simply always logs an error like this.
- the BSC's A-Reset FSM receives the EV_N_DISCONNECT
  - [edit] and if it has received three EV_N_DISCONNECT in a row, the FSM transitions to ST_DISC ("Disconnect") which leads to a BSSMAP Reset after two seconds:
- thus the A-Reset starts its T1234, probably to timeout on receiving the "Released" message
- T1234 times out and the BSC invokes a BSSMAP Reset to kill all humans. This seems a bit drastic: Reset the entire A-link because one SCCP conn was found to be released.

So how **should** this be happening instead? It seems to me the BSC should notice that the subscriber conn is indeed no longer active since the BSC itself acked that by a BSSMAP Clear Complete message.

#2 - 03/08/2018 01:42 AM - neels

When I compare a sequence leading to a BSSMAP Reset with one that doesn't lead to a BSSMAP Reset, I notice that my above interpretation of the logging was a bit wrong.
(I will shortly edit above post to be more accurate, which will not be visible in mails IIRC, only on the ticket website.)

The "successful" case:

- BSC receives the SCCP RLSD (packet 12339)
- Says "ERROR OsmoBSC sccp connection (id=6) not cleared (gsm subscriber connection still active) -- forcefully clearing it now!" (12354)
  - this actually seems to be a normal sequence of events, and should not be logged as an error at all.
- The A-RESET FSM also receives the EV_N_DISCONNECT, but this time it doesn't change the states:

The main difference comes from this code in a_reset.c:

```
#define RESET_RESEND_INTERVAL 2 /* sec */
#define RESET_RESEND_TIMER_NO 1234      /* FIXME: dig out the real timer number */
#define BAD_CONNECTION_THRESOLD 3       /* connection failures */

[...]

/* Connected state */
static void fsm_conn_cb(struct osmo_fsm_inst *fi, uint32_t event, void *data)
{
        struct a_reset_ctx *reset = (struct a_reset_ctx *)data;
        OSMO_ASSERT(reset);

        switch (event) {
        case EV_N_DISCONNECT:
                if (reset->conn_loss_counter >= BAD_CONNECTION_THRESOLD) {
                        LOGPFSML(reset->fsm, LOGL_NOTICE, "SIGTRAN connection down, reconnecting...\n");
                        osmo_fsm_inst_state_chg(fi, ST_DISC, RESET_RESEND_INTERVAL, RESET_RESEND_TIMER_NO);
                } else
                        reset->conn_loss_counter++;
                break;
        case EV_N_CONNECT:
                reset->conn_loss_counter = 0;
                break;
        }
}
```

So for each EV_N_CONNECT, the conn_loss_counter is set to 0. Then each EV_N_DISCONNECT is seen as a connection loss (instead of a planned release) and with the third release, we say "SIGTRAN connection down" and go to state ST_DISC ("Disconnect"). But this is the A-interface. The connect and disconnect events don't necessarily come in pairs. Maybe ten SCCP links get established in a row, then a bit later all ten are done with their business and disconnect. The logic of the conn_loss_counter completely evades me.

Maybe the EV_N_DISCONNECT was planned to be an unplanned connection loss? Then these sequence of events, which are clearly a planned release of a connection, should not lead to EV_N_DISCONNECT? Then osmo_bsc_sigtran_del_conn() should obviously differentiate between planned and inadvertent connection loss; right now it always seems to indicate error.

Even if an inadvertent connection loss leads to the entire A interface going into Reset: what justifies counting to three? If one of our connections fails to communicate, then we wait for two more, and only then do we mind, and rip down all other connections that might still be going on fine? I don't get it.

#3 - 03/08/2018 01:47 AM - neels

- Assignee set to neels

- % Done changed from 0 to 10

need to ask pmaier about it, to clarify whether something is going wrong or whether the logic of the code is wrong.

#4 - 03/08/2018 01:47 AM - neels

- Status changed from New to In Progress

- Priority changed from Normal to High

#5 - 03/08/2018 03:30 AM - neels

This is with the following software versions:

===== osmo-bsc =====
338897f9fb617ea34894b6d9baeb8fa9ed5fa1d6
--> current master (f93970b167aba2805cc67e1326591f31fbe93ada) with a few cosmetic patches on top

===== libosmocore =====
ed3afa77701591f4f74a39dc8633e4eec25f804a

===== libosmo-sccp =====
3137be99ef2e75bd5bdd616a6c435513a64125ec

===== libosmo-abis =====
61460fd6431d6ea62752d74ad05425f132d7abbe

===== osmo-msc =====
48d4ec06e180cfb60556ce6c565620078bbea8db

===== libosmo-netif =====
525256a15a581daec7afd9edd65f10b827ff2f51

===== osmo-bts-sysmo =====
0.7.0.38-ef8c (first BTS) and
0.7.0.32-3c96d (second BTS)

#6 - 03/08/2018 07:50 AM - laforge

Could this somehow be related to #3035 ? It was the most recent change related to
BSSAmAP RESET (although on the MSC side...)

#7 - 03/08/2018 12:49 PM - neels

laforge wrote:

> Could this somehow be related to #3035 ?

No, that seems unrelated to me. Though I'm not sure whether abovementioned conn_loss_counter is involved in detecting an MSC restart.
In any case, the MSC was behaving well, and Reset Requests were only issued by the BSC in the midst of normal operation, without an MSC restart involved.

#8 - 03/08/2018 01:05 PM - neels

- File bssmap_reset_after_every_4th_bssmap_clear.pcapng added

In another manual test I add logging of the conn_loss_counter and do USSD requests from a single subscriber. Exactly after every fourth BSSMAP Clear, the FSM goes into BSSMAP Reset. The log also shows that the conn_loss_counter is never reset to zero when a new CM Service Request comes in. See attached pcap.

Hence it should be fairly easy to write a ttcn3 test against this.

#9 - 03/08/2018 01:08 PM - neels

- Subject changed from osmo-bsc initiates BSSMAP Reset apparently for no reason to osmo-bsc initiates BSSMAP Reset after every fourth BSSMAP Clear

- Status changed from In Progress to Feedback

- Assignee changed from neels to dexter

dexter, can you take a look and clarify whether the FSM is wrong or the error is elsewhere?

#10 - 03/10/2018 04:26 AM - neels

drafted a ttcn3 test in osmo-ttcn3-hacks.git on branch neels/os3041 but it is obviously not doing what I want yet.
Noob question: how do I make that alt statement "go in the background" instead of the test halting completely and running into T_guard?

#11 - 03/10/2018 08:30 AM - laforge

Hi Neels,

On Sat, Mar 10, 2018 at 04:26:49AM +0000, neels [REDMINE] wrote:

> drafted a ttcn3 test in osmo-ttcn3-hacks.git on branch neels/os3041 but it is obviously not doing what I want yet.
> Noob question: how do I make that alt statement "go in the background" instead of the test halting completely and running into T_guard?

you have to turn the alt into an altstep, and then 'activate' this altstep.

See e.g. "d := activate(pingpong());" in the GGSN_Tests test suite which activates
an altstep automatically answering with GTP PONG to GTP PING

#12 - 03/11/2018 01:25 AM - neels

just fyi, I use this brute osmo-bsc.git hack to disable the conn_loss_counter during testing of current patches, so that no odd BSSMAP Resets get in the way:

```
--- a/src/libbsc/a_reset.c
+++ b/src/libbsc/a_reset.c
@@ -71,10 +71,12 @@ static void fsm_conn_cb(struct osmo_fsm_inst *fi, uint32_t event, void *data)

        switch (event) {
        case EV_N_DISCONNECT:
+#if 0
                if (reset->conn_loss_counter >= BAD_CONNECTION_THRESOLD) {
                        LOGPFSML(reset->fsm, LOGL_NOTICE, "SIGTRAN connection down, reconnecting...\n");
                        osmo_fsm_inst_state_chg(fi, ST_DISC, RESET_RESEND_INTERVAL, RESET_RESEND_TIMER_NO
                } else
+#endif
                        reset->conn_loss_counter++;
                break;
        case EV_N_CONNECT:
```

#13 - 03/11/2018 03:08 AM - neels

about the ttcn3 test: now I have
http://git.osmocom.org/osmo-ttcn3-hacks/commit/?h=neels/os3041&id=032f5837022dc740fefb4b119c672a93b74ddb37
and I get

```
02:38:03.910093 mtc BSC_Tests.ttcn:239 Message with id 2 was extracted from the queue of BSSAP.
02:38:03.910161 mtc BSC_Tests.ttcn:240 setverdict(fail): none -> fail reason: "unexpected BSSMAP Reset", new c
omponent reason: "unexpected BSSMAP Reset"
02:38:03.910208 mtc BSC_Tests.ttcn:318 Default with id 2 (altstep no_bssmap_reset) finished. Skipping current
alt statement or receiving operation.
02:38:03.910259 mtc BSC_Tests.ttcn:323 Dynamic test case error: Copying an unbound value of type @RSL_Types.RS
L_Message.
02:38:03.910326 mtc BSC_Tests.ttcn:323 setverdict(error): fail -> error
```

So I receive the BSSMAP Reset I don't want to see, and it goes to 'fail'. Fair enough, that's the result I want.
Then what,
"Copying an unbound value of type @RSL_Types.RSL_Message."
without any information on what might be unbound anywhere; fail -> error.
Any hints?

Also the osmo-bsc log looks different from what I'd expect.
First "problem" are the continuous BSSMAP Reset firing before and after the actual test execution,
need to ignore them and try to find the timespan when the test was actually running,
and determine whether one of those BSSMAP Reset is the one I want to trigger with the test: kind of hard.

Then from triggering the failure, I actually expect to see the message
DMSC NOTICE "SIGTRAN connection to MSC No.: %i down, reconnecting...",
which doesn't appear in the osmo-bsc.log.
So I might still be triggering my desired failure on one of those "normal" BSSMAP Resets,
and my RLSD messages from the MSC side may not have the desired effect at all.
Maybe I should be playing out more of a "real" situation...?

All in all I get the impression that I'm not understanding yet.
Could use some titan help I guess, if only to reassure that I'm on the right track,
and about debugging above "unbound value".

#14 - 03/11/2018 07:16 PM - laforge

On Sun, Mar 11, 2018 at 03:08:43AM +0000, neels [REDMINE] wrote:

> 02:38:03.910259 mtc BSC_Tests.ttcn:323 Dynamic test case error: Copying an unbound value of type @RSL_Types.RSL_Message.

Check line 323 of your source file. You appear to be using some kind RSL_Message there which
isn't properly bound. If you declare a variable, it's completely unbound. If yo ever want to
send it, all fields must be bound (assigned) to a known value.

This way TTCN-3 prevents you from ever sending the equivalent of "uninitialized memory"

if you simply log() the variable above line 323 it will show you which fields are <unbound> in the
log output.

#15 - 03/12/2018 03:46 AM - neels

ttcn3 tests: https://gerrit.osmocom.org/7223https://gerrit.osmocom.org/7224https://gerrit.osmocom.org/7219

#16 - 03/12/2018 03:47 AM - neels

- Description updated

#17 - 03/23/2018 01:27 PM - neels

- Status changed from Feedback to Rejected

Update: The new gscon FSM in osmo-bsc has changed the picture.
Before the FSM, a_reset_conn_fail() used to be called basically on every BSSMAP Clear.
Now a_reset_conn_fail() is never invoked anymore. That's why BSC_Test.TC_bssap_rlsd_does_not_cause_bssmap_reset now passes.

(The other "_does_not_cause_bssmap_reset" tests fail for other reasons.)

Technically, this issue here is resolved though, by not calling the reset FSM anymore.
Opening a new issue about the remaining problems of abandoned reset FSM: #3102.

#18 - 03/23/2018 01:30 PM - neels

- Related to Feature #3102: clean up and use a_reset.c in osmo-bsc added

#19 - 09/04/2019 01:31 PM - pespin

- Related to Bug #4188: BSC sends COMPLETE L3 before RESET added

Files

osmo-bsc_does_bssmap_reset2.pcapng	4.7 MB	03/08/2018	neels
bssmap_reset_after_every_4th_bssmap_clear.pcapng	66 KB	03/08/2018	neels