Cellular Network Infrastructure - Bug #4667

mgcp_client_fsm 'terminate' logic has no timeout

07/15/2020 09:20 AM - laforge

Status:	Resolved	**Start date:**	07/15/2020
Priority:	Normal	**Due date:**	
Assignee:	dexter	**% Done:**	100%
Category:			
Target version:			
Spec Reference:			

Description

While reading the mgcp_client_fsm code I noticed the 'terminate' logic whihc delays sending a DLCX if some other operation is still pending.

I don't really understand why this is required (it's perfectly legal to send MGCP commands without waiting for the response first, right?) - the problem is that there appears to be no related timeout. So if we set that 'terminate' flag, and we never get a response to whatver previous MGCP ommand, then we are stuck in this state indefinitiely and we never send the DLCX.

The main point of osmo_fsm was to introduce mechanisms to avoid such 'stuck' situations and to ensure there always is a timer running to recover.

Maybe my reading of the code os wrong, please clarify.

History

#1 - 09/07/2020 02:49 PM - dexter

- Status changed from New to In Progress

- % Done changed from 0 to 50

I have reread the code. I do not know exactly anymore why I implemented the wait logic. It would be legal to send the DLCX directly, however, I guess it seemed logical to wait until one operation is complete before carrying out another.

I have checked back the timeout problem. The problem exists in fsm_crcx_resp_cb (ST_CRCX_RESP) and fsm_mdcx_resp_cb (ST_MDCX_RESP).

Lets look at ST_CRCX_RESP: When EV_CRCX_RESP never arrives because the MGW never answeres then the FSM seems to be stuck. ST_CRCX_RESP is entered from ST_CRCX (fsm_crcx_cb), where I find the line: osmo_fsm_inst_state_chg(fi, ST_CRCX_RESP, MGCP_MGW_TIMEOUT, MGCP_MGW_TIMEOUT_TIMER_NR). The state change is guarded by a timeout then. To me this looks sane.

#2 - 09/14/2020 10:26 AM - dexter

- Status changed from In Progress to Resolved

- % Done changed from 50 to 100