Cellular Network Infrastructure - Bug #4499

Verify we set TCP_NODELAY socket option where needed

04/17/2020 12:09 PM - pespin

<table>
<thead>
<tr>
<th>Status:</th>
<th>New</th>
<th>Start date:</th>
<th>04/17/2020</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
<td>Due date:</td>
<td></td>
</tr>
<tr>
<td>Assignee:</td>
<td>pespin</td>
<td>% Done:</td>
<td>0%</td>
</tr>
<tr>
<td>Category:</td>
<td></td>
<td>Spec Reference:</td>
<td></td>
</tr>
<tr>
<td>Target version:</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Description**

Recently it was spotted that we were not setting TCP_NODELAY option in OML and RSL connections, which means some messages being sent can be delay by considerable amounts of times. This kind of behavior is not beneficial in the context of signalling we do where latency is far more important than amount of data / network performance.

This ticket is a reminder to look for all protocols we implement that use TCP and verify whether it makes sense to enable TCP_NODELAY sockopt. Some I can think of include:

- Abis (bts and bsc side)
- OML (bts and bsc side)
- GSUP (client and server side).

```diff
diff --git a/src/input/ipaccess.c b/src/input/ipaccess.c
index e1936af..4d21416 100644
--- a/src/input/ipaccess.c
+++ b/src/input/ipaccess.c
@@ -271,6 +271,7 @@ static int ipaccess_rcvmsg(struct e1inp_line *line, struct msgb *msg,
   struct osmo_fd *newbfd;
 struct e1inp_line *new_line;
+   int on = 1;

   +/* set TCP_NODELAY (FIXME: just call update_fd_settings() here instead?) */
   ret = setssockopt(newbfd->fd, IPPROTO_TCP, TCP_NODELAY, &on, sizeof(on));
   if (ret < 0)
     LOGP(DLINP, LOGL_ERROR, "Failed to set TCP_NODELAY: \%s\n", strerror(errno));
```

**History**

05/15/2020
#1 - 04/17/2020 01:02 PM - osmith

- Assignee changed from osmith to pespin

(I'm on holiday next week, so assigning back to you as discussed.)