Support for SCTP multi-homing

My humble assumption here is that this is done entirely in the kernel. It will automatically learn about the peers’ additional addresses and use them as needed.

In OsmoSTP, the VTY would have to allow for manual specification of multiple IP addresses for each peer, so that during start up all of those addresses are attempted (think of the case where the normal/primary address is unreachable at link-bringup time). After the association is up, SCTP exchanges information about the additional IPs dynamically.

The actual switch between using the addresses is entirely done in the kernel, we just get notifications (SCTP_PEER_ADDR_CHANGE) about this and can log them.

My humble assumption here is that this is done entirely in the kernel. It will automatically learn about the peers’ additional addresses and use them as needed.

In my experience the kernel will need a bit of help before the multi-homing works as we expect it to.

Consider a scenario where the STP runs on a server with 4 IPs (A, B, C and D). A and B are used for signaling, but C and D are firewalled and used for OAM/local SSH logins.

If we open a multi-homed SCTP connection to another host with kernel defaults, then the kernel (not knowing about our overall network design) will associate all 4 IPs (A, B, C and D) with this connection.

As a consequence, the peer will successfully communicate with our IPs A and B - but endless try (and fail) to communicate with our IPs C and D.

Therefore it can make sense to explicitly specify:
1. Which peer IPs this SCTP association connects to
2. Which IPs we bind on our end of the SCTP association

This explicit binding of IPs on our side can done by using sctp_bindx() on the fd before calling sctp_connectx().

thanks a lot for your most valuable input.

Assignee set to pespin
The SCTP protocol provides built-in functionality for multi-homing, that is multiple IP addresses on either side of the SCTP association. This functionality is present in the Linux kernel SCTP implementation, which in turn is used by osmo-stp. However, osmo-stp currently does not provide any means to configure which of the hosts IP addresses are used for what particular AS or ASP. The implementation of this feature will include

1. Extension of the VTY interface to configure multiple local and remote IP addresses
2. Extension of the osmo-stp internal data structures to handle multiple IP addresses
3. Corresponding configuration of the Linux kernel SCTP stack using its socket API from osmo-stp
   1. local IP addresses are used for the sctp_bindx() socket API call
   2. remote IP addresses are used for the sctp_connectx() socket API call
4. Implementation of automatic test cases in TTCN-3

For point 3-1, please limit the local IP addresses to those actually some SCTP application. One of our vendors actually has this bug (never fixed) where sctp_bindx() picks up all the local addresses (including localhost, etc) that do not service SCTP, and that messes up multi-homing.

In osmo-* programs, SCTP socket is handled over osmo_stream_*.

We are using one-to-one style socket paradigm there afaict (https://tools.ietf.org/html/draft-ietf-tsvwg-sctpsocket-15#section-4.1.1).

SCTP osmo_stream_srv_link is configured in osmo_ss7_xua_server_create(), and osmo_ss7_xua_server_bind() calls osmo_stream_srv_open(), which in turn calls osmo_sock_init(AF_INET, SOCK_STREAM, link->proto,... (link->proto=IPPROTO_SCTP), which creates the socket() and calls listen() (meaning new SCTP associations can be done at this point).

SCTP osmo_stream_cli is configured in libosmo-sccp.git osmo_ss7_asp_restart(). osmo_stream_cli_open2() calls osmo_sock_init2(AF_INET, SOCK_STREAM, cli->proto,... (cli->proto=IPPROTO_SCTP)

SCTP osmo_stream_srv i processed through xua_srv_conn_cb() in libosmo-sccp. In there there's a call to sctp_recvmsg(). Sending in done in osmo_ss7_asp_send() which calls osmo_stream_srv_send() and enqueue the msg, and eventually osmo_stream_cli_write() will call sctp_send(). In this later case, we do stuff with SCTP_ASSOC_CHANGE and other notifications.

SCTP send() is in osmo_stream_srv_write()
SCTP sctp_recvmsg() is in osmo_stream_svc_recv()

As per what I understand so far, dummy multi-homing is already supported right now in the generic socket API (in clients like BSC and MSC, not sure about servers):

If addr is specified as a wildcard (INADDR_ANY for an IPv4 address, or as IN6ADDR_ANY_INIT or in6addr_any for an IPv6 address), the operating system will associate the endpoint with an optimal address set of the available interfaces.

Similarly for connect (we always do bind() before thought, but in case it's bind(ANY_ADDR):

If a bind() is not called prior to the connect() call, the system picks an ephemeral port and will choose an address set equivalent to binding with INADDR_ANY and IN6ADDR_ANY for IPv4 and IPv6 socket respectively. One of those addresses will be the primary address for

In osmo-bsc/osmo-msc (osmo_stream_clc), if no "local-ip" under "asp" node is used ("0.0.0.0"), the above will happen.
In osmo-stp (osmo_stream_srv), if no "local-ip" under "listen" node is used (aka "0.0.0.0"), the above will happen.

With that config, I can see in my case both the STP and the BSC/MSC announcing all the IP addrs on my system and later on attempting HEARTBEATS for each of them.
So the new APIs are really needed to create specifics sets for local addresses (bind). From VTY user point of view, simply allowing several "local-ip" cmds (first one being primary) and "no local-ip" to remove them should be sufficient. Maybe adding an optional parameter "[primary]" at the end of "local-ip" cmd.

Regarding decision on who to send:

[msg_name] is used to indicate a preferred peer address if the sender wishes to discourage the stack from sending the message to the primary address of the receiver. If the transport address given is not part of the current association, the data will not be sent and a SCTP_SEND_FAILED event will be delivered to the application if send failure events are enabled.

So in theory we should have a ["primary"] option in "remote-ip" too, or even some param specific order 1...N in order to decide easily which remote IP addr to send messages. That'd be used when scpt_connectx also.

I still need to read part of the socket API to better find what and how I need to implement it (I'm at point "5.2. SCTP msg_control Structures" right now).

#9 - 10/10/2019 11:04 AM - pespin

It seems ip addr list in scpt_sendx() is actually only used for implicit connection if no connect() was called before it, so we cannot really specify that time which dst addr to use. Moreover, specs says:

The list of addresses is provided for implicit association setup. In such a case the list of addresses serves the same purpose as the addresses given in scpt_connectx (see Section 8.9).

This is also interesting:

The system uses stream 0 as the default stream for send() and sendto(), recv() and recvfrom() return data from any stream, but the caller can not distinguish the different streams.

There's also:

7.1.10. Set Primary Address (SCTP_PRIMARY_ADDR)

Requests that the local SCTP stack use the enclosed peer address as the association primary. The enclosed address must be one of the association peer's addresses.

So after all what we need is probably accepting multiple "local-ip" and "remote-ip", and perhaps adding an optional ["primary"] cmd to be able to use SCTP_PRIMARY_ADDR on that IP addr. If no primary param is provided, I'd expect the first "remote-ip" line (and first in array to connectx()) to be stream 0 and thus the primary one.

#10 - 10/10/2019 07:00 PM - pespin

I have been working on libosmocore's API osmo_sock_init2_multiaddr() to create the socket with multiple local and remote addresses. I also added support for it in libosmo-netif's osmo_stream_server_link*() APIs.

Now I'm looking at libosmo-sccp.git. First problem while trying to implement the "local-ip" set in server VTY. For instance:

listen m3ua 2905
accept-asp-connections dynamic-permitted
local-ip 192.168.30.1
local-ip 192.168.30.1

- "listen" vty node calls "osmo_ss7_xua_server_create()" which creates xua server with host=NULL (0.0.0.0)
- Right now "local-ip" calls osmo_ss7_xua_server_bind()->osmo_stream_srv_link_open() every time, so it's impossible to set more than one local-ip. That bind needs to be removed and be done later. There's 2 possibilities:
  - let the main.c in osmo-stp do the right thing. It already does thanks to a commit I did a while ago in libosmo-sccp.git (10d4815bb1b4b548ec0bc97611b267ac45e0ebc5). But other applications may break maybe because they don't call that API introduced in
the commit?

- Apply bind() upon exit to the node. That seems to be the cleanest option imho, but requires the feature of "exit on node callback" be added to libosmocore.

Right now with the new osmo_sock_init2_multiaddr() which calls scpt_bindx binding to one specific IP (12.7.0.0.1 in this case) already seems to change the behavior, that is, osmo-stp doesn't bind+listen to all the IP addr:

```bash
$ cat /proc/net/sctp/assocs
ASSOC      SOCK  STY  SST  ST  HBKT  ASSOC-ID  TX_QUEUE  RX_QUEUE  UID  INODE  LPORT  RPORT  LADDRS  <--> RADDRS  HBINT  INS
           OUTS MAXRT  T1X  T2X  RTXC  wmem  wmemq  sndbuf  rcvbuf
0414cd21  71236246 2 1 3 0 22 0 0 1000 761112 2905 44660 127.0.0.1 <> *127.0.0.1
127.0.0.0 2 192.168.30.100 192.168.1.132 192.168.1.134 10.8.0.11 9000 10 10 10
0 0 0 1 0 212992 212992
2ded729f  12c601ec 2 1 3 0 21 0 0 1000 759276 44660 127.0.0.0 127.0.0.2 192.168.30.1 192.168.1.132 192.168.1.134 10.8.0.11 <> *127.0.0.1
9000 10 10 10
0 0 0 1 0 212992 212992
b2d3c479  3c936823 2 1 3 0 19 0 0 1000 755464 58089 127.0.0.1 127.0.0.2 192.168.30.1 192.168.1.132 192.168.1.134 10.8.0.11 <> *127.0.0.1
9000 10 10 10
0 0 0 1 0 212992 212992
8dbe9932  680b97d8 2 1 3 0 20 0 0 1000 752381 2905 58089 127.0.0.0 127.0.0.2 192.168.30.1 192.168.1.132 192.168.1.134 10.8.0.11 <> *127.0.0.1
9000 10 10 10
0 0 0 1 0 212992 212992
And here I passed a manually handcrafted ip addr from my system (192.168.30.1) into the code in libosmo-sccp (because VTY support is not yet there), so local binding seems to be working so far:

```bash
$ cat /proc/net/sctp/assocs
ASSOC      SOCK  STY  SST  ST  HBKT  ASSOC-ID  TX_QUEUE  RX_QUEUE  UID  INODE  LPORT  RPORT  LADDRS  <--> RADDRS  HBINT  INS
           OUTS MAXRT  T1X  T2X  RTXC  wmem  wmemq  sndbuf  rcvbuf
46a3bdc3  a349c970 2 1 3 0 47 0 0 1000 901729 41993 2905 127.0.0.1 127.0.0.2 192.168.30.1 192.168.30.100 192.168.1.132 192.168.1.134 10.8.0.11 <> *127.0.0.1
9000 10 10 10
0 0 0 1 0 212992 212992
f123ce70  4f7ce2c8 2 1 3 0 45 0 0 1000 903322 36889 2905 127.0.0.1 127.0.0.2 192.168.30.1 192.168.30.100 192.168.1.132 192.168.1.134 10.8.0.11 <> *127.0.0.1
9000 10 10 10
0 0 0 1 0 212992 212992
f8c62b1b  adbb8780 2 1 3 0 48 0 0 1000 897001 2905 36889 127.0.0.1 127.0.0.2 192.168.30.1 192.168.30.100 192.168.1.132 192.168.1.134 10.8.0.11 <> *127.0.0.1
9000 10 10 10
0 0 0 1 0 212992 212992
35c6ed4d  651b3248 2 1 3 0 46 0 0 1000 904313 36889 2905 127.0.0.1 127.0.0.2 192.168.30.1 192.168.30.100 192.168.1.132 192.168.1.134 10.8.0.11 <> *127.0.0.1
9000 10 10 10
0 0 0 1 0 212992 212992
223db1ca  b3b084e1 2 1 3 0 43 0 0 1000 904311 38603 2905 127.0.0.0 127.0.0.2 192.168.30.1 192.168.30.100 192.168.1.132 192.168.1.134 10.8.0.11 <> *127.0.0.1
9000 10 10 10
0 0 0 1 0 212992 212992
```

#11 - 10/11/2019 12:33 PM - pespin

I finally resolved the VTY issue by making sure the local-ip is bound not during cmd time but upon VTY node exits (so we can record several "local-ip" cmds, then apply all of them).

While doing so, I was hit by a libosmovty bug about go_parent_cb not called at the end of a file which I solved here:

https://gerrit.osmocom.org/c/libosmocore/+/15777

vty: Fix go_parent_cb not called for indented nodes at end of cfg file

#12 - 10/11/2019 07:38 PM - pespin

I submitted a first version implementing the desired features. I did some manual tests with osmo-bsc, osmo-msc and osmo-stp talking each other with different configurations and looks fine as far as I can tell so far.

With these changes, several "local-ip" and "remote-ip" lines are accepted under "listen" and "asp" VTY nodes, allowing to configure an SCTP connection with multiple connections, hence allowing control of SCTP multi-homing features. libosmo-sccp clients such as osmo-bsc and osmo-msc also gain support for this feature with these changes.

libosmocore (I still need to fix build for embedded targets):

https://gerrit.osmocom.org/c/libosmocore/+/15781 socket: Introduce API osmo_sock_init2_multiaddr()
remote: https://gerrit.osmocom.org/c/libosmo-netif/+/15783 stream: osmo_stream_cli: Support setting multiple addr

libosmo-sccp/osmo-sctp:
remote: https://gerrit.osmocom.org/c/libosmo-sccp/+/15784 Defer xua server binding until exit of VTY node
remote: https://gerrit.osmocom.org/c/libosmo-sccp/+/15785 ss7: Support multiple addresses in SCTP connections
remote: https://gerrit.osmocom.org/c/libosmo-sccp/+/15786 ss7: Log local and remote address set upon ASP restart

So current status is:

1. Extension of the VTY interface to configure multiple local and remote IP addresses <- DONE
2. Extension of the osmo-stp internal data structures to handle multiple IP addresses <- DONE
3. Corresponding configuration of the Linux kernel SCTP stack using its socket API from osmo-stp <- DONE (TBD: fix build under embedded env)
   1. local IP addresses are used for the sctp_bindx() socket API call <- DONE
   2. remote IP addresses are used for the sctp_connectx() socket API call <- DONE
4. Implementation of automatic test cases in TTCN-3 <- TBD

#13 - 11/05/2019 04:34 PM - laforge
- Checklist item [x] implementation in libosmo-sccp/osmo-stp set to Done
- % Done changed from 0 to 80

FYI: all patches in master for some time already. only part missing is automatic tests.

#14 - 11/07/2019 07:04 PM - pespin
- Status changed from In Progress to Feedback

FYI that's already being tested in a simple way in python unit tests of libosmo-sccp.git:
https://git.osmocom.org/libosmo-sccp/tree/tests/vty/vty_test_runner.py#n110

Not sure if we want to add more complex tests in TTCN3 for now.

#15 - 11/13/2019 04:51 PM - laforge
- Checklist item changed from TTCN-3 test cases to test cases
  Checklist item [x] test cases set to Done
- Status changed from Feedback to Closed
- % Done changed from 80 to 100

We decided that python based tests are more applicable for this feature. important factor is that we do have automatic tests, and they do pass. closing.