Building and Running OpenBSC with Asterisk

Pierre (Seungju) Kim
admin@manateeshome.com

Abstract

This document is intended to guide people who want to build a complete IP-Based GSM Network-In-a-Box using OpenBSC with Linux Call Router and Asterisk. Although OpenBSC supports partial Network-In-a-Box mode using osmo-nitb, it is limited to calls within the same network only. But with Asterisk, you have greater freedom on your configurations. In this tutorial I will set up OpenBSC so that it connects to Asterisk using chan_lcr of Linux Call Router. Basically chan_lcr module will serve as a bridge between OpenBSC and Asterisk. For this tutorial I used the newest git development versions as of February 9th, 2013. I used Debian 6 for host operating systen and OpenBSC is used with 1900MHz nanoBTS EDGE unit.

Requirements

- A computer with *NIX installed
- An IP-Based GSM base transceiver station(ip.access nanoBTS, sysmocom sysmoBTS)
- A GSM Phone and SIM card

1. Installing dependencies

```
apt-get install build-essential autoconf automake libtool libgsm1-dev
libdbi0-dev libdbd-sqlite3 git-core asterisk asterisk-dev ncurses-base
libncurses5-dev libncursesw5-dev libortp-dev sqlite3 pkg-config
```

2. Setting up the build environment and getting the development files

```
mkdir bsc
cd bsc
git clone git://git.misdn.org/mISDN.git/
git clone git://git.misdn.org/mISDNuser.git/
git clone git://git.osmocom.org/libosmocore.git/
git clone git://git.osmocom.org/libosmo-abis.git/
git clone git://git.osmocom.org/openbsc.git/
git clone git://git.misdn.org/lcr.git/
```

3. Building from source

1) mISDN

```
cd mISDN
./configure
cp mISDN.cfg.default standalone/mISDN.cfg
make modules
make modules_install
```

2) mISDNuser

```
cd mISDNuser
make             //mISDNuser generates the configuration with ‘make’
./configure
make
sudo make install
```

3) libosmocore

```
cd ../libosmocore
autoreconf -i
./configure
make
sudo make install
```

4) libosmo-abis

```
cd ../libosmo-abis
autoreconf -i
./configure
make
sudo make install
```

5) OpenBSC

```
cd ../openbsc/openbsc
autoreconf -i
./configure
make
```

6) LCR

```
cd ../../lcr
ln -s ../libosmocore/ .
ln -s ../openbsc/openbsc/ .
sh autogen.sh
./configure --prefix=/usr --with-asterisk --with-gsm-bs
make
sudo make install

cp chan_lcr.so /usr/lib/asterisk/modules/
```

On Debian 6 chan_lcr module will be loaded automatically on service restart, but in case it does not load automatically, I recommend adding the following line to your /etc/modules.conf.

```
load => chan_lcr.so
```

4. Configuring the system envrironment

```
ldconfig
depmod –a

modprobe mISDN_core
modprobe mISDN_dsp
modprobe mISDN_l1loop nchannel=30 interfaces=2
```

(Optional) Add following lines to your /etc/modules file if you want your system to load the modules automatically during boot

```
mISDN_core
mISDN_dsp
mISDN_l1loop nchannel=30 interfaces=2
```

5. Configuring LCR

```
1) Configuring GSM-BS

cd /usr/etc/lcr
nano interface.conf
```

Uncomment GSM section where it says 'gsm-bs'. It is an interface for GSM Base Station, in this case, osmo-nitb. Now LCR will connect to OpenBSC using its MNCC socket.

Add 'bridge ast' at the bottom of the GSM section. Now all calls from the GSM Base Station will be forwarded to Asterisk.

```
2) Set up chan_lcr

[ast]
remote asterisk
context from-lcr
earlyb no
tones yes
bridge GSM
```

LCR will connect to Asterisk using chan_lcr using context name 'from-lcr'. Calls from Asterisk will be forwarded to OpenBSC. The phone numbers defined in hlr.sqlite3 will be used to identify mobiles within the network. Please refer to the example Asterisk configuration on the next page.

```
3) Remove unnecessary interfaces
```

Comment the last two interfaces named Ext and Int. We do not need these interfaces. Now you have chan_lcr and GSM Base Station interface bridged to each other. All calls from one interface will be redirected to the other. No more routing configurations are necessary at this point.

```
4) Set up permissions

nano options.conf
```

Uncomment the following 2 lines. LCR will not connect to chan_lcr if you do not set the user and group it runs on.

```
socketuser asterisk
socketgroup asterisk
```

6. Configuring Asterisk

Following is a simple example of asterisk dialplan that can be used to route all calls from OpenBSC back to OpenBSC. But a few extensions listed in the default dialplan are accessible. Add the following lines to your /etc/asterisk/extensions.conf

```
[from-lcr]
include => default
exten => _X.,1,Dial(LCR/ast/${EXTEN:0},60)
```

This case, I am adding the default Asterisk dialplan. In Asterisk, default dialplan provides some features to test drive your network. You can call 1234 for information on Asterisk, and 600 for latency test. Please refer to the configuration file for more extensions. You might want to remove the default dialplan after you confirm that the network is working properly.

7. Setting up the base station

Now set up your base station so that it connects to OpenBSC. Use ip.access tools included in OpenBSC source directory. If you are not sure about what unit ID to use. 1800/0/0 is a safe choice. Use the IP of your computer as OML IP.

```
cd ipaccess
./ipaccess-config –u <Unit ID> –o <OML IP> <base station IP>
```

8. Configuring OpenBSC

Copy and paste the default OpenBSC configuration file from openbsc/doc/examples/osmo-nitb/nanobts/openbsc.cfg. Use one that matches your setup.

Edit the configuration file to suit your needs. I will not cover detailed configurations here at the moment. Please refer to the official OpenBSC wiki for more information on how to set up osmo-nitb.

You may want to check the following:

- Mobile Network Code/Mobile Country Code
- Operating band (GSM850, GSM900, DCS1800, PCS1900)
- ARFCN
- Base Station Unit ID

Warning: Running a GSM network with frequency band used in your area is highly discouraged. It may interfere with commercial networks and in most places such activity is considered illegal!

9. Running the network

```
service asterisk start
sudo lcr start
sudo ./osmo-nitb –m –P
```

Now you have a fully featured GSM network running from computer. Now take your phone and let it camp into your network. Please let me know if you have any questions or doubts. I am going to revise the document later on.

Version 1.0 - 2/13/2013