sysmocom

sysmocom - s.f.m.c. GmbH

OsmoBSC User Manual

by Holger Freyther, Harald Welte, and Neels Hofmeyr

The Asciidoc source code of this manual can be found at http://git.osmocom.org/osmo-gsm-manuals/

HISTORY			
NUMBER	DATE	DESCRIPTION	NAME
1	February 2016	Initial OsmoBSC manual, recycling OsmoNITB sections	HW
2	October 2018	Add Handover chapter: document new neighbor configuration, HO algorithm 2 and inter-BSC handover.	NH

Contents

Foreword

Digital cellular networks based on the GSM specification were designed in the late 1980ies and first deployed in the early 1990ies in Europe. Over the last 25 years, hundreds of networks were established globally and billions of subscribers have joined the associated networks.

The technological foundation of GSM was based on multi-vendor interoperable standards, first created by government bodies within CEPT, then handed over to ETSI, and now in the hands of 3GPP. Nevertheless, for the first 17 years of GSM technology, the associated protocol stacks and network elements have only existed in proprietary *black-box* implementations and not as Free Software.

In 2008 Dieter Spaar and I started to experiment with inexpensive end-of-life surplus Siemens GSM BTSs. We learned about the A-bis protocol specifications, reviewed protocol traces and started to implement the BSC-side of the A-bis protocol as something originally called `bs11-abis`. All of this was *just for fun*, in order to learn more and to boldly go where no Free Software developer has gone before. The goal was to learn and to bring Free Software into a domain that despite its ubiquity, had not yet seen any Free / Open Source software implementations.

`bs11-abis` quickly turned into `bsc-hack`, then *OpenBSC* and its *OsmoNITB* variant: A minimal implementation of all the required functionality of an entire GSM network, exposing A-bis towards the BTS. The project attracted more interested developers, and surprisingly quickly also commercial interest, contribution and adoption. This allowed adding support for more BTS models.

After having implemented the network-side GSM protocol stack in 2008 and 2009, in 2010 the same group of people set out to create a telephone-side implementation of the GSM protocol stack. This established the creation of the Osmocom umbrella project, under which OpenBSC and the OsmocomBB projects were hosted.

Meanwhile, more interesting telecom standards were discovered and implemented, including TETRA professional mobile radio, DECT cordless telephony, GMR satellite telephony, some SDR hardware, a SIM card protocol tracer and many others.

Increasing commercial interest particularly in the BSS and core network components has lead the way to 3G support in Osmocom, as well as the split of the minimal *OsmoNITB* implementation into separate and fully featured network components: OsmoBSC, OsmoMSC, OsmoHLR, OsmoMGW and OsmoSTP (among others), which allow seamless scaling from a simple "Network In The Box" to a distributed installation for serious load.

It has been a most exciting ride during the last eight-odd years. I would not have wanted to miss it under any circumstances.

— Harald Welte, Osmocom.org and OpenBSC founder, December 2017.

Acknowledgements

My deep thanks to everyone who has contributed to Osmocom. The list of contributors is too long to mention here, but I'd like to call out the following key individuals and organizations, in no particular order:

- Dieter Spaar for being the most amazing reverse engineer I've met in my career
- Holger Freyther for his many code contributions and for shouldering a lot of the maintenance work, setting up Jenkins - and being crazy enough to co-start sysmocom as a company with me ;)
- Andreas Eversberg for taking care of Layer2 and Layer3 of OsmocomBB, and for his work on OsmoBTS and OsmoPCU
- Sylvain Munaut for always tackling the hardest problems, particularly when it comes closer to the physical layer
- Chaos Computer Club for providing us a chance to run real-world deployments with tens of thousands of subscribers every year
- Bernd Schneider of Netzing AG for funding early ip.access nanoBTS support
- On-Waves ehf for being one of the early adopters of OpenBSC and funding a never ending list of features, fixes and general improvement of pretty much all of our GSM network element implementations
- sysmocom, for hosting and funding a lot of Osmocom development, the annual Osmocom Developer Conference and releasing this manual.

- Jan Luebbe, Stefan Schmidt, Daniel Willmann, Pablo Neira, Nico Golde, Kevin Redon, Ingo Albrecht, Alexander Huemer, Alexander Chemeris, Max Suraev, Tobias Engel, Jacob Erlbeck, Ivan Kluchnikov

May the source be with you!

— Harald Welte, Osmocom.org and OpenBSC founder, January 2016.

Endorsements

This version of the manual is endorsed by Harald Welte as the official version of the manual.

While the GFDL license (see Appendix C) permits anyone to create and distribute modified versions of this manual, such modified versions must remove the above endorsement.

Preface

First of all, we appreciate your interest in Osmocom software.

Osmocom is a Free and Open Source Software (FOSS) community that develops and maintains a variety of software (and partially also hardware) projects related to mobile communications.

Founded by people with decades of experience in community-driven FOSS projects like the Linux kernel, this community is built on a strong belief in FOSS methodology, open standards and vendor neutrality.

FOSS lives by contribution!

If you are new to FOSS, please try to understand that this development model is not primarily about "free of cost to the GSM network operator", but it is about a collaborative, open development model. It is about sharing ideas and code, but also about sharing the effort of software development and maintenance.

If your organization is benefitting from using Osmocom software, please consider ways how you can contribute back to that community. Such contributions can be many-fold, for example

- sharing your experience about using the software on the public mailing lists, helping to establish best practises in using/operating it,
- providing qualified bug reports, work-arounds
- sharing any modifications to the software you may have made, whether bug fixes or new features, even experimental ones
- providing review of patches
- testing new versions of the related software, either in its current "master" branch or even more experimental feature branches
- sharing your part of the maintenance and/or development work, either by donating developer resources or by (partially) funding those people in the community who do.

We're looking forward to receiving your contributions.

Osmocom and sysmocom

Some of the founders of the Osmocom project have established *sysmocom - systems for mobile communications GmbH* as a company to provide products and services related to Osmocom.

sysmocom and its staff have contributed by far the largest part of development and maintenance to the Osmocom mobile network infrastructure projects.

As part of this work, sysmocom has also created the manual you are reading.

At sysmocom, we draw a clear line between what is the Osmocom FOSS project, and what is sysmocom as a commercial entity. Under no circumstances does participation in the FOSS projects require any commercial relationship with sysmocom as a company.

Corrections

We have prepared this manual in the hope that it will guide you through the process of installing, configuring and debugging your deployment of cellular network infrastructure elements using Osmocom software. If you do find errors, typos and/or omissions, or have any suggestions on missing topics, please do take the extra time and let us know.

Legal disclaimers

Spectrum License

As GSM and UMTS operate in licensed spectrum, please always double-check that you have all required licenses and that you do not transmit on any ARFCN or UARFCN that is not explicitly allocated to you by the applicable regulatory authority in your country.

Warning
Depending on your jurisdiction, operating a radio transmitter without a proper license may be considered a felony under criminal law!

Software License

The software developed by the Osmocom project and described in this manual is Free / Open Source Software (FOSS) and subject to so-called *copyleft* licensing.

Copyleft licensing is a legal instrument to ensure that this software and any modifications, extensions or derivative versions will always be publicly available to anyone, for any purpose, under the same terms as the original program as developed by Osmocom.

This means that you are free to use the software for whatever purpose, make copies and distribute them - just as long as you ensure to always provide/release the *complete and corresponding* source code.

Every Osmocom software includes a file called `COPYING` in its source code repository which explains the details of the license. The majority of programs is released under GNU Affero General Public License, Version 3 (AGPLv3).

If you have any questions about licensing, don't hesitate to contact the Osmocom community. We're more than happy to clarify if your intended use case is compliant with the software licenses.

Trademarks

All trademarks, service marks, trade names, trade dress, product names and logos appearing in this manual are the property of their respective owners. All rights not expressly granted herein are reserved.

For your convenience we have listed below some of the registered trademarks referenced herein. This is not a definitive or complete list of the trademarks used.

Osmocom® and *OpenBSC*® are registered trademarks of Holger Freyther and Harald Welte.

sysmocom® and *sysmoBTS*® are registered trademarks of *sysmocom - systems for mobile communications GmbH*.

ip.access® and *nanoBTS*® are registered trademarks of *ip.access Ltd.*

Liability

The software is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the License text included with the software for more details.

Documentation License

Please see Appendix C for further information.

Introduction

Required Skills

Please note that even while the capital expenses of running mobile networks has decreased significantly due to Osmocom software and associated hardware like sysmoBTS, GSM networks are still primarily operated by large GSM operators.

Neither the GSM specification nor the GSM equipment was ever designed for networks to be installed and configured by anyone but professional GSM engineers, specialized in their respective area like radio planning, radio access network, back-haul or core network.

If you do not share an existing background in GSM network architecture and GSM protocols, correctly installing, configuring and optimizing your GSM network will be tough, irrespective whether you use products with Osmocom software or those of traditional telecom suppliers.

GSM knowledge has many different fields, from radio planning through site installation to core network configuration/administration.

The detailed skills required will depend on the type of installation and/or deployment that you are planning, as well as its associated network architecture. A small laboratory deployment for research at a university is something else than a rural network for a given village with a handful of cells, which is again entirely different from an urban network in a dense city.

Some of the useful skills we recommend are:

- general understanding about RF propagation and path loss in order to estimate coverage of your cells and do RF network planning.
- general understanding about GSM network architecture, its network elements and key transactions on the Layer 3 protocol
- general understanding about voice telephony, particularly those of ISDN heritage (Q.931 call control)
- understanding of GNU/Linux system administration and working on the shell
- understanding of TCP/IP networks and network administration, including tcpdump, tshark, wireshark protocol analyzers.
- ability to work with text based configuration files and command-line based interfaces such as the VTY of the Osmocom network elements

Getting assistance

If you do have a support package / contract with sysmocom (or want to get one), please contact support@sysmocom.de with any issues you may have.

If you don't have a support package / contract, you have the option of using the resources put together by the Osmocom community at http://projects.osmocom.org/, checking out the wiki and the mailing-list for community-based assistance. Please always remember, though: The community has no obligation to help you, and you should address your requests politely to them. The information (and software) provided at osmocom.org is put together by volunteers for free. Treat them like a friend whom you're asking for help, not like a supplier from whom you have bought a service.

Overview

This manual should help you getting started with OsmoBSC. It will cover aspects of configuring and running the OsmoBSC.

About OsmoBSC

OsmoBSC is the Osmocom implementation of a Base Station Controller. It implements:

- an *A-bis* interface towards BTSs and
- an *A* interface towards an MSC. It is important to be aware that there are two variants of the *A* interface, see Section 4.2.2.

Software Components

OsmoBSC contains a variety of different software components, which we'll briefly describe in this section.

A-bis Implementation

OsmoBSC implements the ETSI/3GPP specified A-bis interface, including TS 08.56 (LAPD), TS 08.58 (RSL) and TS 12.21 (OML). In addition, it supports a variety of vendor-specific extensions and dialects in order to communicate with BTSs from Siemens, Nokia, Ericsson, ip.access, Octasic and sysmocom, as well as various USRP based BTS implementations, using OsmoBTS and OsmoTRX (like the Ettus B200 series, the Fairwaves XTRX or the LimeSDR, to name a few).

For more information, see Section 9 and Section 10.

A Implementation

OsmoBSC implements a sub-set of the GSM A interface as specified in TS 08.08 (BSSAP) and TS 04.08 (DTAP).

Osmocom offers two variants of the *A* interface's protocol stacking:

- *BSSAP/SCCPlite*
- *BSSAP/SCCP/M3UA* (called AoIP)

Traditionally, OsmoBSC only implemented the BSSAP/SCCPlite protocol, but since a proper M3UA implementation became available in 2017 as part of *libosmo-sigtran* (*libosmo-sccp.git*), the stock OsmoBSC now supports BSSAP/SCCP/M3UA. SCCPlite support has been subsequently added to libosmo-sigtran in 2018, and now both variants of the *A* interface are supported by `osmo-bsc`.

The difference between an BSSAP/SCCPlite and BSSAP/SCCP/M3UA is illustrated in Figure 1 and Figure 2.

BSSAP/SCCPlite

Unlike classic A interface implementations for E1 interfacs, `osmo-bsc` implements a variant of encapsulating the A interface over IP. To do so, the SCCP messages are wrapped in an IPA multiplex and then communicated over TCP. The audio channels are mapped to RTP streams.

This protocol stacking is sometimes called "SCCPlite".

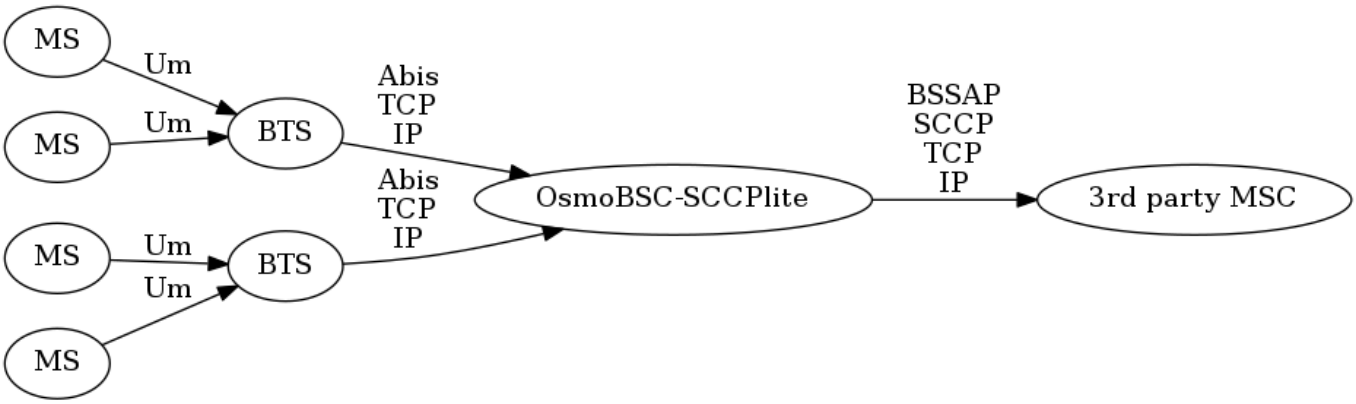


Figure 1: `osmo-bsc-sccplite` operation using *BSSAP/SCCPlite*

BSSAP/SCCP/M3UA

The default OsmoBSC's A interface uses the M3UA variant of SIGTRAN protocol stacking:

BSSAP
SCCP
M3UA
SCTP
IP

It is recommended to use the M3UA variant, which is required to operate with OsmoMSC.

To route SCCP/M3UA messages between OsmoBSC and and MSC, an STP instance like OsmoSTP is required.

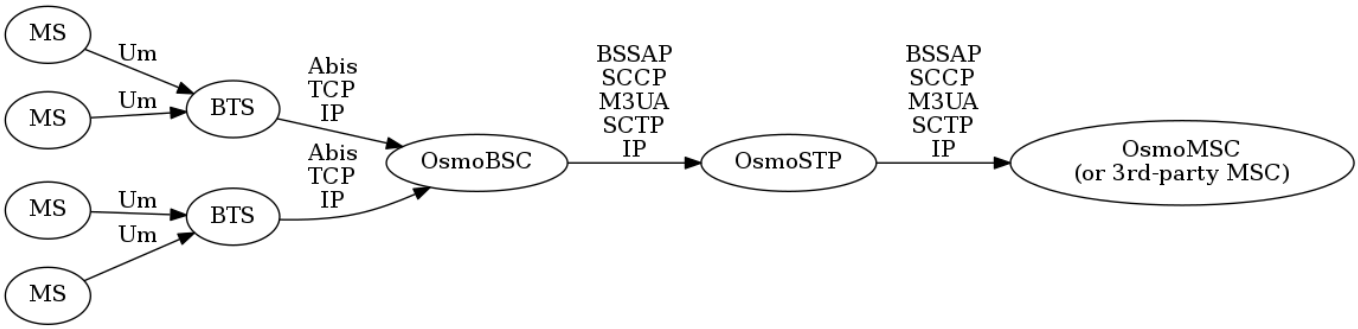


Figure 2: `osmo-bsc` operation using *BSSAP/SCCP/M3UA*

BSC Implementation

The BSC implementation covers the classic functionality of a GSM Base Station Controller, i.e.

- configuring and bringing up BTSs with their TRXs and TSs
- implementing the A-bis interface / protocols for signalling and actual voice data (TRAU frames).
- processing measurement results from the mobile stations in dedicated mode, performing hand-over decision and execution.
- Terminating the TS 04.08 RR (Radio Resource) sub-layer from the MS.

For more information, see [?], Section 9 and Section 10.

TRAU mapper / E1 sub-channel muxer

Unlike classic GSM networks, OsmoBSC does not perform any transcoding. Rather, a compatible codec is selected for both legs of a call, and codec frames are passed through transparently. In order to achieve this with E1 based BTS, OsmoBSC contains a E1 sub-channel de- and re-multiplexer as well as a TRAU mapper that can map uplink to downlink frames and vice versa.

Running OsmoBSC

The OsmoBSC executable (`osmo-bsc`) offers the following command-line arguments:

SYNOPSIS

osmo-bsc [-hl-V] [-d *DBGMASK*] [-D] [-c *CONFIGFILE*] [-s] [-T] [-e *LOGLEVEL*] [-l *IP*] [-r *RFCTL*]

OPTIONS

-h, --help
Print a short help message about the supported options

-V, --version
Print the compile-time version number of the program

-d, --debug *DBGMASK,DBGLEVELS*
Set the log subsystems and levels for logging to stderr. This has mostly been superseded by VTY-based logging configuration, see Section 7 for further information.

-D, --daemonize
Fork the process as a daemon into background.

-c, --config-file *CONFIGFILE*
Specify the file and path name of the configuration file to be used. If none is specified, use `osmo-bsc.cfg` in the current working directory.

-s, --disable-color
Disable colors for logging to stderr. This has mostly been deprecated by VTY based logging configuration, see Section 7 for more information.

-T, --timestamp
Enable time-stamping of log messages to stderr. This has mostly been deprecated by VTY based logging configu- ration, see Section 7 for more information.

-e, --log-level *LOGLEVEL*
Set the global log level for logging to stderr. This has mostly been deprecated by VTY based logging configuration, see Section 7 for more information.

-l, --local=*IP*
Specify the local IP address of the OsmoBSC-MGCP

-r, --rf-ctl *RFCTL*
Offer a Unix domain socket for RF control at the path/filename *RFCTL* in the file system.

Multiple instances

Running multiple instances of `osmo-bsc` on the same host is possible if all interfaces (VTY, CTRL) are separated using the appropriate configuration options. The IP based interfaces are binding to local host by default. In order to separate the processes, the user has to bind those services to specific but different IP addresses and/or ports.

The VTY and the Control interface can be bound to IP addresses from the loopback address range, for example:

```
line vty
 bind 127.0.0.2
ctrl
 bind 127.0.0.2
```

For the following links, OsmoBSC acts as a client and does not listen/bind to a specific interface, and will hence not encounter conflicts for multiple instances running on the same interface:

- The SCCP/M3UA links are established by OsmoBSC contacting an STP.
- The MGCP link is established by OsmoMSC contacting an MGW.

To run multiple OsmoBSC instances on the same A-interface (SCCP/M3UA), each BSC has to configure a distinct point-code. See Section 8.

Configure primary links

Connect to an MSC's *A* interface

Configure SCCP/M3UA (AoIP)

OsmoBSC acts as client to contact an STP instance and establish an SCCP/M3UA link.

An example configuration of OsmoBSC's AoIP SCCP link, assuming the BSC at point-code 1.23.3 and the MSC reachable at point-code 0.23.1 via an SG listening for M3UA at 127.0.0.1:2905:

```
cs7 instance 0
 point-code 1.23.3
 asp asp-clnt-msc-0 2905 0 m3ua
  remote-ip 127.0.0.1
  sctp-role client
 sccp-address msc
  point-code 0.23.1
msc 0
 msc-addr msc
```

This configuration is explained in detail in Section 8.

Configure SCCPlite

Traditionally, OsmoBSC implemented only an SCCPlite based A-interface, an ad-hoc standard encapsulating BSSAP in an IPA Multiplex. Since 2017, OsmoBSC supports primarily a proper 3GPP compliant SCCP/M3UA A-interface known as AoIP, by a new libosmo-sigtran implementation. In 2018, SCCPlite compatibility was added to libosmo-sigtran, re-enabling the option of using an SCCPlite based A-interface. For details, see the OsmoSTP manual, chapter "IPA / SCCPlite backwards compatibility".

Here is an example configuration of OsmoBSC for SCCPlite, assuming the BSC at point-code 1.23.3 and an SCCPlite MSC listening on 127.0.0.1:5000 with own point-code 0.23.1:


```
cs7 instance 0
 point-code 1.23.3
 asp asp-clnt-msc-0 5000 0 ipa
  remote-ip 127.0.0.1
 as as-clnt-msc-0 ipa
  asp asp-clnt-msc-0
  routing-key 0 1.23.3
  point-code override dpc 0.23.1
 sccp-address remote_msc
  point-code 0.23.1
msc 0
 msc-addr remote_msc
```

Configure MGCP to connect to an MGW

OsmoBSC uses a media gateway (typically OsmoMGW) to direct RTP streams. By default, an MGW is expected to receive MGCP requests on the IANA-registered default port for MGCP (2427) on local host (127.0.0.1).

Here is an example configuration for a remote MGW:

```
msc 0
 mgw remote-ip 10.9.8.7
 mgw remote-port 2427
```

Configure Lb to connect to an SMLC

Enable the Lb interface. OsmoBSC will then use the default point-codes to establish a connection to the SMLC.

```
smlc
 enable
```

More detailed configuration is described in Section 15.1.

The Osmocom VTY Interface

All human interaction with Osmocom software is typically performed via an interactive command-line interface called the *VTY*.

Note
Integration of your programs and scripts should **not** be done via the telnet VTY interface, which is intended for human interaction only: the VTY responses may arbitrarily change in ways obvious to humans, while your scripts' parsing will likely break often. For external software to interact with Osmocom programs (besides using the dedicated protocols), it is strongly recommended to use the Control interface instead of the VTY, and to actively request / implement the Control interface commands as required for your use case.

The interactive telnet VTY is used to

- explore the current status of the system, including its configuration parameters, but also to view run-time state and statistics,
- review the currently active (running) configuration,
- perform interactive changes to the configuration (for those items that do not require a program restart),
- store the current running configuration to the config file,

- enable or disable logging; to the VTY itself or to other targets.

The Virtual Tele Type (VTY) has the concept of *nodes* and *commands*. Each command has a name and arguments. The name may contain a space to group several similar commands into a specific group. The arguments can be a single word, a string, numbers, ranges or a list of options. The available commands depend on the current node. there are various keyboard shortcuts to ease finding commands and the possible argument values.

Configuration file parsing during program start is actually performed the VTY's CONFIG node, which is also available in the telnet VTY. Apart from that, the telnet VTY features various interactive commands to query and instruct a running Osmocom program. A main difference is that during config file parsing, consistent indenting of parent vs. child nodes is required, while the interactive VTY ignores indenting and relies on the *exit* command to return to a parent node.

Note
In the *CONFIG* node, it is not well documented which commands take immediate effect without requiring a program restart. To save your current config with changes you may have made, you may use the `write file` command to **overwrite** your config file with the current configuration, after which you should be able to restart the program with all changes taking effect.

This chapter explains most of the common nodes and commands. A more detailed list is available in various programs' VTY reference manuals, e.g. see [vty-ref-osmomsc].

There are common patterns for the parameters, these include IPv4 addresses, number ranges, a word, a line of text and choice. The following will explain the commonly used syntactical patterns:

Table 1: VTY Parameter Patterns

Pattern	Example	Explanation
`A.B.C.D`	`127.0.0.1`	An IPv4 address
`A.B.C.D/M`	`192.168.1.0/24`	An IPv4 address and mask
`X:X::X:X`	`::1`	An IPv6 address
`X:X::X:X/M`	`::1/128`	An IPv6 address and mask
`TEXT`	`example01`	A single string without any spaces, tabs
`.TEXT`	`Some information`	A line of text
`(OptionA|OptionB|OptionC)`	`OptionA`	A choice between a list of available options
`<0-10>`	`5`	A number from a range

Accessing the telnet VTY

The VTY of a given Osmocom program is implemented as a telnet server, listening to a specific TCP port.

Please see Appendix A to check for the default TCP port number of the VTY interface of the specific Osmocom software you would like to connect to.

As telnet is insecure and offers neither strong authentication nor encryption, the VTY by default only binds to localhost (127.0.0.1) and will thus not be reachable by other hosts on the network.

Warning
By default, any user with access to the machine running the Osmocom software will be able to connect to the VTY. We assume that such systems are single-user systems, and anyone with local access to the system also is authorized to access the VTY. If you require stronger security, you may consider using the packet filter of your operating system to restrict access to the Osmocom VTY ports further.

VTY Nodes

The VTY by default has the following minimal nodes:

VIEW
When connecting to a telnet VTY, you will be on the *VIEW* node. As its name implies, it can only be used to view the system status, but it does not provide commands to alter the system state or configuration. As long as you are in the non-privileged *VIEW* node, your prompt will end in a > character.

ENABLE
The *ENABLE* node is entered by the `enable` command, from the *VIEW* node. Changing into the *ENABLE* node will unlock all kinds of commands that allow you to alter the system state or perform any other change to it. The *ENABLE* node and its children are signified by a *#* character at the end of your prompt.
You can change back from the *ENABLE* node to the *VIEW* node by using the `disable` command.

CONFIG
The *CONFIG* node is entered by the `configure terminal` command from the *ENABLE* node. The config node is used to change the run-time configuration parameters of the system. The prompt will indicate that you are in the config node by a `(config)#` prompt suffix.
You can always leave the *CONFIG* node or any of its children by using the `end` command.
This node is also automatically entered at the time the configuration file is read. All configuration file lines are processed as if they were entered from the VTY *CONFIG* node at start-up.

Other
Depending on the specific Osmocom program you are running, there will be few or more other nodes, typically below the *CONFIG* node. For example, the OsmoBSC has nodes for each BTS, and within the BTS node one for each TRX, and within the TRX node one for each Timeslot.

Interactive help

The VTY features an interactive help system, designed to help you to efficiently navigate is commands.

Note
The VTY is present on most Osmocom GSM/UMTS/GPRS software, thus this chapter is present in all the relevant manuals. The detailed examples below assume you are executing them on the OsmoMSC VTY. They will work in similar fashion on the other VTY interfaces, while the node structure will differ in each program.

The question-mark (?) command

If you type a single `?` at the prompt, the VTY will display possible completions at the exact location of your currently entered command.

If you type `?` at an otherwise empty command (without having entered even only a partial command), you will get a list of the first word of all possible commands available at this node:

Example: Typing ? at start of OsmoMSC prompt

```
OsmoMSC> ❶
  show         Show running system information
  list         Print command list
  exit         Exit current mode and down to previous mode
  help         Description of the interactive help system
  enable       Turn on privileged mode command
  terminal     Set terminal line parameters
  who          Display who is on vty
  logging      Configure logging
  no           Negate a command or set its defaults
  sms          SMS related commands
  subscriber   Operations on a Subscriber
```


❶ Type `?` here at the prompt, the `?` itself will not be printed.

If you have already entered a partial command, `?` will help you to review possible options of how to continue the command. Let's say you remember that `show` is used to investigate the system status, but you don't remember the exact name of the object. Hitting `?` after typing `show` will help out:

Example: Typing ? after a partial command

```
OsmoMSC> show ❶
  version          Displays program version
  online-help      Online help
  history          Display the session command history
  cs7              ITU-T Signaling System 7
  logging          Show current logging configuration
  alarms           Show current logging configuration
  talloc-context   Show talloc memory hierarchy
  stats            Show statistical values
  asciidoc         Asciidoc generation
  rate-counters    Show all rate counters
  fsm              Show information about finite state machines
  fsm-instances    Show information about finite state machine instances
  sgs-connections  Show SGs interface connections / MMEs
  subscriber       Operations on a Subscriber
  bsc              BSC
  connection       Subscriber Connections
  transaction      Transactions
  statistics       Display network statistics
  sms-queue        Display SMSqueue statistics
  smpp          SMPP Interface
```

❶ Type `?` after the `show` command, the `?` itself will not be printed.

You may pick the `bsc` object and type `?` again:

Example: Typing ? after show bsc

```
OsmoMSC> show bsc
  <cr>
```

By presenting `<cr>` as the only option, the VTY tells you that your command is complete without any remaining arguments being available, and that you should hit enter, a.k.a. "carriage return".

TAB completion

The VTY supports tab (tabulator) completion. Simply type any partial command and press `<tab>`, and it will either show you a list of possible expansions, or completes the command if there's only one choice.

Example: Use of <tab> pressed after typing only s as command

```
OsmoMSC> s❶
show       sms        subscriber
```

❶ Type `<tab>` here.

At this point, you may choose `show`, and then press `<tab>` again:

Example: Use of <tab> pressed after typing show command


```
OsmoMSC> show ❶
version    online-help history    cs7        logging    alarms
talloc-context stats      asciidoc   rate-counters fsm        fsm-instances
sgs-connections subscriber bsc        connection transaction statistics
sms-queue smpp
```

❶ Type `<tab>` here.

The `list` command

The `list` command will give you a full list of all commands and their arguments available at the current node:

Example: Typing list at start of OsmoMSC VIEW node prompt

```
OsmoMSC> list
  show version
  show online-help
  list
  exit
  help
  enable
  terminal length <0-512>
  terminal no length
  who
  show history
  show cs7 instance <0-15> users
  show cs7 (sua|m3ua|ipa) [<0-65534>]
  show cs7 instance <0-15> asp
  show cs7 instance <0-15> as (active|all|m3ua|sua)
  show cs7 instance <0-15> sccp addressbook
  show cs7 instance <0-15> sccp users
  show cs7 instance <0-15> sccp ssn <0-65535>
  show cs7 instance <0-15> sccp connections
  show cs7 instance <0-15> sccp timers
  logging enable
  logging disable
  logging filter all (0|1)
  logging color (0|1)
  logging timestamp (0|1)
  logging print extended-timestamp (0|1)
  logging print category (0|1)
  logging print category-hex (0|1)
  logging print level (0|1)
  logging print file (0|1|basename) [last]
  logging set-log-mask MASK
  logging level (rll|cc|mm|rr|mncc|pag|msc|mgcp|ho|db|ref|ctrl|smpp|ranap|vlr|iucs|bssap| ↩
      sgs|lglobal|llapd|linp|lmux|lmi|lmib|lsms|lctrl|lgtp|lstats|lgsup|loap|lss7|lsccp|lsua ↩
      |lm3ua|lmgcp|ljibuf|lrspro) (debug|info|notice|error|fatal)
  logging level set-all (debug|info|notice|error|fatal)
  logging level force-all (debug|info|notice|error|fatal)
  no logging level force-all
  show logging vty
  show alarms
  show talloc-context (application|all) (full|brief|DEPTH)
  show talloc-context (application|all) (full|brief|DEPTH) tree ADDRESS
  show talloc-context (application|all) (full|brief|DEPTH) filter REGEXP
  show stats
  show stats level (global|peer|subscriber)
  show asciidoc counters
  show rate-counters
```



```
  show fsm NAME
  show fsm all
  show fsm-instances NAME
  show fsm-instances all
  show sgs-connections
  show subscriber (msisdn|extension|imsi|tmsi|id) ID
  show subscriber cache
  show bsc
  show connection
  show transaction
  sms send pending
  sms delete expired
  subscriber create imsi ID
  subscriber (msisdn|extension|imsi|tmsi|id) ID sms sender (msisdn|extension|imsi|tmsi|id)  ↩
      SENDER_ID send .LINE
  subscriber (msisdn|extension|imsi|tmsi|id) ID silent-sms sender (msisdn|extension|imsi| ↩
      tmsi|id) SENDER_ID send .LINE
  subscriber (msisdn|extension|imsi|tmsi|id) ID silent-call start (any|tch/f|tch/any|sdcch)
  subscriber (msisdn|extension|imsi|tmsi|id) ID silent-call stop
  subscriber (msisdn|extension|imsi|tmsi|id) ID ussd-notify (0|1|2) .TEXT
  subscriber (msisdn|extension|imsi|tmsi|id) ID ms-test close-loop (a|b|c|d|e|f|i)
  subscriber (msisdn|extension|imsi|tmsi|id) ID ms-test open-loop
  subscriber (msisdn|extension|imsi|tmsi|id) ID paging
  show statistics
  show sms-queue
  logging filter imsi IMSI
  show smpp esme
```

Tip

Remember, the list of available commands will change significantly depending on the Osmocom program you are accessing, its software version and the current node you're at. Compare the above example of the OsmoMSC *VIEW* node with the list of the OsmoMSC *NETWORK* config node:

Example: Typing list at start of OsmoMSC NETWORK config node prompt

```
OsmoMSC(config-net)# list
  help
  list
  write terminal
  write file
  write memory
  write
  show running-config
  exit
  end
  network country code <1-999>
  mobile network code <0-999>
  short name NAME
  long name NAME
  encryption a5 <0-3> [<0-3>] [<0-3>] [<0-3>]
  authentication (optional|required)
  rrlp mode (none|ms-based|ms-preferred|ass-preferred)
  mm info (0|1)
  timezone <-19-19> (0|15|30|45)
  timezone <-19-19> (0|15|30|45) <0-2>
  no timezone
  periodic location update <6-1530>
  no periodic location update
```


The attribute system

The VTY allows to edit the configuration at runtime. For many VTY commands the configuration change is immediately valid but for some commands a change becomes valid on a certain event only. In some cases it is even necessary to restart the whole process.

To give the user an overview, which configuration change applies when, the VTY implemets a system of attribute flags, which can be displayed using the `show` command with the parameter `vty-attributes`

Example: Typing show vty-attributes at the VTY prompt

```
OsmoBSC> show vty-attributes
  Global attributes:
    !  This command applies immediately
    @  This command applies on VTY node exit
  Library specific attributes:
    A  This command applies on ASP restart
    I  This command applies on IPA link establishment
    L  This command applies on E1 line update
  Application specific attributes:
    o  This command applies on A-bis OML link (re)establishment
    r  This command applies on A-bis RSL link (re)establishment
    l  This command applies for newly created lchans
```

The attributes are symbolized through a single ASCII letter (flag) and do exist in three levels. This is more or less due to the technical aspects of the VTY implementation. For the user, the level of an attribute has only informative purpose.

The global attributes, which can be found under the same attribute letter in every osmocom application, exist on the top level. The Library specific attributes below are used in various osmocom libraries. Like with the global attributes the attribute flag letter stays the same throughout every osmocom application here as well. On the third level one can find the application specific attributes. Those are unique to each osmocom application and the attribute letters may have different meanings in different osmocom applications. To make the user more aware of this, lowercase letters were used as attribute flags.

The `list` command with the parameter `with-flags` displays a list of available commands on the current VTY node, along with attribute columns on the left side. Those columns contain the attribute flag letters to indicate to the user how the command behaves in terms of how and when the configuration change takes effect.

Example: Typing list with-flags at the VTY prompt

```
OsmoBSC(config-net-bts)# list with-flags
  . ...  help
  . ...  list [with-flags]
  . ...  show vty-attributes
  . ...  show vty-attributes (application|library|global)
  . ...  write terminal
  . ...  write file [PATH]
  . ...  write memory
  . ...  write
  . ...  show running-config
  . ...  exit
  . ...  end
  . o..  type (unknown|bs11|nanobts|rbs2000|nokia_site|sysmobts)
  . ...  description .TEXT
  . ...  no description
  . o..  band BAND
  . .r.  cell_identity <0-65535>
  . .r.  dtx uplink [force]
  . .r.  dtx downlink
  . .r.  no dtx uplink
  . .r.  no dtx downlink
  . .r.  location_area_code <0-65535>
  . o..  base_station_id_code <0-63>
  . o..  ipa unit-id <0-65534> <0-255>
```



```
. o..  ipa rsl-ip A.B.C.D
. o..  nokia_site skip-reset (0|1)
! ...  nokia_site no-local-rel-conf (0|1)
! ...  nokia_site bts-reset-timer  <15-100>
```

There are multiple columns because a single command may be associated with multiple attributes at the same time. To improve readability each flag letter gets a dedicated column. Empty spaces in the column are marked with a dot (".")

In some cases the listing will contain commands that are associated with no flags at all. Those commands either play an exceptional role (interactive commands outside "configure terminal", vty node navigation commands, commands to show / write the config file) or will require a full restart of the overall process to take effect.

libosmocore Logging System

In any reasonably complex software it is important to understand how to enable and configure logging in order to get a better insight into what is happening, and to be able to follow the course of action. We therefore ask the reader to bear with us while we explain how the logging subsystem works and how it is configured.

Most Osmocom Software (like `osmo-bts`, `osmo-bsc`, `osmo-nitb`, `osmo-sgsn` and many others) uses the same common logging system.

This chapter describes the architecture and configuration of this common logging system.

The logging system is composed of

- log targets (where to log),
- log categories (who is creating the log line),
- log levels (controlling the verbosity of logging), and
- log filters (filtering or suppressing certain messages).

All logging is done in human-readable ASCII-text. The logging system is configured by means of VTY commands that can either be entered interactively, or read from a configuration file at process start time.

Log categories

Each sub-system of the program in question typically logs its messages as a different category, allowing fine-grained control over which log messages you will or will not see. For example, in OsmoBSC, there are categories for the protocol layers `rsl`, `rr`, `mm`, `cc` and many others. To get a a list of categories interactively on the vty, type: `logging level ?`

Log levels

For each of the log categories (see Section 7.1), you can set an independent log level, controlling the level of verbosity. Log levels include:

fatal
: Fatal messages, causing abort and/or re-start of a process. This *shouldn't happen*.

error
: An actual error has occurred, its cause should be further investigated by the administrator.

notice
: A noticeable event has occurred, which is not considered to be an error.

info
: Some information about normal/regular system activity is provided.

debug
Verbose information about internal processing of the system, used for debugging purpose. This will log the most.

The log levels are inclusive, e.g. if you select *info*, then this really means that all events with a level of at least *info* will be logged, i.e. including events of *notice*, *error* and *fatal*.

So for example, in OsmoBSC, to set the log level of the Mobility Management category to info, you can use the following command: `log level mm info`.

There is also a special command to set all categories as a one-off to a desired log level. For example, to silence all messages but those logged as notice and above issue the command: `log level set-all notice`

Afterwards you can adjust specific categories as usual.

A similar command is `log level force-all <level>` which causes all categories to behave as if set to log level <level> until the command is reverted with `no log level force-all` after which the individually-configured log levels will again take effect. The difference between `set-all` and `force-all` is that `set-all` actually changes the individual category settings while `force-all` is a (temporary) override of those settings and does not change them.

Log printing options

The logging system has various options to change the information displayed in the log message.

log color 1
With this option each log message will log with the color of its category. The color is hard-coded and can not be changed. As with other options a *0* disables this functionality.

log timestamp 1
Includes the current time in the log message. When logging to syslog this option should not be needed, but may come in handy when debugging an issue while logging to file.

log print extended-timestamp 1
In order to debug time-critical issues this option will print a timestamp with millisecond granularity.

log print category 1
Prefix each log message with the category name.

log print category-hex 1
Prefix each log message with the category number in hex (<*000b*>).

log print level 1
Prefix each log message with the name of the log level.

log print file 1
Prefix each log message with the source file and line number. Append the keyword `last` to append the file information instead of prefixing it.

Log filters

The default behavior is to filter out everything, i.e. not to log anything. The reason is quite simple: On a busy production setup, logging all events for a given subsystem may very quickly be flooding your console before you have a chance to set a more restrictive filter.

To request no filtering, i.e. see all messages, you may use: `log filter all 1`

In addition to generic filtering, applications can implement special log filters using the same framework to filter on particular context.

For example in OsmoBSC, to only see messages relating to a particular subscriber identified by his IMSI, you may use: `log filter imsi 262020123456789`

Log targets

Each of the log targets represent certain destination for log messages. It can be configured independently by selecting levels (see Section 7.2) for categories (see Section 7.1) as well as filtering (see Section 7.4) and other options like `logging timestamp` for example.

Logging to the VTY

Logging messages to the interactive command-line interface (VTY) is most useful for occasional investigation by the system administrator.

Logging to the VTY is disabled by default, and needs to be enabled explicitly for each such session. This means that multiple concurrent VTY sessions each have their own logging configuration. Once you close a VTY session, the log target will be destroyed and your log settings be lost. If you re-connect to the VTY, you have to again activate and configure logging, if you wish.

To create a logging target bound to a VTY, you have to use the following command: `logging enable` This doesn't really activate the generation of any output messages yet, it merely creates and attaches a log target to the VTY session. The newly-created target still doesn't have any filter installed, i.e. *all log messages will be suppressed by default*

Next, you can configure the log levels for desired categories in your VTY session. See Section 7.1 for more details on categories and Section 7.2 for the log level details.

For example, to set the log level of the Call Control category to debug, you can use: `log level cc debug`

Finally, after having configured the levels, you still need to set the filter as it's described in Section 7.4.

Tip
If many messages are being logged to a VTY session, it may be hard to impossible to still use the same session for any commands. We therefore recommend to open a second VTY session in parallel, and use one only for logging, while the other is used for interacting with the system. Another option would be to use different log target.

To review the current vty logging configuration, you can use: `show logging vty`

Logging to the ring buffer

To avoid having separate VTY session just for logging output while still having immediate access to them, one can use `alarms` target. It lets you store the log messages inside the ring buffer of a given size which is available with `show alarms` command.

It's configured as follows:

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# log alarms 98
OsmoBSC(config-log)#
```

In the example above 98 is the desired size of the ring buffer (number of messages). Once it's filled, the incoming log messages will push out the oldest messages available in the buffer.

Logging via gsmtap

When debugging complex issues it's handy to be able to reconstruct exact chain of events. This is enabled by using GSMTAP log output where frames sent/received over the air are intersperced with the log lines. It also simplifies the bug handling as users don't have to provide separate .pcap and .log files anymore - everything will be inside self-contained packet dump.

It's configured as follows:


```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# log gsmtap 192.168.2.3
OsmoBSC(config-log)#
```

The hostname/ip argument is optional: if omitted the default 127.0.0.1 will be used. The log strings inside GSMTAP are already supported by Wireshark. Capturing for `port 4729` on appropriate interface will reveal log messages including source file name and line number as well as application. This makes it easy to consolidate logs from several different network components alongside the air frames. You can also use Wireshark to quickly filter logs for a given subsystem, severity, file name etc.

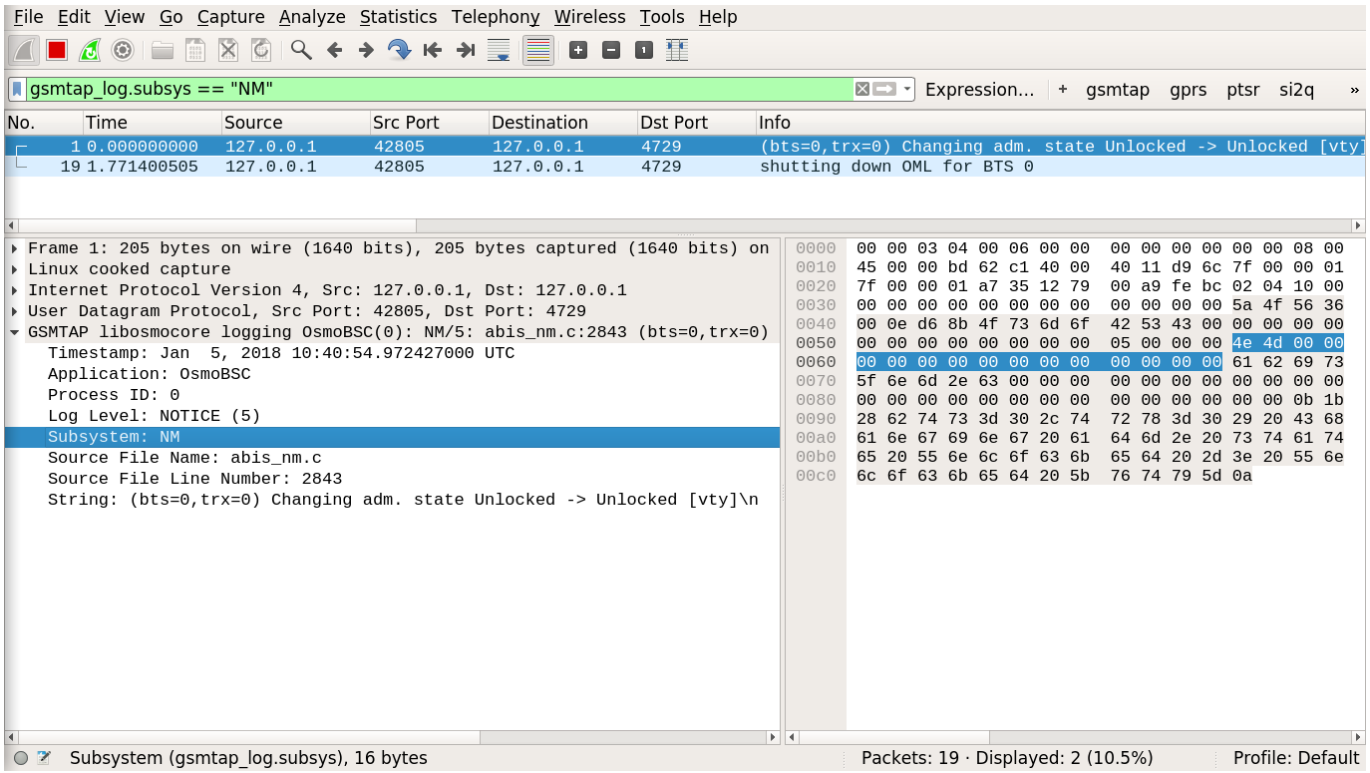


Figure 3: Wireshark with logs delivered over GSMTAP

Note: the logs are also duplicated to stderr when GSMTAP logging is configured because stderr is the default log target which is initialized automatically. To descrease stderr logging to absolute minimum, you can configure it as follows:

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# log stderr
OsmoBSC(config-log)# logging level force-all fatal
```

Logging to a file

As opposed to Logging to the VTY, logging to files is persistent and stored in the configuration file. As such, it is configured in sub-nodes below the configuration node. There can be any number of log files active, each of them having different settings regarding levels / subsystems.

To configure a new log file, enter the following sequence of commands:

```
OsmoBSC> enable
OsmoBSC# configure terminal
```



```
OsmoBSC(config)# log file /path/to/my/file
OsmoBSC(config-log)#
```

This leaves you at the config-log prompt, from where you can set the detailed configuration for this log file. The available commands at this point are identical to configuring logging on the VTY, they include `logging filter`, `logging level` as well as `logging color` and `logging timestamp`.

Tip
Don't forget to use the `copy running-config startup-config` (or its short-hand `write file`) command to make your logging configuration persistent across application re-start.

Note
libosmocore provides file close-and-reopen support by SIGHUP, as used by popular log file rotating solutions such as https://github.com/logrotate/logrotate found in most GNU/Linux distributions.

Logging to syslog

syslog is a standard for computer data logging maintained by the IETF. Unix-like operating systems like GNU/Linux provide several syslog compatible log daemons that receive log messages generated by application programs.

libosmocore based applications can log messages to syslog by using the syslog log target. You can configure syslog logging by issuing the following commands on the VTY:

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# log syslog daemon
OsmoBSC(config-log)#
```

This leaves you at the config-log prompt, from where you can set the detailed configuration for this log file. The available commands at this point are identical to configuring logging on the VTY, they include `logging filter`, `logging level` as well as `logging color` and `logging timestamp`.

Note
Syslog daemons will normally automatically prefix every message with a time-stamp, so you should disable the libosmocore time-stamping by issuing the `logging timestamp 0` command.

Logging to stderr

If you're not running the respective application as a daemon in the background, you can also use the stderr log target in order to log to the standard error file descriptor of the process.

In order to configure logging to stderr, you can use the following commands:

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# log stderr
OsmoBSC(config-log)#
```


Configure SCCP/M3UA

All CNI programs using SCCP/M3UA act as M3UA ASP role and SCTP client, expecting to connect to a Signalling Gateway (STP/SG) implementing the M3UA SG role as SCTP server. The STP/SG then routes M3UA messages between its ASPs, typically by point-codes.

In an all-Osmocom CNI, the typical simple/minimal usage is:

- OsmoSTP acts as the STP/SG (server role) and routes between the ASP,
- All other Osmocom CNI programs act as SCTP client and provide ASP implementations.

For example, in an all-Osmocom minimal setup,

- OsmoMSC contacts an OsmoSTP and subscribes its point-code 0.23.1;
- then OsmoBSC also contacts the same OsmoSTP, subscribes with its own point-code 1.23.3.
- Using these established links, OsmoBSC initiates an A-interface link by directing a BSSAP RESET message to the MSC's point-code 0.23.1,
- and the RESET ACK response from the MSC is routed back to the BSC's point-code 1.23.3.

The details of SCCP/M3UA are configured in the *cs7* section of the VTY configuration.

Osmocom programs automatically configure missing SCCP/M3UA configuration, by assuming sane defaults for small/minimal all-Osmocom installations, which may not be what you want in larger networks integrating with non-Osmocom core network elements.

If no explicit `routing-key` is set, it may be determined at runtime by negotiation with OsmoSTP — see OsmoSTP manual chapter "Osmocom M3UA Routing Key Management Extensions", regarding config option `accept-asp-connections dynamic-permitted`.

The complete active configuration of an Osmocom program can be obtained by the VTY command `show cs7 config` (the usual `show running-config` omits automatically configured items). Here is an example of OsmoMSC's default configuration:

```
OsmoMSC> show cs7 config
cs7 instance 0
 point-code 0.23.1
 asp asp-clnt-OsmoMSC-A-Iu 2905 0 m3ua
  remote-ip 127.0.0.1
  sctp-role client
 as as-clnt-OsmoMSC-A-Iu m3ua
  asp asp-clnt-OsmoMSC-A-Iu
  routing-key 2 0.23.1
```

At the time of writing, SCCP/M3UA links involving Osmocom program are:

- A-interface: OsmoBSC to OsmoMSC
- IuCS-interface: OsmoHNBGW to OsmoMSC
- IuPS-interface: OsmoHNBGW to OsmoSGSN

Connect to STP Instance

By default, an STP instance is assumed to listen on the default M3UA port (2905) on the local host (127.0.0.1).

Establishing an SCCP/M3UA link towards a remote STP instance can be configured as:

```
cs7 instance 0
 asp my-asp 2905 0 m3ua
  # IP address of the remote STP:
  remote-ip 10.23.24.1
  # optional: local bind to a specific IP
  local-ip 10.9.8.7
```

Be aware that such an `asp` needs to be linked to an `as`, see Section 8.5.

Local Point-Code

Each CNI program on an SCCP/M3UA link typically has a local point-code, configurable by:

```
cs7 instance 0
 point-code 7.65.4
```

If an explicit routing context is configured, this point-code is repeated in the `routing-key` configuration:

```
cs7 instance 0
 point-code 0.23.1
 as my-as m3ua
  routing-key 2 0.23.1
```

See also Section 8.4.

Remote Point-Code

Programs establishing communication across SCCP links need a remote SCCP address, typically by point-code, to contact.

For example,

- OsmoBSC needs to know the MSC's point-code, to be able to establish the A-interface.
- OsmoHNBGW needs to know the MSC's point-code, to be able to establish the IuCS-interface.
- OsmoHNBGW needs to know the SGSN's point-code, to be able to establish the IuPS-interface.

To maintain remote SCCP addresses, each `cs7` instance maintains an SCCP address book:

```
cs7 instance 0
 sccp-address remote-pc-example
  point-code 1.23.1
```

This address book entry on its own has no effect. It is typically referenced by specific configuration items depending on the individual programs.

Examples:

- An OsmoBSC configures the MSC's remote SCCP address:

```
cs7 instance 0
 sccp-address my-remote-msc
  point-code 1.23.1
msc 0
 msc-addr my-remote-msc
```


- An HNBGW configures both the remote MSC's and SGSN's SCCP addresses:

```
cs7 instance 0
 sccp-address my-msc
  point-code 0.23.1
 sccp-address my-sgsn
  point-code 0.23.2
hnbgw
 iucs
  remote-addr my-msc
 iups
  remote-addr my-sgsn
```

Besides a point-code, an SCCP address can have several routing indicators:

- PC: routing by point-code is the default for Osmocom.
- GT: routing by Global Title is configurable by `routing-indicator GT`.
- IP: routing by IP address is configurable by `routing-indicator IP`.

In OsmoSTP, only routing by point-code is currently implemented.

Point-Code Format

Point-codes can be represented in various formats. For details, see OsmoSTP manual, chapter "Point Codes".

By default, Osmocom uses a point-code representation of 3.8.3, i.e. first digit of 3 bit, second digit of 8 bit, and third digit of 3 bit.

```
cs7 instance 0
 point-code format 3 8 3
 point-code 0.23.1
```

Often, point-codes are also represented as a single decimal number:

```
cs7 instance 0
 point-code format 24
 point-code 185
```

It is also possible to use a dash as delimiter.

```
cs7 instance 0
 point-code delimiter dash
 point-code 0-23-1
```

AS and ASP

Each CNI program needs at least one Application Server `as` and one Application Server Process `asp` configured on its `cs7` to be able to communicate on SCCP/M3UA. An `asp` needs to be part of at least one `as`. For details, see the OsmoSTP manual, chapters "Application Server" and "Application Server Process".

In Osmocom's `cs7`, any amount of `as` and `asp` can be configured by name, and an `as` references the `asp` entries belonging to it by their names.

In a simple/minimal Osmocom setup, an Osmocom CNI program would have exactly one `as` with one `asp`.

For example:


```
cs7 instance 0
 asp my-asp 2905 0 m3ua
  # where to reach the STP:
  remote-ip 127.0.0.1
  sctp-role client
 as my-as m3ua
  asp my-asp
```

In Osmocom CNI programs, it is possible to omit the `as` and/or `asp` entries, which the program will then attempt to configure automatically.

When configuring both `as` and `asp` manually, make sure to link them by name. For example, the following configuration will **fail**, because `as` and `asp` are not linked:

```
cs7 instance 0
 asp my-asp 2905 0 m3ua
  remote-ip 127.0.0.1
 as my-as m3ua
  routing-key 2 0.23.1
```

To **fix** above config, link the `asp` to an `as` by adding `asp my-asp`:

```
cs7 instance 0
 asp my-asp 2905 0 m3ua
  remote-ip 127.0.0.1
 as my-as m3ua
  asp my-asp
  routing-key 2 0.23.1
```

Subsystem Number (SSN)

Osmocom CNI programs typically route SCCP/M3UA messages by PC+SSN: each ASP, having a given SCCP address, receives messages for one or more specific subsystems, identified by a Subsystem Number (SSN).

For example, the A-interface between BSC and MSC uses SSN = BSSAP (254). In Osmocom programs, SSNs do not need to be configured; they implicitly, naturally relate to the interfaces that a program implements.

For example, OsmoBSC takes the configured remote MSC's SCCP address and adds the SSN = BSSAP to it in order to contact the MSC's A-interface. To receive A-interface messages from the MSC, OsmoBSC subscribes a local user for this SSN on the ASP.

Routing Context / Routing Key

In SCCP/M3UA, messages can be routed by various Routing Indicators (PC+SSN, PC, GT, ...). Osmocom CNI programs typically use PC+SSN as Routing Indicator.

On the SG (for example OsmoSTP), each ASP's distinct Routing Indicator needs to be indexed by a distinct Routing Context (a simple index number scoped per SG), to forward M3UA to the correct peer.

The Osmocom SG implementation employs Routing Key Management (RKM, see OsmoSTP manual) to automatically determine a distinct Routing Context index for each connected ASP. Routing Contexts can also be configured manually — some non-Osmocom SG implementations require this.

Each Routing Context is associated with a Routing Indicator and address; this association is called a Routing Key.

For example, to configure an OsmoBSC with a local point-code of 1.23.3 to receive M3UA with Routing Context of 2 and RI=PC:


```
cs7 instance 0
 point-code 1.23.3
 as my-as m3ua
  routing-key 2 1.23.3
```

Osmocom programs so far implement Routing Keys by Destination Point Code (DPC), plus optional Subsystem Number (SSN) and/or Service Indicator (SI):

```
routing-key RCONTEXT DPC
routing-key RCONTEXT DPC si (aal2|bicc|b-isup|h248|isup|sat-isup|sccp|tup)
routing-key RCONTEXT DPC ssn SSN
routing-key RCONTEXT DPC si (aal2|bicc|b-isup|h248|isup|sat-isup|sccp|tup) ssn SSN
```

Reviewing and Provisioning BTS configuration

The main functionality of the BSC component is to manage BTSs. As such, provisioning BTSs within the BSC is one of the most common tasks during BSC operation. Just like about anything else in OsmoBSC, they are configured using the VTY.

BTSs are internally numbered with integer numbers starting from "0" for the first BTS. BTS numbers have to be contiguous, so you cannot configure 0,1,2 and then 5.

Reviewing current BTS status and configuration

In order to view the status and properties of a BTS, you can issue the `show bts` command. If used without any BTS number, it will display information about all provisioned BTS numbers.

```
OsmoBSC> show bts 0
BTS 0 is of nanobts type in band DCS1800, has CI 0 LAC 1, BSIC 63, TSC 7 and 1 TRX
Description: (null)
MS Max power: 15 dBm
Minimum Rx Level for Access: -110 dBm
Cell Reselection Hysteresis: 4 dBm
RACH TX-Integer: 9
RACH Max transmissions: 7
System Information present: 0x0000007e, static: 0x00000000
  Unit ID: 200/0/0, OML Stream ID 0xff
  NM State: Oper 'Enabled', Admin 2, Avail 'OK'
  Site Mgr NM State: Oper 'Enabled', Admin 0, Avail 'OK'
  Paging: 0 pending requests, 0 free slots
  OML Link state: connected.
  Current Channel Load:
                   TCH/F:   0% (0/5)
                  SDCCH8:   0% (0/8)
```

You can also review the status of the TRXs configured within the BTSs of this BSC by using `show trx`:

```
OsmoBSC> show trx 0 0
TRX 0 of BTS 0 is on ARFCN 871
Description: (null)
  RF Nominal Power: 23 dBm, reduced by 0 dB, resulting BS power: 23 dBm
  NM State: Oper 'Enabled', Admin 2, Avail 'OK'
  Baseband Transceiver NM State: Oper 'Enabled', Admin 2, Avail 'OK'
  ip.access stream ID: 0x00
```

The output can be restricted to the TRXs of one specified BTS number (`show trx 0`) or even that of a single specified TRX within a specified BTS (`show trx 0 0`).

Furthermore, information on the individual timeslots can be shown by means of `show timeslot`. The output can be restricted to the timeslots of a single BTS (`show timeslot 0`) or that of a single TRX (`show timeslot 0 0`). Finally, you can restrict the output to a single timeslot by specifying the BTS, TRX and TS numbers (`show timeslot 0 0 4`).

```
OsmoBSC> show timeslot 0 0 0
BTS 0, TRX 0, Timeslot 0, phys cfg CCCH, TSC 7
  NM State: Oper 'Enabled', Admin 2, Avail 'OK'
OsmoBSC> show timeslot 0 0 1
BTS 0, TRX 0, Timeslot 1, phys cfg SDCCH8, TSC 7
  NM State: Oper 'Enabled', Admin 2, Avail 'OK'
```

Provisioning a new BTS

In order to provision BTSs, you have to enter the BTS config node of the VTY. In order to configure BTS 0, you can issue the following sequence of commands:

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# network
OsmoBSC(config-net)# bts 0
OsmoBSC(config-net-bts)#
```

At this point, you have a plethora of commands, in fact an entire hierarchy of commands to configure all aspects of the BTS, as well as each of its TRX and each timeslot within each TRX. For a full reference, please consult the telnet VTY integrated help or the respective chapter in the VTY reference.

BTS configuration depends quite a bit on the specific BTS vendor and model. The section below provides just one possible example for the case of a sysmoBTS.

Note that from the `configure terminal` command onwards, the telnet VTY commands above are identical to configuration file settings, for details see Section 6.

Starting with `network` as above, your complete sysmoBTS configuration may look like this:

```
network
 bts 0
  type sysmobts
  band DCS1800
  description The new BTS in Baikonur
  location_area_code 2342
  cell_identity 5
  base_station_id_code 63
  ip.access unit_id 8888 0
  ms max power 40
  trx 0
   arfcn 871
   nominal power 23
   max_power_red 0
   timeslot 0
    phys_chan_config CCCH+SDCCH4
   timeslot 1
    phys_chan_config TCH/F
   timeslot 2
    phys_chan_config TCH/F
   timeslot 3
    phys_chan_config TCH/F
   timeslot 4
    phys_chan_config TCH/F
   timeslot 5
    phys_chan_config TCH/F
   timeslot 6
```



```
 phys_chan_config TCH/F
timeslot 7
 phys_chan_config PDCH
```

System Information configuration

A GSM BTS periodically transmits a series of *SYSTEM INFORMATION* messages to mobile stations, both via the BCCH in idle mode, was well as via the SACCH in dedicated mode. There are many different types of such messages. For their detailed contents and encoding, please see *3GPP TS 24.008* [3gpp-ts-24-008].

For each of the *SYSTEM INFORMATION* message types, you can configure to have the BSC generate it automatically (*computed*), or you can specify the respective binary message as a string of hexadecimal digits.

The default configuration is to compute all (required) *SYSTEM INFORMATION* messages automatically.

Please see the *OsmoBSC VTY Reference Manual* [vty-ref-osmobsc] for further information, particularly on the following commands:

- `system-information (1|2|3|4|5|6|7|8|9|10|13|16|17|18|19|20|2bis|2ter|2quater|5bis|5ter) mode (static|computed)`
- `system-information (1|2|3|4|5|6|7|8|9|10|13|16|17|18|19|20|2bis|2ter|2quater|5bis|5ter) static HEXSTRING`

Neighbor List configuration

Every BTS sends a list of ARFCNs of neighbor cells . within its *SYSTEM INFORMATION 2* (and 2bis/2ter) messages on the BCCH . within its *SYSTEM INFORMATION 5* messages on SACCH in dedicated mode

For every BTS config node in the VTY, you can specify the behavior of the neighbor list using the `neighbor list mode` VTY command:

automatic
: Automatically generate a list of neighbor cells using all other BTSs configured in the VTY

manual
: Manually specify the neighbor list by means of `neighbor-list (add|del) arfcn <0-1023>` commands, having identical neighbor lists on BCCH (SI2) and SACCH (SI5)

manual-si5
: Manually specify the neighbor list by means of `neighbor-list (add|del) arfcn <0-1023>` for BCCH (SI2) and a separate neighbor list by means of `si5 neighbor-list (add|del) arfcn <0-1023>` for SACCH (SI5).

Configuring GPRS PCU parameters of a BTS

In the case of BTS models using Abis/IP (IPA), the GPRS PCU is located inside the BTS. The BTS then establishes a Gb connection to the SGSN.

All the BTS-internal PCU configuration is performed via A-bis OML by means of configuring the *CELL*, *NSVC* (NS Virtual Connection and *NSE* (NS Entity).

There is one *CELL* node and one *NSE* node, but there are two *NSVC* nodes. At the time of this writing, only the NSVC 0 is supported by OsmoBTS, while both NSVC are supported by the ip.access nanoBTS.

The respective VTY configuration parameters are described below. They all exist beneath each BTS VTY config node.

But let's first start with a small example

Example configuration of GPRS PCU parameters at VTY BTS node


```
OsmoBSC(config-net-bts)# gprs mode gprs
OsmoBSC(config-net-bts)# gprs routing area 1
OsmoBSC(config-net-bts)# gprs cell bvci 1234
OsmoBSC(config-net-bts)# gprs nsei 1234
OsmoBSC(config-net-bts)# gprs nsvc 0 nsvci 1234
OsmoBSC(config-net-bts)# gprs nsvc 0 local udp port 23000
OsmoBSC(config-net-bts)# gprs nsvc 0 remote udp port 23000
OsmoBSC(config-net-bts)# gprs nsvc 0 remote ip 192.168.100.239
```

More explanation about the PCU config parameters

gprs mode (none|gprs|egprs)

This command determines if GPRS (or EGPRS) services are to be enabled in this cell at all.

gprs cell bvci <2-65535>

Configures the *BSSGP Virtual Circuit Identifier*. It must be unique between all BGSGP connections to one SGSN.

Note
It is up to the system administrator to ensure all PCUs are allocated an unique bvci. OsmoBSC will not ensure this policy.

gprs nsei <0-65535>

Configures the *NS Entity Identifier*. It must be unique between all NS connections to one SGSN.

Note
It is up to the system administrator to ensure all PCUs are allocated an unique bvci. OsmoBSC will not ensure this policy.

gprs nsvc <0-1> nsvci <0-65535>

Configures the *NS Virtual Connection Identifier*. It must be unique between all NS virtual connections to one SGSN.

Note
It is up to the system administrator to ensure all PCUs are allocated an unique nsvci. OsmoBSC will not ensure this policy.

gprs nsvc <0-1> local udp port <0-65535>

Configures the local (PCU side) UDP port for the NS-over-UDP link.

gprs nsvc <0-1> remote udp port <0-65535>

Configures the remote (SGSN side) UDP port for the NS-over-UDP link.

gprs nsvc <0-1> remote ip A.B.C.D

Configures the remote (SGSN side) UDP port for the NS-over-UDP link.

gprs ns timer (tns-block|tns-block-retries|tns-reset|tns-reset-retries|tns-test|tns-alive|tns-alive-retries) <0-255>

Configures the various GPRS NS related timers. Please check the GPRS NS specification for the detailed meaning of those timers.

Dynamic Timeslot Configuration (TCH / PDCH)

A dynamic timeslot is in principle a voice timeslot (TCH) that is used to serve GPRS data (PDCH) when no voice call is active on it. This enhances GPRS bandwidth while no voice calls are active, which is dynamically scaled down as voice calls need to be served. This is a tremendous improvement in service over statically assigning a fixed number of timeslots for voice and data.

The causality is as follows: to establish a voice call, the MSC requests a logical channel of a given TCH kind from the BSC. The BSC assigns such a channel from a BTS' TRX's timeslot of its choice. The knowledge that a given timeslot is dynamic exists only on the BSC level. When the MSC asks for a logical channel, the BSC may switch off PDCH on a dynamic timeslot and then assign a logical TCH channel on it. Hence, though compatibility with the BTS needs to be ensured, any MSC is compatible with dynamic timeslots by definition.

OsmoBSC supports two kinds of dynamic timeslot handling, configured via the `network` / `bts` / `trx` / `timeslot` / `phys_chan_config` configuration. Not all BTS models support dynamic channels.

Table 2: Dynamic timeslot support by various BTS models

	`TCH/F_TCH/H_PDCH`	`TCH/F_PDCH`
ip.access nanoBTS	-	supported
Ericsson RBS	supported	-
sysmoBTS using *osmo-bts-sysmo*	supported	supported
various SDR platforms using *osmo-bts-trx*	supported	supported
Nutaq Litecell 1.5 using *osmo-bts-litecell15*	supported	supported
Octasic OctBTS using *osmo-bts-octphy*	supported	supported

The *OsmoBTS Abis Protocol Specification* [osmobts-abis-spec] describes the non-standard RSL messages used for these timeslot kinds.

Note
Same as for dedicated PDCH timeslots, you need to enable GPRS and operate a PCU, SGSN and GGSN to provide the actual data service.

Osmocom Style Dynamic Timeslots (TCH/F_TCH/H_PDCH)

Timeslots of the `TCH/F_TCH/H_PDCH` type dynamically switch between TCH/F, TCH/H and PDCH, depending on the channel kind requested by the MSC. The RSL messaging for `TCH/F_TCH/H_PDCH` timeslots is compatible with Ericsson RBS.

BTS models supporting this timeslot kind are shown in Table 2.

In the lack of transcoding capabilities, this timeslot type may cause mismatching codecs to be selected for two parties of the same call, which would cause call routing to fail ("`Cannot patch through call with different channel types: local =TCH_F, remote =TCH_H`"). A workaround is to disable TCH/F on this timeslot type, i.e. to allow only TCH/H. To disable TCH/F on Osmocom style dynamic timeslots, use a configuration of

```
network
 dyn_ts_allow_tch_f 0
```

In OsmoNITB, disabling TCH/F on Osmocom dynamic timeslots is the default. In OsmoBSC, the default is to allow both.

ip.access Style Dynamic Timeslots (TCH/F_PDCH)

Timeslots of the `TCH/F_PDCH` type dynamically switch between TCH/F and PDCH. The RSL messaging for `TCH/F_PDCH` timeslots is compatible with ip.access nanoBTS.

BTS models supporting this timeslot kind are shown in Table 2.

Avoid PDCH Exhaustion

To avoid disrupting GPRS, configure at least one timeslot as dedicated PDCH. With only dynamic timeslots, a given number of voice calls would convert all timeslots to TCH, and no PDCH timeslots would be left for GPRS service.

Dynamic Timeslot Configuration Examples

This is an extract of an `osmo-bsc`` config file. A timeslot configuration with five Osmocom style dynamic timeslots and one dedicated PDCH may look like this:

```
network
 bts 0
  trx 0
   timeslot 0
    phys_chan_config CCCH+SDCCH4
   timeslot 1
    phys_chan_config SDCCH8
   timeslot 2
    phys_chan_config TCH/F_TCH/H_PDCH
   timeslot 3
    phys_chan_config TCH/F_TCH/H_PDCH
   timeslot 4
    phys_chan_config TCH/F_TCH/H_PDCH
   timeslot 5
    phys_chan_config TCH/F_TCH/H_PDCH
   timeslot 6
    phys_chan_config TCH/F_TCH/H_PDCH
   timeslot 7
    phys_chan_config PDCH
```

With the ip.access nanoBTS, only `TCH/F_PDCH` dynamic timeslots are supported, and hence a nanoBTS configuration may look like this:

```
network
 bts 0
  trx 0
   timeslot 0
    phys_chan_config CCCH+SDCCH4
   timeslot 1
    phys_chan_config SDCCH8
   timeslot 2
    phys_chan_config TCH/F_PDCH
   timeslot 3
    phys_chan_config TCH/F_PDCH
   timeslot 4
    phys_chan_config TCH/F_PDCH
   timeslot 5
    phys_chan_config TCH/F_PDCH
   timeslot 6
    phys_chan_config TCH/F_PDCH
   timeslot 7
    phys_chan_config PDCH
```


Tuning Access to the BTS

OsmoBSC offers several configuration options to fine-tune access to the BTS. It can allow only a portion of the subscribers access to the network. This can also be used to ramp up access to the network on startup by slowly letting in more and more subscribers. This is especially useful for isolated cells with a huge number of subscribers.

Other options control the behaviour of the MS when it needs to access the random access channel before a dedicated channel is established.

If the BTS is connected to the BSC via a high-latency connection the MS should wait longer for an answer to a RACH request. If it does not the network will have to deal with an increased load due to duplicate RACH requests. However, in order to minimize the delay when a RACH request or response gets lost the MS should not wait too long before retransmitting.

Access Control Class Load Management

Every SIM card is member of one of the ten regular ACCs (0-9). Access to the BTS can be restricted to SIMs that are members of certain ACCs.

Furthermore, high priority users (such as PLMN staff, public or emergency services, etc.) may be members of one or more ACCs from 11-15.

Since the ACCs 0-9 are distributed uniformly across all SIMs, for instance allowing only ACCs 0-4 to connect to the BTS should reduce its load by 50% at the expense of not serving 50% of the subscribers.

The default is to allow all ACCs to connect.

OsmoBSC supports several levels of ACC management to allow or restrict access either permanently or temporarily on each BTS.

The first level of management consists of an access list to flag specific ACCs as permanently barred (the list can be updated at any time through VTY as seen below). As indicated above, the default is to allow all ACCs (0-15).

Example: Restrict permanent access to the BTS by ACC

```
network
 bts 0
  rach access-control-class 1 barred ❶
  rach access-control-class 9 allowed ❷
```

❶ Disallow SIMs with access-class 1 from connecting to the BTS

❷ Permit SIMs with access-class 9 to connect to the BTS.

On really crowded areas, a BTS may struggle to service all mobile stations willing to use it, and which may end up in collapse. In this kind of scenarios it is a good idea to temporarily further restrict the amount of allowed ACCs (hence restrict the amount of subscribers allowed to reach the BTS). However, doing so on a permanent basis would be unfair to subscribers from barred ACCs. Hence, OsmoBSC can be configured to temporarily generate ACC subsets of the permanent set presented above, and rotate them over time to allow fair access to all subscribers. This feature is only aimed at ACCs 0-9, since ACCs 11-15 are considered high priority and hence are always configured based on permanent list policy.

Example: Configure rotative access to the BTS

```
network
 bts 0
  access-control-rotate 3 ❶
  access-control-rotate-quantum 20 ❷
```

❶ Only allow up to 3 concurrent allowed ACCs from the permanent list

❷ Rotate the generated permanent list subsets every 20 seconds in a fair fashion

Furthermore, cells with large number of subscribers and limited overlapping coverage may become overwhelmed with traffic after the cell starts brodacasting. This is especially true in areas with little to no reception from other networks. To manage the load, OsmoBSC has an option to further restrict the rotating ACC subset during startup and slowly increment it over time and taking current load into account.

Example: Ramp up access to the BTS after startup

```
network
 bts 0
  access-control-class-ramping ❶
  access-control-class-ramping-step-interval 30 ❷
  access-control-class-ramping-step-size 1 ❸
```

❶ Turn on access-control-class ramping

❷ Enable more ACCs every 30 seconds

❸ At each step enable one more ACC

Here a full example of all the mechanisms combined can be found:

Example: Full ACC Load Management config setup

```
bts 0
  rach access-control-class 5 barred ❶
  rach access-control-class 6 barred
  rach access-control-class 7 barred
  rach access-control-class 8 barred
  rach access-control-class 9 barred
  access-control-class-rotate 3 ❷
  access-control-class-rotate-quantum 20 ❸
  access-control-class-ramping ❹
  access-control-class-ramping-step-size 1 ❺
  access-control-class-ramping-step-interval dynamic ❻
```

❶ ACCs 5-9 are administratively barred, ie they will never be used until somebody manually enables them in VTY config

❷ Allow access through temporary subsets of len=3 from ACC set 0-4: (0,1,2) → (1,2,3) → (2,3,4) → (3,4,0), etc.

❸ Each subset iteration will happen every 20 seconds

❹ During startup since ramping is enabled, it will further restrict the rotate subset size parameter (len=3)

❺ The rotate subset size parameter will be increased one ACC slot at a time: len=0 → len=1 → len=2 → len=3

❻ The time until the subset size is further increased will be calculated based on current channel load

RACH Parameter Configuration

The following parameters allow control over how the MS can access the random access channel (RACH). It is possible to set a minimum receive level under which the MS will not even attempt to access the network.

The RACH is a shared channel which means multiple MS can choose to send a request at the same time. To minimize the risk of a collision each MS will choose a random number of RACH slots to wait before trying to send a RACH request.

On very busy networks the range this number is chosen from should be high to avoid collisions, but a lower range reduces the overall delay when trying to establish a channel.

The option `rach tx integer N` controls the range from which this number X is chosen. It is `0 <=X < max(8,N)`.

After sending a RACH request the MS will wait a random amount of slots before retransmitting its RACH request. The range it will wait is also determined by the option `rach tx integer N`, but calculating it is not so straightforward. It is defined as `S <=X < S+N` where `S` is determined from a table.

In particular `S` is lowest when `N` is one of 3, 8, 14 or 50 and highest when `N` is 7, 12 or 32.

For more information see *3GPP TA 44.018* [3gpp-ts-44-018] Ch. 3.3.1.1.2 and Table 3.3.1.1.2.1 in particular.

The amount of times the MS attempts to retransmit RACH requests can also be changed. A higher number means more load on the RACH while a lower number can cause channel establishment to fail due to collisions or bad reception.

Example: Configure RACH Access Parameters

```
network
 bts 0
  rxlev access min 20 ❶
  rach tx integer 50❷
  rach max transmission ❸
```

❶ Allow access to the network if the MS receives the BCCH of the cell at -90dBm or better (20dB above -110dBm).

❷ This number affects how long the MS waits before (re-)transmitting RACH requests.

❸ How often to retransmit the RACH request.

OsmoBSC example configuration files

The `osmo-bsc/doc/examples/osmo-bsc` directory in the OpenBSC source tree contains a collection of example configuration files, sorted by BTS type.

This chapter is illustrating some excerpts from those examples

Example configuration for OsmoBSC with one single-TRX nanoBTS

Example 10.1 OsmoBSC with one single-TRX nanoBTS

```
e1_input
 e1_line 0 driver ipa ❶
network
 network country code 1
 mobile network code 1
 encryption a5 0
 neci 1
 handover 0
 bts 0
  type nanobts ❷
  band DCS1800 ❸
  cell_identity 0
  location_area_code 1
  training_sequence_code 7
  base_station_id_code 63
  ms max power 15
  cell reselection hysteresis 4
  rxlev access min 0
  channel allocator ascending
  rach tx integer 9
  rach max transmission 7
  ipa unit-id 1801 0 ❹
  oml ipa stream-id 255 line 0
  gprs mode none
  trx 0
   rf_locked 0
   arfcn 871 ❺
```



```
   nominal power 23
   max_power_red 20 ❻
   rsl e1 tei 0
    timeslot 0
     phys_chan_config CCCH+SDCCH4
    timeslot 1
     phys_chan_config SDCCH8
    timeslot 2
     phys_chan_config TCH/F
    timeslot 3
     phys_chan_config TCH/F
    timeslot 4
     phys_chan_config TCH/F
    timeslot 5
     phys_chan_config TCH/F
    timeslot 6
     phys_chan_config TCH/F
    timeslot 7
     phys_chan_config TCH/F
```

❶ You have to configure one virtual E1 line with the IPA driver in order to use Abis/IP. One e1_line is sufficient for any number of A-bis/IP BTSs, there is no limit like in physical E1 lines.

❷ The BTS type must be set using `type nanobts`

❸ The GSM band must be set according to the BTS hardware.

❹ The IPA Unit ID parameter must be set to what has been configured on the BTS side using the *BTS Manager* or `ipaccess-config`.

❺ The ARFCN of the BTS.

❻ All known nanoBTS units have a nominal transmit power of 23 dBm. If a `max_power_red` of 20 (dB) is configured, the resulting output power at the BTS Tx port is 23 - 20 = 3 dBm.

Note
The `nominal_power` setting does *not* influence the transmitted power to the BTS! It is a setting by which the system administrator tells the BSC about the nominal output power of the BTS. The BSC uses this as basis for calculations.

Example configuration for OsmoBSC with multi-TRX nanoBTS

Example 10.2 OsmoBSC configured for dual-TRX (stacked) nanoBTS

```
e1_input
 e1_line 0 driver ipa
network
 network country code 1
 mobile network code 1
 encryption a5 0
 neci 1
 handover 0
 bts 0
  type nanobts
  band DCS1800
  cell_identity 0
  location_area_code 1
  training_sequence_code 7
  base_station_id_code 63
```



```
 ms max power 15
 cell reselection hysteresis 4
 rxlev access min 0
 channel allocator ascending
 rach tx integer 9
 rach max transmission 7
 ipa unit-id 1800 0 ❶
 oml ipa stream-id 255 line 0
 gprs mode none
 trx 0
  rf_locked 0
  arfcn 871
  nominal power 23
  max_power_red 0
  rsl e1 tei 0
   timeslot 0
    phys_chan_config CCCH+SDCCH4
   timeslot 1
    phys_chan_config SDCCH8
   timeslot 2
    phys_chan_config TCH/F
   timeslot 3
    phys_chan_config TCH/F
   timeslot 4
    phys_chan_config TCH/F
   timeslot 5
    phys_chan_config TCH/F
   timeslot 6
    phys_chan_config TCH/F
   timeslot 7
    phys_chan_config TCH/F
 trx 1
  rf_locked 0
  arfcn 873
  nominal power 23
  max_power_red 0
  rsl e1 tei 0
   timeslot 0
    phys_chan_config SDCCH8
   timeslot 1
    phys_chan_config TCH/F
   timeslot 2
    phys_chan_config TCH/F
   timeslot 3
    phys_chan_config TCH/F
   timeslot 4
    phys_chan_config TCH/F
   timeslot 5
    phys_chan_config TCH/F
   timeslot 6
    phys_chan_config TCH/F
   timeslot 7
    phys_chan_config TCH/F
```

❶ In this example, the IPA Unit ID is specified as `1800 0`. Thus, the first nanoBTS unit (`trx 0`) needs to be configured to 1800/0/0 and the second nanoBTS unit (`trx 1`) needs to be configured to 1800/0/1. You can configure the BTS unit IDs using the `ipaccess-config` utility included in OsmoBSC.

Note
For building a multi-TRX setup, you also need to connect the TIB cables between the two nanoBTS units, as well as the coaxial/RF AUX cabling.

BSC level configuration

Hand-over

Hand-over in GSM

Hand-over is the process of changing a MS with a currently active dedicated channel from one BTS to another BTS. As opposed to idle mode, where the MS autonomously performs cell re-selection, in dedicated mode this happens under network control.

In order to determine when to perform hand-over, and to which cells, the network requests the MS to perform measurements on a list of neighbor cell channels, which the MS then reports back to the network in the form of GSM RR *Measurement Result* messages. Those messages contain the downlink measurements as determined by the MS.

Furthermore, the BTS also performs measurements on the uplink, and communicates those by means of RSL to the BSC.

The hand-over decision is made by an algorithm that processes those measurement results and determines when to perform the hand-over.

Configuration of hand-over in OsmoBSC

OsmoBSC only support so-called intra-BSC hand-over, where the hand-over is performed between two BTSs within the same BSC.

Hand-over is enabled and configured by the use of a set of `handover` commands. Using those, you can tune the key parameters of the hand-over algorithm and adapt it to your specific environment.

Example handover configuration snippet

```
handover 1 ❶
handover window rxlev averaging 10 ❷
handover window rxqual averaging 1 ❸
handover window rxlev neighbor averaging 10 ❹
handover power budget interval 6 ❺
handover power budget hysteresis 3 ❻
handover maximum distance 9999 ❼
```

❶ Enable hand-over

❷ Set the RxLev averaging window for the serving cell to 10 measurements

❸ Set the RxQual averaging window for the serving cell to 1 measurement (no window)

❹ Set the RxLev averaging for neighbor cells to 10 measurements

❺ Check for the conditions of a power budget hand-over every 6 SACCH frames

❻ A neighbor cell must be at least 3 dB stronger than the serving cell to be considered a candidate for hand-over

❼ Perform a maximum distance hand-over if TA is larger 9999 (i.e. never)

Timer Configuration

The GSM specification specifies a variety of timers both on the network as well as on the mobile station side.

Those timers can be configured using the `timer tXXXX` command.

Table 3: Configurable Timers

node	timer	default	description
network	t3101	10	Timeout for *Immediate Assignment* (sec)
network	t3103	?	Timeout for Handover (sec)
network	t3105	40	Repetition of *Physical Information* (sec)
network	t3107	?	?
network	t3109	?	RSL SACCH deactivation timeout (sec)
network	t3111	?	RSL timeout to wait before releasing the RF channel (sec)
network	t3113	60	Time to try paging for a subscriber (sec)
network	t3115	?	?
network	t3117	?	?
network	t3119	?	?
network	t3122	10	Waiting time after *Immediate Assignment Reject*
network	t3141	?	?

Discontinuous Transmission (DTX)

GSM provides a full-duplex voice call service. However, in any civilized communication between human beings, only one of the participants is speaking at any given point in time. This means that most of the time, one of the two directions of the radio link is transmitting so-called *silence frames*.

During such periods of quiescence in one of the two directions, it is possible to suppress transmission of most of the radio bursts, as there is no voice signal to transport. GSM calls this feature *Discontinuous Transmission*. It exists separately for uplink (DTXu) and downlink (DTXd).

Downlink DTX is only permitted on non-primary transceivers (!= TRX0), as TRX0 must always transmit at constant output power to ensure it is detected during cell selection.

Uplink DTX is possible on any TRX, and serves primarily two uses:

possible on any TRX, and serves primarily two uses:

1. reducing the MS battery consumption by transmitting at a lower duty cycle
2. reducing the uplink interference caused in surrounding cells that re-use the same ARFCN.

DTS for both uplink and downlink is implemented in the BTS. Not all BTS models support it.

The Osmocom BSC component can instruct the BTS to enable or disable uplink and/or downlink DTX by means of A-bis OML.

Handover

Handover is the process of moving a continuously used channel (lchan) from one cell to another. Usually, that is an ongoing call, so that phones are able to move across cell coverage areas without interrupting the voice transmission.

A handover can

- stay within one given cell (intra-cell, i.e. simply a new RR Assignment Command);
- occur between two cells that belong to the same BSS (intra-BSC, via RR Handover Command);
- cross BSS boundaries (inter-BSC, via BSSMAP handover procedures);
- move to another MSC (inter-MSC, inter-PLMN);
- move to another RAN type, e.g. from 2G to 3G (inter-RAT, inter-Radio-Access-Technology).

The physical distance is by definition always very near, but handover negotiation may range from being invisible to the MSC all the way to orchestrating completely separate RAN stacks.

OsmoBSC currently supports handover within one BSS and between separate BSS. Whether inter-MSC is supported depends on the MSC implementation (to the BSC, inter-MSC handover looks identical to inter-BSC handover). Inter-RAT handover is currently not implemented. However, you may still advertise 3G and 4G neighbor cells in order to facilitate cell/RAT re-selection to those neighbors.

Since 2019, OsmoMSC fully supports both inter-BSC and inter-MSC handover.

Table 4: Handover support in Osmocom at the time of writing

	intra-BSC HO (local BSS)	inter-BSC HO (remote BSS)	inter-MSC HO	inter-RAT HO
OsmoBSC	rxlev, load-based	rxlev	(planned)	-
OsmoMSC	(not involved, except for codec changes)	(planned)	(planned)	-

Most handover related procedures are explained in 3GPP TS 48.008.

How Handover Works

This chapter generally explains handover operations between 2G cells.

Internal / Intra-BSC Handover

The BSS is configured to know which cell is physically adjacent to which other cells, its "neighbors". On the MS/BTS/BSS level, individual cells are identified by ARFCN+BSIC (frequency + 6-bit identification code).

The BSC instructs each BTS with a list of ARFCNs (i.e. GSM frequency bands) that qualify as neighbor cells, as part of the System Information Type 2. Each MS served by a BTS receives the System Information Type 2 and thus knows which ARFCNs to measure for potential handover. Each MS with an active channel then returns up to 6 measurements of reception levels (RXLEV) to the BTS, to be forwarded to the BSC in RSL Measurement Report messages.

Note that the BTS and MS are told only the ARFCNs, not the BSICs, of neighbor cells; the BSICs are however included in the measurements that an MS returns to BTS and BSC. Commonly, each ARFCN is owned by one specific operator, so, an MS considers all visible cells on a given ARFCN as possible neighbors. However, as soon as an MS reports RXLEV of a specific neighbor cell, the BSC needs to know which exact cell to possibly handover to, which is why the MS pinpoints the specific BSIC that it reported measurements for.

The BSC is the point of decision whether to do handover or not. This can be a hugely complex combination of heuristics, knowledge of cell load and codec capabilities. The most important indicator for handover though is: does an MS report a neighbor with a better signal than the current cell? See Figure 4.

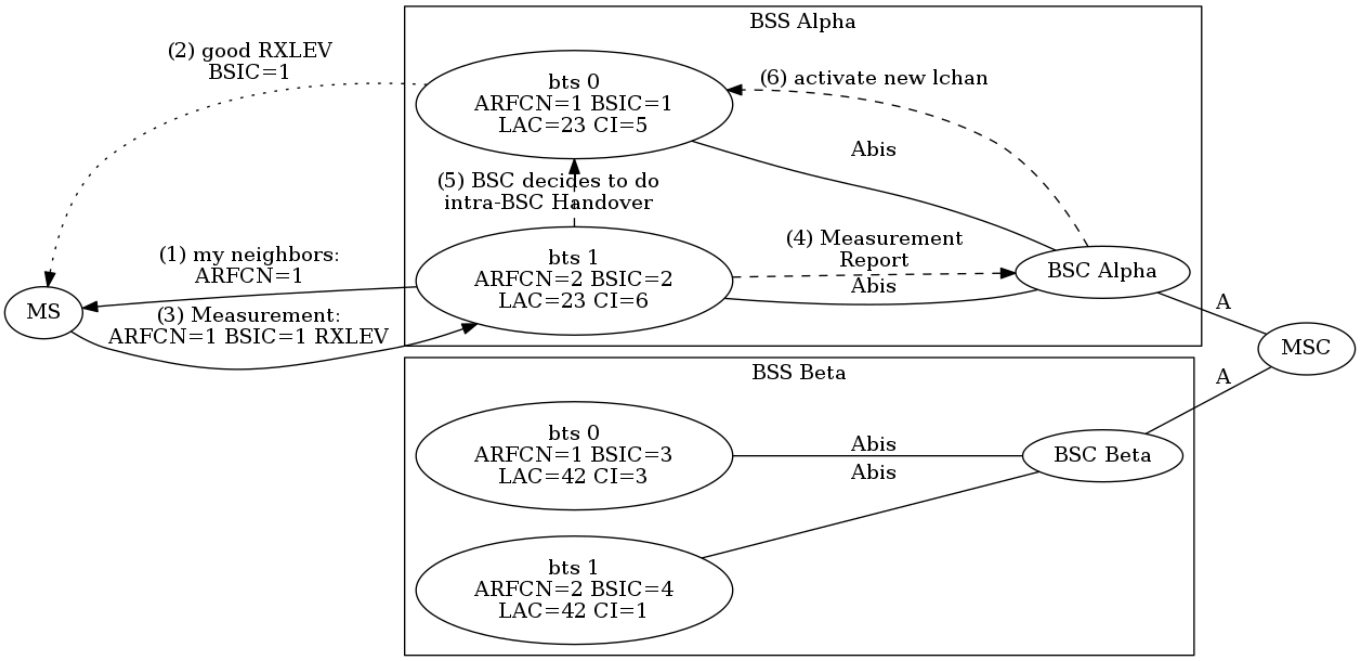


Figure 4: Intra-BSC Handover stays within the BSS (shows steps only up to activation of the new lchan — this would be followed by an RR Handover Command, RACH causing Handover Detection, Handover Complete, . . .)

If the BSC sees the need for handover, it will:

- activate a new lchan (with a handover reference ID),
- send an RR Handover Command to the current lchan, and
- wait for the MS to send a Handover RACH to the new lchan ("Handover Detect").
- The RTP stream then is switched over to the new lchan,
- an RSL Establish Indication is expected on the new lchan,
- and the old lchan is released.

Should handover fail at any point, e.g. the new lchan never receives a RACH, or the MS reports a Handover Failure, then the new lchan is simply released again, and the old lchan remains in use. If the RTP stream has already been switched over to the new lchan, it is switched back to the old lchan.

This is simple enough if the new cell is managed by the same BSC: the OsmoMGW is simply instructed to relay the BTS-side of the RTP stream to another IP address and port, and the BSC continues to forward DTAP to the MSC transparently. The operation happens completely within the BSS, except for the BSSMAP Handover Performed message sent to the MSC once the handover is completed (see 3GPP TS 48.008).

External / Inter-BSC Handover

If the handover target cell belongs to a different BSS, the RR procedure for handover remains the same, but we need to tell the *remote* BSC to allocate the new lchan.

The only way to reach the remote BSC is via the MSC, so the MSC must be able to:

- identify which other BSC we want to talk to,

- forward various BSSMAP Handover messages between old and new BSC,
- redirect the core-side RTP stream to the new BSS at the appropriate time,
- and must finally BSSMAP Clear the connection to the old BSS to conclude the inter-BSC handover.

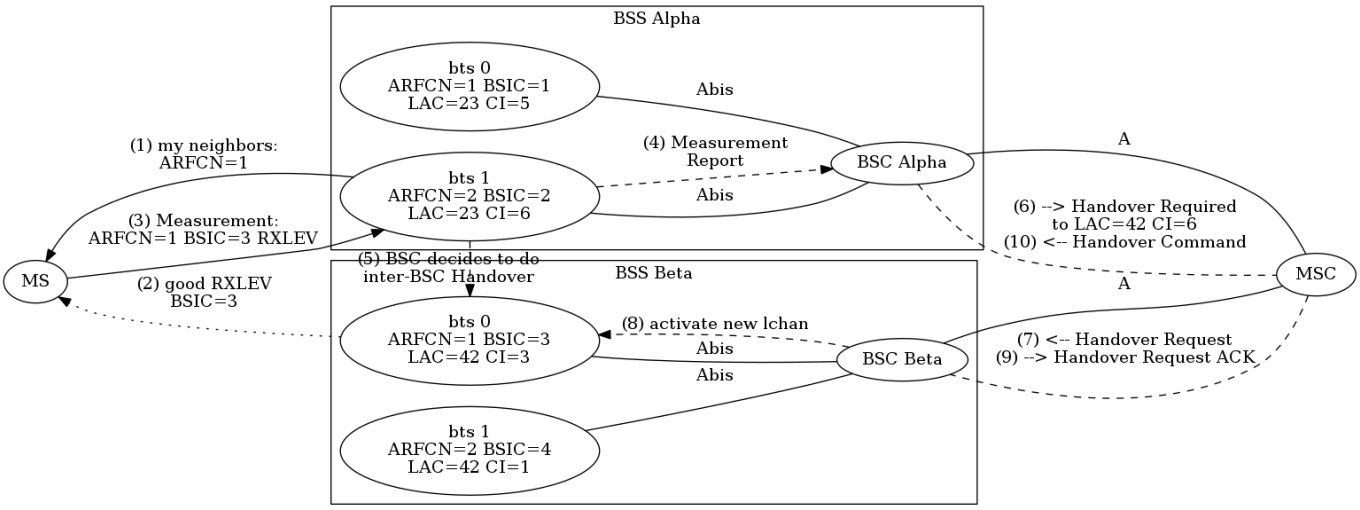


Figure 5: Inter-BSC Handover requires the MSC to relay between two BSCs (shows steps only up to the BSSMAP Handover Command — this would be followed by an RR Handover Command, RACH causing Handover Detection, Handover Complete, . . .)

The first part, identifying the remote BSC, is not as trivial as it sounds: as mentioned above, on the level of cell information seen by BTS and MS, the neighbor cells are identified by ARFCN+BSIC. However, on the A-interface and in the MSC, there is no knowledge of ARFCN+BSIC configurations. Instead, each cell is identified by a LAC and CI (Location Area Code and Cell Identifier).

Note
There are several different cell identification types on the A-interface: from Cell Global Identifier (MCC+MNC+LAC+CI) down to only LAC. OsmoBSC supports most of these (see Table 5). For simplicity, this description focuses on LAC+CI identification.

Hence:

- the BSC needs to know which remote-BSS cells' ARFCN+BSIC correspond to exactly which global LAC+CI, and
- the MSC needs to know which LAC+CI are managed by which BSC.

In other words, each BSC requires prior knowledge about the cell configuration of its remote-BSS neighbor cells, and the MSC requires prior knowledge about each BSC's cell identifiers; i.e. these config items are spread reduntantly.

The most obvious reason for using LAC+CI in BSSMAP is that identical ARFCN+BSIC are typically re-used across many cells of the same network operator: an operator will have only very few ARFCNs available, and the 6bit BSIC opens only a very limited range of distinction between cells. As long as each cell has no more than one neighbor per given ARFCN+BSIC, these values can be re-used any number of times across a network, and even between cells managed by one and the same BSC.

Configuring Neighbors

The most important step to enable handover in OsmoBSC is to configure each cell with the ARFCN+BSIC identities of its adjacent neighbors — both local-BSS and remote-BSS.

For a long time, OsmoBSC has offered configuration to manually enter the ARFCN+BSIC sent out as neighbors on various System Information messages (all `neighbor-list` related commands). This is still possible; however, particularly for re-using ARFCN+BSIC within one BSS, this method will not work well.

With the addition of inter-BSC handover support, the new `neighbor` config item has been added to the `bts` config node, to maintain explicit cell-to-cell neighbor relations, with the possibility to re-use ARFCN+BSIC in each cell.

It is recommended to completely replace `neighbor-list` configurations with the new `neighbor` configuration described below.

Table 5: Overview of neighbor configuration on the `bts` config node

Local	Remote BSS	type of `neighbor` config line, by example
✓		`neighbor bts 5`
✓		`neighbor lac 200`
✓		`neighbor lac-ci 200 3`
✓		`neighbor cgi 001 01 200 3`
✓	✓	`neighbor lac 200 arfcn 123 bsic 1`
✓	✓	`neighbor lac-ci 200 3 arfcn 123 bsic 1`
✓	✓	`neighbor cgi 001 01 200 3 arfcn 123 bsic 1`

Default: All Local Cells are Neighbors

For historical reasons, the default behavior of OsmoBSC is to add all local-BSS cells as neighbors for every other cell. To maintain a backwards compatible configuration file format, this is still the case: as long as no explicit neighbor cell is configured with a `neighbor` command (either none was configured, or all configured `neighbor` lines have been removed again), a cell automatically lists all of the local-BSS cells as neighbors. These are implicit mappings in terms of the legacy neighbor configuration scheme, and re-using ARFCN+BSIC combinations within a BSS will not work well this way.

As soon as the first explicit `neighbor` relation is added to a cell, the legacy behavior is switched off, and only explicit neighbors are in effect.

Note
If a cell is required to not have any neighbors, it is recommended to switch off handover for that cell with `handover 0`.

Local-BSS Neighbors

Local neighbors can be configured by just the local BTS number, or by LAC+CI, or any other supported A-interface type cell identification; also including the ARFCN+BSIC is optional, it will be derived from the local configuration if omitted.

OsmoBSC will log errors in case the configuration includes ambiguous ARFCN+BSIC relations (when one given cell has more than one neighbor for any one ARFCN+BSIC).

Neighbor relations must be configured explicitly in both directions, i.e. each cell has to name all of its neighbors, even if the other cell already has an identical neighbor relation in the reverse direction.

Example: configuring neighbors within the local BSS in osmo-bsc.cfg, identified by local BTS number

```
network
 bts 0
  neighbor bts 1
 bts 1
  neighbor bts 0
```

Example: configuring neighbors within the local BSS in osmo-bsc.cfg, identified by LAC+CI


```
network

 bts 0
  # this cell's LAC=23 CI=5
  location_area_code 23
  cell_identity 5
  # reference bts 1
  neighbor lac-ci 23 6

 bts 1
  # this cell's LAC=23 CI=6
  location_area_code 23
  cell_identity 6
  # reference bts 0
  neighbor lac-ci 23 5
```

It is allowed to include the ARFCN and BSIC of local neighbor cells, even though that is redundant with the already known local configuration of the target cell. The idea is to ease generating the neighbor configuration automatically, in that local-BSS and remote-BSS neighbors can have identical configuration formatting. If the cell identification (LAC+CI) matches a local cell but a mismatching ARFCN+BSIC follows on the same config line, OsmoBSC will report errors. For human readability and maintainability, it may instead be desirable to use the `neighbor bts <0-255>` format, or omit the redundant `arfcn` and `bsic`.

Example: configuring neighbors within the local BSS in osmo-bsc.cfg, redundantly identified by LAC+CI as well as ARFCN+BSIC

```
network

 bts 0
  # this cell's LAC=23 CI=5
  location_area_code 23
  cell_identity 5
  # this cell's ARFCN=1 BSIC=1
  trx 0
   arfcn 1
  base_station_id_code 1
  # reference bts 1
  neighbor lac-ci 23 6 arfcn 2 bsic 2

 bts 1
  # LAC=23 CI=6
  location_area_code 23
  cell_identity 6
  # this cell's ARFCN=2 BSIC=2
  trx 0
   arfcn 2
  base_station_id_code 2
  # reference bts 0
  neighbor lac-ci 23 5 arfcn 1 bsic 1
```

Remote-BSS Neighbors

Remote-BSS neighbors always need to be configured with full A-interface identification *and* ARFCN+BSIC, to allow mapping a cell's neighbor ARFCN+BSIC to a BSSMAP Cell Identifier (see 3GPP TS 48.008 3.1.5.1 Handover Required Indication and 3.2.1.9 HANDOVER REQUIRED).

Example: configuring remote-BSS neighbors in osmo-bsc.cfg, identified by LAC+CI (showing both BSCs' configurations)

```
# BSC Alpha's osmo-bsc.cfg
network
```



```
 bts 0
  # this cell's LAC=23 CI=6
  location_area_code 23
  cell_identity 6
  # this cell's ARFCN=2 BSIC=2
  trx 0
   arfcn 2
  base_station_id_code 2
  # fully describe the remote cell by LAC+CI and ARFCN+BSIC
  neighbor lac-ci 42 3 arfcn 1 bsic 3

# BSC Beta's osmo-bsc.cfg
network
 bts 0
  # this cell's LAC=42 CI=3
  location_area_code 42
  cell_identity 3
  # this cell's ARFCN=1 BSIC=3
  trx 0
   arfcn 1
  base_station_id_code 3
  # fully describe the remote cell by LAC+CI and ARFCN+BSIC
  neighbor lac-ci 23 6 arfcn 2 bsic 2
```

Note
It is strongly recommended to stick to a single format for remote-BSS neighbors' cell identifiers all across an OsmoBSC configuration; i.e. decide once to use `lac`, `lac-ci` or `cgi` and then stick to that within a given osmo-bsc.cfg. The reason is that the *Cell Identifier List* sent in the *BSSMAP Handover Required* message must have one single cell identifier type for all list items. Hence, to be able to send several alternative remote neighbors to the MSC, the configured cell identifiers must be of the same type. If in doubt, use the full CGI identifier everywhere.

Reconfiguring Neighbors in a Running OsmoBSC

When modifying a cell's neighbor configuration in a telnet VTY session while a cell is already active, the neighbor configuration will merely be cached in the BSC's local config. To take actual effect, it is necessary to

- either, re-connect the cell to the BSC (e.g. via `drop bts connection <0-255> oml`)
- or, re-send the System Information using `bts <0-255> resend-system-information`.

Configuring Handover Decisions

For a long time, OsmoBSC has supported handover based on reception level hysteresis (RXLEV) and distance (TA, Timing Advance), known as `algorithm 1`.

Since 2018, OsmoBSC also supports a load-based handover decision algorithm, known as `algorithm 2`, which also takes cell load, available codecs and oscillation into consideration. Algorithm 2 had actually been implemented for the legacy OsmoNITB program many years before the OsmoMSC split, but remained on a branch, until it was forward-ported to OsmoBSC in 2018.

Table 6: What handover decision algorithms take into account

algorithm 1	algorithm 2	
✓	✓	RXLEV
✓	✓	RXQUAL
✓	✓	TA (distance)

Table 6: (continued)

✓	✓	interference (good RXLEV, bad RXQUAL)
	✓	load (nr of free lchans, minimum RXLEV and RXQUAL)
	✓	penalty time to avoid oscillation
	✓	voice rate / codec bias
✓		inter-BSC: RXLEV hysteresis
	✓	inter-BSC: only below minimum RXLEV, RXQUAL

Common Configuration

Handover is disabled by default; to disable/enable handover, use `handover (0|1)`.

Once enabled, algorithm 1 is used by default; choose a handover algorithm with `handover algorithm (1|2)`:

```
network
 # Enable handover
 handover 1

 # Choose algorithm
 handover algorithm 2

 # Tweak parameters for algorithm 2 (optional)
 handover2 min-free-slots tch/f 4
 handover2 penalty-time failed-ho 30
 handover2 retries 1
```

All handover algorithms share a common configuration scheme, with an overlay of three levels:

- immutable compile-time default values,
- configuration on the `network` level for all cells,
- individual cells' configuration on each `bts` node.

Configuration settings relevant for algorithm 1 start with `handover1`, for algorithm 2 with `handover2`.

The following example overrides the compile-time default for all cells, and furthermore sets one particular cell on its own individual setting, for the `min-free-slots tch/f` value:

```
network
 handover2 min-free-slots tch/f 4
 bts 23
  handover2 min-free-slots tch/f 2
```

The order in which these settings are issued makes no difference for the overlay; i.e., the following configuration is perfectly identical to the above, and the individual cell's value remains in force:

```
network
 bts 23
  handover2 min-free-slots tch/f 2
 handover2 min-free-slots tch/f 4
```

Each setting can be reset to a default value with the `default` keyword. When resetting an individual cell's value, the globally configured value is used. When resetting the global value, the compile-time default is used (unless individual cells still have explicit values configured). For example, this telnet VTY session removes above configuration first from the cell, then from the global level:


```
OsmoBSC(config)# network
OsmoBSC(config-net)# bts 23
OsmoBSC(config-net-bts)# handover2 min-free-slots tch/f default
% 'handover2 min-free-slots tch/f' setting removed, now is 4
OsmoBSC(config-net-bts)# exit
OsmoBSC(config-net)# handover2 min-free-slots tch/f default
% 'handover2 min-free-slots tch/f' setting removed, now is 0
```

Handover Algorithm 1

Algorithm 1 takes action only when RR Measurement Reports are received from a BTS. As soon as a neighbor's average RXLEV is higher than the current cell's average RXLEV plus a hysteresis distance, handover is triggered.

If a handover fails, algorithm 1 will again attempt handover to the same cell with the next Measurement Report received.

Configuration settings relevant for algorithm 1 start with `handover1`. For further details, please refer to the OsmoBSC VTY Reference ([vty-ref-osmobsc]) or the telnet VTY online documentation. See the `handover1` settings on the `config-net` and `config-net-bts` nodes.

Handover Algorithm 2

Algorithm 2 is specifically designed to distribute load across cells. A subscriber will not necessarily remain attached to the cell that has the best RXLEV average, if that cell is heavily loaded and a less loaded neighbor is above the minimum allowed RXLEV.

Algorithm 2 also features penalty timers to avoid oscillation: for each subscriber, if handover to a specific neighbor failed (for a configurable number of retries), a holdoff timer prevents repeated attempts to handover to that same neighbor. Several hold-off timeouts following specific situations are configurable (see `handover2 penalty-time` configuration items).

Configuration settings relevant for algorithm 2 start with `handover2`. For further details, please refer to the OsmoBSC VTY Reference [vty-ref-osmobsc] or the telnet VTY online documentation. See the `handover2` settings on the `config-net` and `config-net-bts` nodes.

Load Distribution

Load distribution is only supported by algorithm 2.

Load distribution occurs:

- explicitly: every N seconds, OsmoBSC considers all local cells and actively triggers handover operations to reduce congestion, if any. See `min-free-slots` below, and the `congestion-check` setting.
- implicitly: when choosing the best neighbor candidate for a handover triggered otherwise, a congested cell (in terms of `min-free-slots`) is only used as handover target if there is no alternative that causes less cell load.

In either case, load distribution will only occur towards neighbor cells that adhere to minimum reception levels and distance, see `min rxlev` and `max distance`.

Load distribution will take effect only for already established channels. For example, an MS will always first establish a voice call with its current cell choice; in load situations, it might be moved to another cell shortly after that. Considering the best neighbor *before* starting a new voice call might be desirable, but is currently not implemented. Consider that RXLEV/RXQUAL ratings are averaged over a given number of measurement reports, so that the neighbor ratings may not be valid/reliable yet during early call establishment. In consequence, it is recommended to ensure a sufficient number of unused logical channels at all times, though there is no single correct configuration for all situations.

Most important for load distribution are the `min-free-slots tch/f` and `min-free-slots tch/h` settings. The default is zero, meaning *no* load distribution. To enable, set `min-free-slots` >= 1 for `tch/f` and/or `tch/h` as appropriate. This setting refers to the minimum number of voice channels that should ideally remain unused in each individual BTS at all times.

Note
it is not harmful to configure `min-free-slots` for a TCH kind that is not actually present. Such settings will simply be ignored.

Note
the number of TCH/F timeslots corresponds 1:1 to the number indicated by `min-free-slots tch/f`, because each TCH/F physical channel has exactly one logical channel. In contrast, for each TCH/H timeslot, there are two logical channels, hence `min-free-slots tch/h` corresponds to twice the number of TCH/H timeslots configured per cell. In fact, a more accurate naming would have been "min-free-lchans".

Think of the `min-free-slots` setting as the threshold at which load distribution is considered. If as many logical channels as required by this setting are available in a given cell, only changes in RXLEV/RXQUAL/TA trigger handover away from that cell. As soon as less logical channels remain free, the periodical congestion check attempts to distribute MS to less loaded neighbor cells. Every time, the one MS that will suffer the least RXLEV loss while still reducing congestion will be instructed to move first.

If a cell and its neighbors are all loaded past their `min-free-slots` settings, the algorithmic aim is equal load: a load-based handover will never cause the target cell to be more congested than the source cell.

The min-free-slots setting is a tradeoff between immediate voice service availability and optimal reception levels. A sane choice could be:

- Start off with `min-free-slots` set to half the available logical channels.
- Increase `min-free-slots` if you see MS being rejected too often even though close neighbors had unused logical channels.
- Decrease `min-free-slots` if you see too many handovers happening for no apparent reason.

Choosing the optimal setting is not trivial, consider these examples:

- Configure `min-free-slots` = 1: load distribution to other cells will occur exactly when the last available logical channel has become occupied. The next time the congestion check runs, at most one handover will occur, so that one channel is available again. In the intermediate time, all channels will be occupied, and some MS might be denied immediate voice service because of that, even though, possibly, other neighbor cells would have provided excellent reception levels and were completely unloaded. For those MS that are already in an ongoing voice call and are not physically moving, though, this almost guarantees service by the closest/best cell.
- Set `min-free-slots` = 2: up to two MS can successfully request voice service simultaneously (e.g. one MS is establishing a new voice call while another MS is travelling into this cell). Ideally, two slots have been kept open and are immediately available. But if a third MS is also traveling into this cell at the same time, it will not be able to handover into this cell until load distribution has again taken action to make logical channels available. The same scenario applies to any arbitrary number of MS asking for voice channels simultaneously. The operator needs to choose where to draw the line.
- Set `min-free-slots` >= the number of available channels: as soon as any neighbor is less loaded than a given cell, handover will be attempted. But imagine there are only two active voice calls on this cell with plenty of logical channels still unused, and the closest neighbor rates only just above `min rxlev`; then moving one of the MS *for no good reason* causes all of: increased power consumption, reduced reception stability and channel management overhead.

Note
In the presence of dynamic timeslots to provide GPRS service, the number of voice timeslots left unused also determines the amount of bandwidth available for GPRS.

External / Inter-BSC Handover Considerations

There currently is a profound difference for inter-BSC handover between algorithm 1 and 2:

For algorithm 1, inter-BSC handover is triggered as soon as the Measurement Reports and hysteresis indicate a better neighbor than the current cell, period.

For algorithm 2, a subscriber is "sticky" to the current BSS, and inter-BSC handover is only even considered when RXLEV/TA drop below minimum requirements.

- If your network topology is such that each OsmoBSC instance manages a single BTS, and you would like to encourage handover between these, choose handover algorithm 1. Load balancing will not be available, but RXLEV hysteresis will.
- If your network topology has many cells per BSS, and/or if your BSS boundaries in tendency correspond to physical/semantic boundaries that favor handover to remain within a BSS, then choose handover algorithm 2.

The reason for the difference between algorithm 1 and 2 for remote-BSS handovers is, in summary, the young age of the inter-BSC handover feature in OsmoBSC:

- So far the mechanisms to communicate cell load to remote-BSS available in the BSSMAP Handover messages are not implemented, so, a handover due to cell load across BSS boundaries would be likely to cause handover oscillation between the two BSS (continuous handover of the same MS back and forth between the same two cells).
- Algorithm 1 has no `min rxlev` setting.
- Algorithm 1 does not actually use any information besides the Measurement Reports, and hence can trivially treat all neighbor cells identically.

Advertising 3G/4G neighbors

Despite osmo-bsc not supporting inter-RAT hand-over at this point, it still makes sense to advertise neighbor cells of other network technologies like UMTS/UTRAN (3G) and LTE/EUTRAN (4G). This will help phones with idle-mode re-selection of the best available radio access technology (RAT).

For more information on the inter-RAT cell re-selection algorithm and its parameters, see 3GPP TS 45.008 - particularly Section 6.6.4 describing measurements on cells of other (non-GSM) RATs.

Such neighbors are advertised as part of the SI2quater (System Information Type 2quater).

UMTS/UTRAN/3G neighbors

In order to advertise a 3G neighbor cell you have to specify the following properties:

- the UARFCN (UTRAN Absolute Radio Channel Number) on which the cell broadcasts
- the Scrambling Code of the cell
- whether or not the cell uses diversity

In the following example, we're configuring a 3G neighbor cell on UARFCN 1234 using the scrambling code 511 with no diversity:

```
network
 bts 0
  si2quater neighbor-list add uarfcn 1234 511 0
```

3G neighbor cells can be removed using the same command, just replacing `add` with `del`.

LTE/EUTRAN/4G neighbors

In order to advertise a 4G neighbor cell you have to specify the following properties:

- EARFCN (EUTRAN Absolute Radio Channel Number) on which the cell broadcasts
- Reselection thresholds towards E-UTRAN cells:

0	0 dB
1	2 dB
2	4 dB
3	6 dB
...	...
31	62 dB

- Priority of E-UTRAN frequency: 0 = lowest priority, ..., 7 = highest priority
- QRXLEVMIN parameter: Minimum required RX level in the UTRAN FDD cell (dBm), see 3GPP TS 25.304.

0	-140 dBm
1	-138 dBm
2	-136 dBm
...	...
31	-78 dBm

- Measurement bandwidth in MHz, see 3GPP TS 44.018 and 3GPP TS 44.060. This field specifies the minimum value of the channel bandwidth of all valid E-UTRAN cells on the specified EARFCN. It is defined by the parameter Transmission Bandwidth Configuration, N RB (see 3GPP TS 36.104). The values indicate the number of resource blocks over which the mobile station could measure if the mobile station does not support wideband RSRQ measurements (see 3GPP TS 24.008). A mobile station supporting wideband RSRQ measurements shall measure over the indicated number of resource blocks. The field is coded according to the following table:

0	N_RB = 6
1	N_RB = 15
2	N_RB = 25
3	N_RB = 50
4	N_RB = 75
5	N_RB = 100

In the following example we're configuring a 4G neighbor on EARFCN 2342 with a higher reselection threshold of 40dB, a lower reselection threshold of 20dB, priority 5, QRXLEVMIN of -140 dBm and a measurement bandwidth of 100 resource blocks:

```
network
 bts 0
  si2quater neighbor-list add earfcn 2342 thresh-hi 20 thresh-lo 10 prio 5 qrxlv 0 meas 5
```

4G neighbor cells can be removed using the same command, just replacing `add` with `del`.

SMSCB (Cell Broadcast)

OsmoBSC supports SMS Cell Broadcast (SMSCB) services (CBS). This includes the CBSP protocol to interact with a CBC (Cell Broadcast Centre) such as OsmoCBC, as well as the scheduling of SMSCB messages on both the BASIC and EXTENDED CBCH and transmission of related RSL messages to the attached BTS.

More high-level information can be found at https://en.wikipedia.org/wiki/Cell_Broadcast and the related specification is [?].

In order to use SMSCB with OsmoBSC, you will need to

- Configure OsmoBSC as either CBSP server or client
- Use a channel combination including a CBCH on the BTSs

Enabling a CBCH channel combination

On the Um interface, SMSCB are transmitted via the CBCH (Cell Broadcast Channel). The CBCH is a separate downlink-only logical channel which must be activated on any of the BTSs requiring CBSP support.

The channel combination is configured in the `timeslot` node of each TRX.

The two `phys_chan_config` supporting CBCH are `CCCH+SDCCH4+CBCH` and `SDCCH/8+CBCH`. Please note that the CBCH steals one of the SDCCH, so a SDCCH/4 will only have three remaining SDCCH, and a SDCCH/8 will have only seven remaining SDCCH.

Configuring the CBSP connection

CBSP is the protocol between BSC and CBC. It operates over TCP.

According to 3GPP TS 48.049, a BSC typically operates as a TCP server, and the CBC connects as TCP client. This would require the CBC to have out-of-band knowledge of all the BSCs in the network (and their IP addresses).

In order to comply with the specifications, OsmoBSC supports this mode of operation as CBSP TCP server. However, to make network operation and configuration more simple, it also can operate in TCP client mode, connecting to the CBC. This way the BSCs need to know the CBC IP address, but not vice-versa.

The BSC can operate in either CBSP TCP server mode or CBSP TCP client mode.

The CBC related configuration of OsmoBSC can be found in the `cbc` configuration node of the VTY interface.

The default port number for the CBSP server is 48049, according to the CBSP specification. Hence it normally suffices to configure only the IP addresses for the remote CBC server or the local CBSP server:

Example: Configure CBSP TCP client to connect to CBC at 1.2.3.4:48049 in osmo-bsc.cfg

```
cbc
 mode client
 client
  remote-ip 1.2.3.4
```

In server mode, the default configuration is 127.0.0.1:48049, so it suffices to set `mode server` to accept CBSP connections from localhost:

```
cbc
 mode server
```

To also listen for inbound CBSP connections on all interfaces, both IPv4 and IPv6:

Example: Configure CBSP TCP server to listen on all interfaces in osmo-bsc.cfg

```
cbc
 mode server
 server
  local-ip ::
```


Should non-standard port numbers be required, these can be configured with the `client` / `local-port` or the `server` / `remote-port` settings.

The `client` config also supports an explicit local bind for connecting to the remote CBC, using `client` / `local-ip` and `local-port`.

IP addresses for client and server can remain configured at the same time, and the `mode` command can be used to switch between client and server operation. The `mode` command takes immediate effect, no restart of OsmoBSC is required. After changing `cbc` IP addresses in the telnet VTY, it is required to switch `mode` to `disabled` and back to `client` or `server` to take effect.

Example: Disable the CBSP link in the telnet VTY

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# cbc
OsmoBSC(config-cbc)# mode disabled
OsmoBSC(config-cbc)# end
```

For more details on the available configuration commands, please check the OsmoBSC VTY Reference.

MSC Pooling

MSC pooling is described in 3GPP TS 23.236 [3gpp-ts-23-236], and is supported by OsmoBSC since mid 2020.

The aim of MSC pooling is to distribute load from a BSC across multiple MSCs, which are equivalent and redundant infrastructure for the same core network.

The main mechanism for MSC pooling is the TMSI identity, which an MSC hands out to its attached subscribers. Typically 10 bits of the TMSI are designated as a Network Resource Identifier (NRI) that identifies the originating MSC, and allows OsmoBSC to direct a subscriber back to the same MSC instance that previously negotiated the IMSI Attach procedure. Typically, the full NRI value range available is divided into N even ranges, where each MSC is assigned one NRI range.

Subscribers attaching without a TMSI identity, or those with unknown NRI value, are evenly distributed across MSC instances. OsmoBSC uses a round-robin approach to distribute load across all connected MSCs.

A Paging Response from a subscriber is always returned back to whichever MSC initiated the Paging, regardless of the Mobile Identity used.

Finally, a NULL-NRI is a special NRI value that indicates that the MSC wishes to offload this subscriber to a different MSC. A NULL-NRI is an arbitrary NRI value that is chosen distinctly for each PLMN served by a BSC, so that a subscriber can be reassigned within that PLMN. Upon (periodic) Location Updating, an offloading MSC hands out a NULL-NRI value in the assigned TMSI, along with a non-broadcast LAI. The subscriber will notice the LAI mismatch, and immediately re-attempt the attach using the TMSI containing the NULL-NRI. The BSC recognises the NULL-NRI and redirects the subscriber to one of the other MSCs. A prerequisite for this to work well is that the particular MSC is previously marked as not accepting new subscribers, in the BSC's configuration.

The mechanisms described above make up the NAS node selection function implemented in the BSC.

3GPP TS 23.236 also defines that an offloading MSC hands subscriber information to the newly assigned MSC, which takes place outside the scope of the BSC.

Configuring MSC Pooling

The NRI ranges assigned to each MSC must match in the BSC and the MSC configuration. If MSC and BSC had inconsistent NRI value ranges configured, attached subscribers would be redirected MSC instances that did not perform the attach, possibly rendering the core network unusable.

Connecting Multiple MSCs

The `cs7 instance` configuration defines the SCCP addresses to reach the MSCs at. In addition, each MSC is configured by its own `msc` section in the configuration. An example osmo-bsc.cfg serving three MSCs:

```
cs7 instance 0
 # SCCP address book entries for the three MSCs
 sccp-address my-msc-0
  point-code 0.23.0
 sccp-address my-msc-1
  point-code 0.23.1
 sccp-address my-msc-2
  point-code 0.23.2

# assign each MSC configuration its remote SCCP address
msc 0
 msc-addr my-msc-0
msc 1
 msc-addr my-msc-1
msc 2
 msc-addr my-msc-2

# configure NRI value ranges
network
 nri bitlen 10
 nri null add 0
msc 0
 nri add 1 341
msc 1
 nri add 342 682
msc 2
 nri add 683 1023
```

NRI Value Bit Length

In OsmoBSC, the NRI value's bit length is freely configurable from 1 to 15 bits. 3GPP TS 23.236 suggests a typical bit length of 10, which is OsmoBSC's default. The NRI bit length must be identical across the entire MSC pool.

Change the NRI value bit length in OsmoBSC's VTY configuration like this:

```
network
 nri bitlen 10
```

In the TMSI bits, regardless of the NRI bit length, the NRI value always starts just after the most significant octet of a TMSI (most significant bit at TMSI's bit 23).

NULL-NRI

Since OsmoBSC supports serving only one PLMN, NULL-NRI are configured globally. Even though 3GPP TS 23.236 indicates that there is a single NULL-NRI per PLMN, OsmoBSC allows configuring multiple NULL-NRI values.

```
network
 nri null add 0
 nri null add 423
```


Assigning NRI Ranges to MSCs

Each MSC configured in OsmoBSC must be assigned a distinct NRI value range. Overlapping NRI value ranges will cause failure to serve subscribers.

NRI values are typically configured in ranges, here dividing a 10bit range (0..1023) into three equal ranges, while leaving 0 available to be configured as NULL-NRI:

```
msc 0
 nri add 1 341
msc 1
 nri add 342 684
msc 2
 nri add 685 1023
```

NRI can also be assigned in single values:

```
msc 0
 nri add 23
```

Ranges can be constructed arbitrarily by a sequence of `add` and `del` configurations, here a contrived example:

```
msc 0
 nri add 0 342
 nri del 23
 nri del 42 235
 nri add 1000 1023
```

To view the current NRI config in a running OsmoBSC instance, use the `show nri` command, here showing the result of the contrived example:

```
OsmoBSC(config-msc)# show nri
msc 0
 nri add 0 22
 nri add 24 41
 nri add 236 342
 nri add 1000 1023
```

On the VIEW and ENABLE VTY nodes, `show nri` shows all MSCs:

```
OsmoBSC> show nri
msc 0
 nri add 1 341
msc 1
 nri add 342 684
msc 2
 nri add 685 1023
```

When configuring overlapping NRI value ranges across MSCs, the telnet VTY warns about it, and starting OsmoBSC with such a configuration will fail:

```
msc 0
 nri add 1 511
msc 1
 nri add 512 1023
msc 2
 nri add 500 555
```

This results in:


```
$ osmo-bsc
DMSC ERROR msc 2: NRI range [500..555] overlaps between msc 2 and msc 0. For overlaps, msc ↩
    0 has higher priority than msc 2
DMSC ERROR msc 2: NRI range [500..555] overlaps between msc 2 and msc 1. For overlaps, msc ↩
    1 has higher priority than msc 2
```

MSC Offloading

To effectively offload a particular MSC, it must be marked as no longer taking new subscribers in OsmoBSC. This can be achieved in the telnet VTY by:

```
msc 0
 no allow-attach
```

This MSC will, as long as it is connected, continue to serve subscribers already attached to it: those that yield an NRI matching this MSC, and those that are being paged by this MSC. But OsmoBSC will no longer direct new subscribers to this MSC.

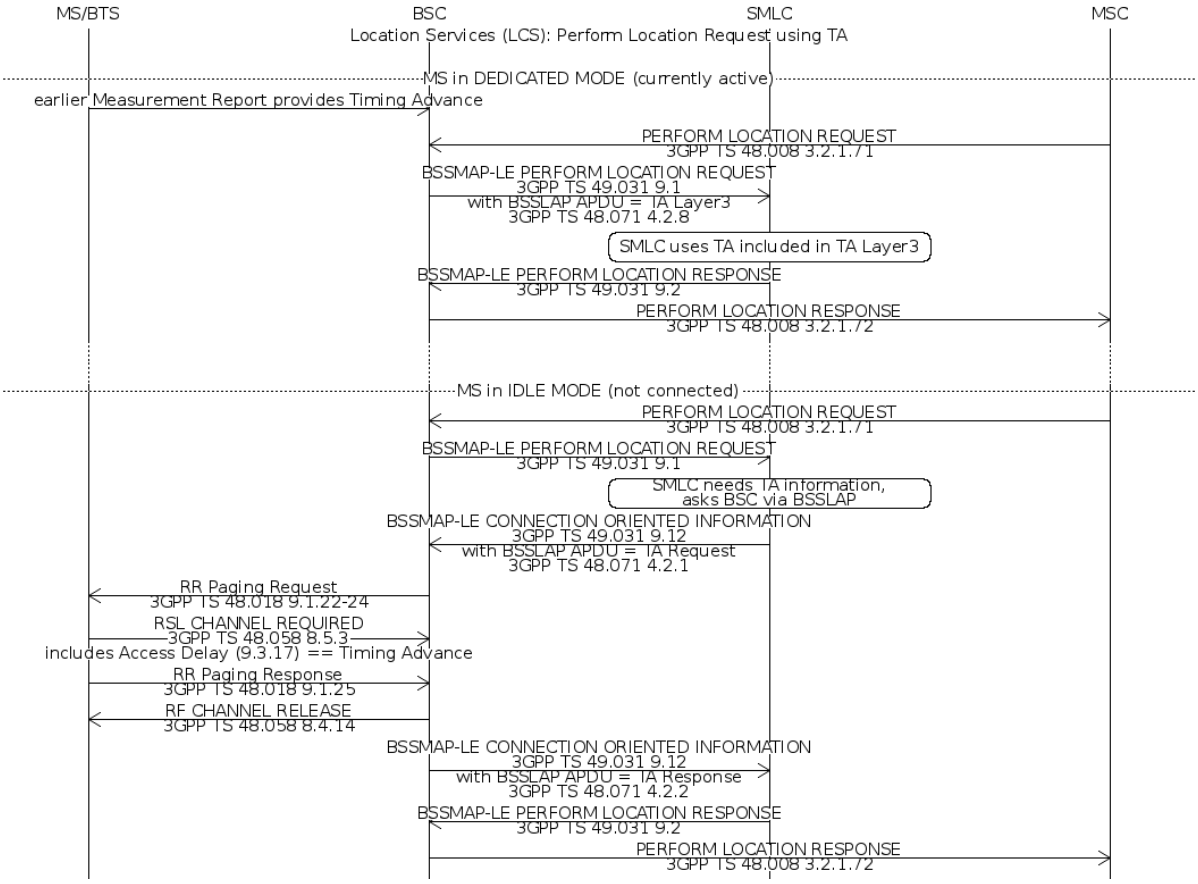
To re-enable an MSC for attaching new subscribers:

```
msc 0
 allow-attach
```

Location Services: Lb interface to SMLC

OsmoBSC and OsmoSMLC support positioning by Timing-Advance (TA), since October 2020.

A Perform Location Request is initiated by the MSC via BSSMAP on the A-interface, for a specific subscriber. The request is typically passed on via BSSMAP-LE on the Lb-interface to the SMLC. If required, the SMLC may request the subscriber's Timing Advance (a.k.a. Access Delay) from the BSC via BSSLAP (encapsulated BSSLAP APDU in a BSSMAP-LE Connection Oriented Information message). The SMLC may combine several location and velocity estimate methods to form a GAD PDU containing the resulting geographic location information. In TA-based positioning, the Timing-Advance information from the BSC is combined with the preconfigured latitude and longitude of the serving cell to form a location estimate. This is returned to the BSC via the Lb-interface, and in turn to the MSC via the A-interface.



Location Services (LCS) are described in 3GPP TS 43.059 [?]. Messages for LCS on the A-interface (BSSMAP, between BSC and MSC) are described in 3GPP TS 48.008 [3gpp-ts-48-008], on the Lb-interface (BSSMAP-LE between BSC and SMLC) in 3GPP TS 49.031 [?]. The resulting geographic location and possibly velocity information is encoded in GAD, described in 3GPP TS 23.032 [?].

Configure Lb-interface

All Lb-interface related configuration is found in the `smlc` section of OsmoBSC's configuration.

By default, the Lb-interface is disabled in OsmoBSC. It is started by `enable`.

```
smlc
 enable
```

On the Lb-interface, OsmoBSC always uses SSN "BSC (BSSMAP-LE)" (SSN code 250) and contacts the remote SMLC on SSN "SMLC (BSSMAP-LE)" (SSN code 252).

The point-codes are configurable, and default to OsmoBSC's local point-code 0.23.3 (187), and remote SMLC point-code 0.23.6 (190).

To configure a different remote SMLC point-code, first configure an arbitrarily named SCCP address in the `cs7` address book, and then apply that to the `smlc-addr` configuration:

```
cs7 instance 0
 sccp-address my-smlc
  point-code 0.42.6
smlc
 smlc-addr my-smlc
```

Similarly, OsmoBSC's local point-code on the Lb-interface is configured by the `bsc-addr` configuration:


```
cs7 instance 0
 sccp-address my-bsc-on-lb
  point-code 0.42.3
smlc
 bsc-addr my-bsc-on-lb
```

The geographic locations of individual cells are configured at the SMLC. See for example OsmoSMLC's user manual [?].

Osmocom Counters

The following gives an overview of all the types of counters available:

Osmo Counters (deprecated)

Osmo counters are the oldest type of counters added to Osmocom projects. They are not grouped.

- Printed as part of VTY show stats
- Increment, Decrement
- Accessible through the control interface: counter.<counter_name>

Rate Counters

Rate counters count rates of events.

- Printed as part of VTY show stats
- Intervals: per second, minute, hour, day or absolute value
- Increment only
- Accessible through the control interface
- Rate counters are grouped and different instances per group can exist

The control interface command to get a counter (group) is:

rate_ctr.per_{sec,min,hour,day,abs}.<group_name>.<idx>.[counter_name]

It is possible to get all counters in a group by omitting the counter name

Stat Item

Stat items are a grouped replacement for osmo counters.

- Printed as part of VTY show stats
- Replacement for osmo counters
- Not yet available through the control interface
- Grouped and indexed like rate counters
- Items have a unit
- Keeps a list of the last values measured, so could return an average, min, max, std. deviation. So far this is not implemented in any of the reporting options.

Statistic Levels

There are three levels on which a statistic can be aggregated in Osmocom projects: globally, per-peer and per-subscriber.

Global

These are global statistics.

Peer

These statistics relate to a peer the program connects to such as the NSVC in an SGSN.

This level also includes reporting global statistics.

Subscriber

These statistics are related to an individual mobile subscriber. An example would be bytes transferred in an SGSN PDP context.

This level also includes global and peer-based statistics.

Stats Reporter

The stats reporter periodically collects osmo counter, rate counter and stat item values and sends them to a backend. Currently implemented are outputting to the configured log targets and a statsd connector.

Configuring a stats reporter

Periodically printing the statistics to the log can be done in the following way:

Example 16.1 Log statistics

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# stats interval 60 ❶
OsmoBSC(config)# stats reporter log ❷
OsmoBSC(config-stats)# level global ❸
OsmoBSC(config-stats)# enable ❹
```

❶ The interval determines how often the statistics are reported.

❷ Write the statistic information to any configured log target.

❸ Report only `global` statistics (can be `global`, `peer`, or `subscriber`).

❹ Enable the reporter, `disable` will disable it again.

The counter values can also be sent to any aggregation/visualization tool that understands the statsd format, for example a statsd server with graphite or prometheus using the statsd_exporter together with grafana.

The statsd format is specified in https://github.com/b/statsd_spec

Example 16.2 Report statistics to statsd

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# stats interval 10
OsmoBSC(config)# stats reporter statsd ❶
OsmoBSC(config-stats)# prefix BSC1 ❷
OsmoBSC(config-stats)# level subscriber ❸
OsmoBSC(config-stats)# remote-ip 1.2.3.4 ❹
OsmoBSC(config-stats)# remote-port 8125 ❺
OsmoBSC(config-stats)# enable
```

❶ Configure the statsd reporter.

❷ Prefix the reported statistics. This is useful to distinguish statistics from multiple instances of the same service.

❸ Report only `global` statistics or include `peer` or `subscriber` statistics as well.

❹ IP address of the statsd server.

❺ UDP port of the statsd server. Statsd by default listens to port 8125.

Setting up a statsd server and configuring the visualization is beyond the scope of this document.

Implemented Counters

These counters and their description based on OsmoBSC 1.4.0.84-3f1f8 (OsmoBSC).

Rate Counters

Table 7: bts - base transceiver station

Name	Reference	Description
chreq:total	[?]	Received channel requests.
chreq:no_channel	[?]	Sent to MS no channel available.
chan:rf_fail	[?]	Received a RF failure indication from BTS.
chan:rll_err	[?]	Received a RLL failure with T200 cause from BTS.
oml_fail	[?]	Received a TEI down on a OML link.
rsl_fail	[?]	Received a TEI down on a OML link.
codec:amr_f	[?]	Count the usage of AMR/F codec by channel mode requested.
codec:amr_h	[?]	Count the usage of AMR/H codec by channel mode requested.
codec:efr	[?]	Count the usage of EFR codec by channel mode requested.
codec:fr	[?]	Count the usage of FR codec by channel mode requested.
codec:hr	[?]	Count the usage of HR codec by channel mode requested.
paging:attempted	[?]	Paging attempts for a subscriber.
paging:already	[?]	Paging attempts ignored as subscriber was already being paged.

Table 7: (continued)

Name	Reference	Description
paging:responded	[?]	Paging attempts with successful paging response.
paging:expired	[?]	Paging Request expired because of timeout T3113.
chan_act:total	[?]	Total number of Channel Activations.
chan_act:nack	[?]	Number of Channel Activations that the BTS NACKed
rsl:unknown	[?]	Number of unknown/unsupported RSL messages received from BTS
rsl:ipa_nack	[?]	Number of IPA (RTP/dyn-PDCH) related NACKs received from BTS
chan:mode_modify_nack	[?]	Number of Channel Mode Modify NACKs received from BTS

Table 8: bts - base transceiver station

Name	Reference	Description
chreq:total	[?]	Received channel requests.
chreq:no_channel	[?]	Sent to MS no channel available.
chan:rf_fail	[?]	Received a RF failure indication from BTS.
chan:rll_err	[?]	Received a RLL failure with T200 cause from BTS.
oml_fail	[?]	Received a TEI down on a OML link.
rsl_fail	[?]	Received a TEI down on a OML link.
codec:amr_f	[?]	Count the usage of AMR/F codec by channel mode requested.
codec:amr_h	[?]	Count the usage of AMR/H codec by channel mode requested.
codec:efr	[?]	Count the usage of EFR codec by channel mode requested.
codec:fr	[?]	Count the usage of FR codec by channel mode requested.
codec:hr	[?]	Count the usage of HR codec by channel mode requested.
paging:attempted	[?]	Paging attempts for a subscriber.
paging:already	[?]	Paging attempts ignored as subscriber was already being paged.
paging:responded	[?]	Paging attempts with successful paging response.
paging:expired	[?]	Paging Request expired because of timeout T3113.
chan_act:total	[?]	Total number of Channel Activations.
chan_act:nack	[?]	Number of Channel Activations that the BTS NACKed
rsl:unknown	[?]	Number of unknown/unsupported RSL messages received from BTS
rsl:ipa_nack	[?]	Number of IPA (RTP/dyn-PDCH) related NACKs received from BTS
chan:mode_modify_nack	[?]	Number of Channel Mode Modify NACKs received from BTS

Table 9: bts - base transceiver station

Name	Reference	Description
chreq:total	[?]	Received channel requests.
chreq:no_channel	[?]	Sent to MS no channel available.
chan:rf_fail	[?]	Received a RF failure indication from BTS.
chan:rll_err	[?]	Received a RLL failure with T200 cause from BTS.
oml_fail	[?]	Received a TEI down on a OML link.
rsl_fail	[?]	Received a TEI down on a OML link.
codec:amr_f	[?]	Count the usage of AMR/F codec by channel mode requested.
codec:amr_h	[?]	Count the usage of AMR/H codec by channel mode requested.
codec:efr	[?]	Count the usage of EFR codec by channel mode requested.
codec:fr	[?]	Count the usage of FR codec by channel mode requested.
codec:hr	[?]	Count the usage of HR codec by channel mode requested.
paging:attempted	[?]	Paging attempts for a subscriber.
paging:already	[?]	Paging attempts ignored as subscriber was already being paged.
paging:responded	[?]	Paging attempts with successful paging response.
paging:expired	[?]	Paging Request expired because of timeout T3113.
chan_act:total	[?]	Total number of Channel Activations.
chan_act:nack	[?]	Number of Channel Activations that the BTS NACKed
rsl:unknown	[?]	Number of unknown/unsupported RSL messages received from BTS
rsl:ipa_nack	[?]	Number of IPA (RTP/dyn-PDCH) related NACKs received from BTS
chan:mode_modify_nack	[?]	Number of Channel Mode Modify NACKs received from BTS

Table 10: e1inp - E1 Input subsystem

Name	Reference	Description
hdlc:abort	[?]	HDLC abort
hdlc:bad_fcs	[?]	HLDC Bad FCS
hdlc:overrun	[?]	HDLC Overrun
alarm	[?]	Alarm
removed	[?]	Line removed

Table 11: bsc - base station controller

Name	Reference	Description
assignment:attempted	[?]	Assignment attempts.
assignment:completed	[?]	Assignment completed.
assignment:stopped	[?]	Connection ended during Assignment.
assignment:no_channel	[?]	Failure to allocate lchan for Assignment.
assignment:timeout	[?]	Assignment timed out.
assignment:failed	[?]	Received Assignment Failure message.
assignment:error	[?]	Assignment failed for other reason.
handover:attempted	[?]	Intra-BSC handover attempts.
handover:completed	[?]	Intra-BSC handover completed.
handover:stopped	[?]	Connection ended during HO.
handover:no_channel	[?]	Failure to allocate lchan for HO.
handover:timeout	[?]	Handover timed out.
handover:failed	[?]	Received Handover Fail messages.
handover:error	[?]	Re-assignment failed for other reason.
interbsc_ho_out:attempted	[?]	Attempts to handover to remote BSS.
interbsc_ho_out:completed	[?]	Handover to remote BSS completed.
interbsc_ho_out:stopped	[?]	Connection ended during HO.
interbsc_ho_out:timeout	[?]	Handover timed out.
interbsc_ho_out:error	[?]	Handover to remote BSS failed for other reason.
interbsc_ho_in:attempted	[?]	Attempts to handover from remote BSS.
interbsc_ho_in:completed	[?]	Handover from remote BSS completed.
interbsc_ho_in:stopped	[?]	Connection ended during HO.
interbsc_ho_in:no_channel	[?]	Failure to allocate lchan for HO.
interbsc_ho_in:failed	[?]	Received Handover Fail message.
interbsc_ho_in:timeout	[?]	Handover from remote BSS timed out.
interbsc_ho_in:error	[?]	Handover from remote BSS failed for other reason.
paging:attempted	[?]	Paging attempts for a subscriber.
paging:detached	[?]	Paging request send failures because no responsible BTS was found.
paging:responded	[?]	Paging attempts with successful response.
abis:unknown_unit_id	[?]	Connection attempts from unknown IPA CCM Unit ID.

Osmo Stat Items

base transceiver station .bts - base transceiver station

Name	Reference	Description	Unit
chanloadavg	[?]	Channel load average.	%
T3122	[?]	T3122 IMMEDIATE ASSIGNMENT REJECT wait indicator.	s
rach_busy	[?]	RACH slots with signal above threshold	%

Name	Reference	Description	Unit
rach_access	[?]	RACH slots with access bursts in them	%

base transceiver station .bts - base transceiver station

Name	Reference	Description	Unit
chanloadavg	[?]	Channel load average.	%
T3122	[?]	T3122 IMMEDIATE ASSIGNMENT REJECT wait indicator.	s
rach_busy	[?]	RACH slots with signal above threshold	%
rach_access	[?]	RACH slots with access bursts in them	%

base transceiver station .bts - base transceiver station

Name	Reference	Description	Unit
chanloadavg	[?]	Channel load average.	%
T3122	[?]	T3122 IMMEDIATE ASSIGNMENT REJECT wait indicator.	s
rach_busy	[?]	RACH slots with signal above threshold	%
rach_access	[?]	RACH slots with access bursts in them	%

Osmo Counters

Abis/IP Interface

A-bis Operation & Maintenance Link

The GSM Operation & Maintenance Link (OML) is specified in 3GPP TS 12.21 and is used between a GSM Base-Transceiver-Station (BTS) and a GSM Base-Station-Controller (BSC). The default TCP port for OML is *3002*. The connection will be opened from the BTS to the BSC.

Abis OML is only specified over E1 interfaces. The Abis/IP implementation of OsmoBTS and OsmoBSC extend and/or deviate from the TS 12.21 specification in several ways. Please see the *OsmoBTS Abis Protocol Specification* [osmobts-abis-spec] for more information.

A-bis Radio Signalling Link

The GSM Radio Signalling Link (RSL) is specified in 3GPP TS 08.58 and is used between a GSM Base-Transceiver-Station and a GSM Base-Station-Controller (BSC). The default TCP port for RSL is *3003*. The connection will be opened from the BTS to BSC after it has been instructed by the BSC.

Abis RSL is only specified over E1 interfaces. The Abis/IP implementation of OsmoBTS and OsmoBSC extend and/or deviate from the TS 08.58 specification in several ways. Please see the *OsmoBTS Abis Protocol Specification* [osmobts-abis-spec] for more information.

Locate Abis/IP based BTS

We can use a tool called abisip-find to be able to find BTS which is connected in the network. This tool is located in the OsmoBSC project repository under: *./src/ipaccess*

abisip-find

abisip-find is a small command line tool which is used to search and find BTS devices in your network (e.g. sysmoBTS, nanoBTS).

It uses broadcast packets of the UDP variant of the Abis-IP protocol on port 3006, and thus will find any BTS that can be reached by the all-network broadcast address 255.255.255.255

When program is started it will print one line for each BTS it can find.

Example: using abisip-find to find BTS in your network

```
$ ./abisip-find
abisip-find (C) 2009 by Harald Welte
This is FREE SOFTWARE with ABSOLUTELY NO WARRANTY

you might need to specify the outgoing
network interface, e.g. ``abisip-find eth0``
Trying to find ip.access BTS by broadcast UDP...

MAC_Address='24:62:78:01:02:03' IP_Address='192.168.0.171' Serial_Number='123'
Unit_ID='sysmoBTS 1002'

MAC_Address='24:62:78:04:05:06' IP_Address='192.168.0.182' Serial_Number='456'
Unit_ID='sysmoBTS 1002'

MAC Address='00:01:02:03:04:05' IP Address='192.168.100.123' Unit ID='65535/0/0'
Location_1='' Location 2='BTS_NBT131G'  Equipment Version='165a029_55'
Software Version='168a302_v142b13d0' Unit Name='nbts-00-02-95-00-4E-B3'
Serial Number='00123456'

^C
```

You may have to start the program as a root:

```
$ sudo ./abisip-find eth0
```

Deploying a new nanoBTS

A tool called ipaccess-config can be used to configure a new ip.access nanoBTS.

ipaccess-config

This program is very helpful tool which is used to configure Unit ID and Primary OML IP. You can find this tool in the OsmoBSC repository under: *./src/ipaccess*

Example: using ipaccess-config to configure Unit ID and Primary OML IP of nanoBTS

```
$ ./ipaccess-config -u 1801/0/0❶  10.9.1.195❷ -o 10.9.1.154❸

ipaccess-config (C) 2009-2010 by Harald Welte and others
This is FREE SOFTWARE with ABSOLUTELY NO WARRANTY

Trying to connect to ip.access BTS ...
```



```
abis_nm.c:316 OC=SITE-MANAGER(00) INST=(ff,ff,ff) STATE CHG:
OP_STATE=Disabled AVAIL=Not installed(07)
abis_nm.c:316 OC=BTS(01) INST=(00,ff,ff) STATE CHG:
OP_STATE=Disabled AVAIL=Not installed(07) ADM=Locked
abis_nm.c:316 OC=BASEBAND-TRANSCEIVER(04) INST=(00,00,ff) STATE CHG:
OP_STATE=Disabled AVAIL=Not installed(07) ADM=Locked
OML link established using TRX 0
setting Unit ID to '1801/0/0'
setting primary OML link IP to '10.9.1.154'
abis_nm.c:316 OC=CHANNEL(03) INST=(00,00,00) STATE CHG:
OP_STATE=Disabled AVAIL=Not installed(07) ADM=Locked
...
abis_nm.c:2433 OC=BASEBAND-TRANSCEIVER(04) INST=(00,00,ff) IPACCESS(0xf0):
SET NVATTR ACK
Set the NV Attributes.
```

❶ Unit ID

❷ IP address of the NITB

❸ IP address of the nanoBTS

Osmocom Control Interface

The VTY interface as described in Section 6 is aimed at human interaction with the respective Osmocom program.

Other programs **should not** use the VTY interface to interact with the Osmocom software, as parsing the textual representation is cumbersome, inefficient, and will break every time the formatting is changed by the Osmocom developers.

Instead, the *Control Interface* was introduced as a programmatic interface that can be used to interact with the respective program.

Control Interface Protocol

The control interface protocol is a mixture of binary framing with text based payload.

The protocol for the control interface is wrapped inside the IPA multiplex header with the stream identifier set to IPAC_PROTO_OSMO (0xEE).

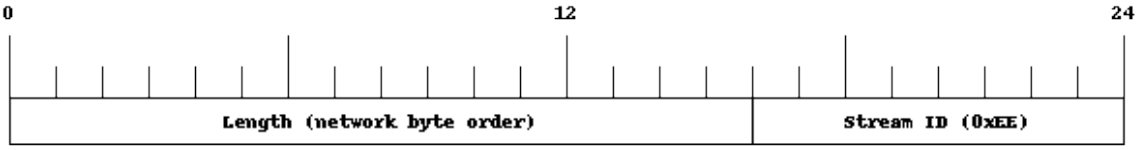


Figure 6: IPA header for control protocol

Inside the IPA header is a single byte of extension header with protocol ID 0x00 which indicates the control interface.

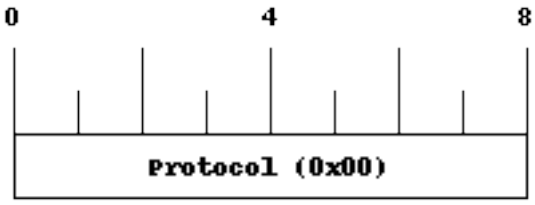


Figure 7: IPA extension header for control protocol

After the concatenation of the two above headers, the plain-text payload message starts. The format of that plain text is illustrated for each operation in the respective message sequence chart in the chapters below.

The fields specified below follow the following meaning:

<id>
A numeric identifier, uniquely identifying this particular operation. Value `0` is not allowed unless it's a TRAP message. It will be echoed back in any response to a particular request.

<var>
The name of the variable / field affected by the GET / SET / TRAP operation. Which variables/fields are available is dependent on the specific application under control.

<val>
The value of the variable / field

<reason>
A text formatted, human-readable reason why the operation resulted in an error.

GET operation

The GET operation is performed by an external application to get a certain value from inside the Osmocom application.

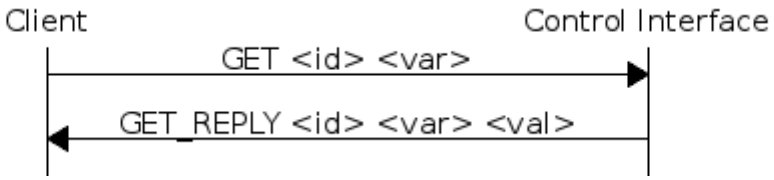


Figure 8: Control Interface GET operation (successful outcome)

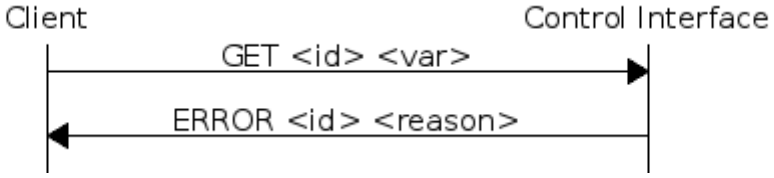


Figure 9: Control Interface GET operation (unsuccessful outcome)

SET operation

The SET operation is performed by an external application to set a value inside the Osmocom application.

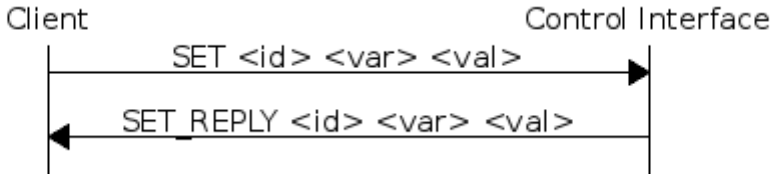


Figure 10: Control Interface SET operation (successful outcome)

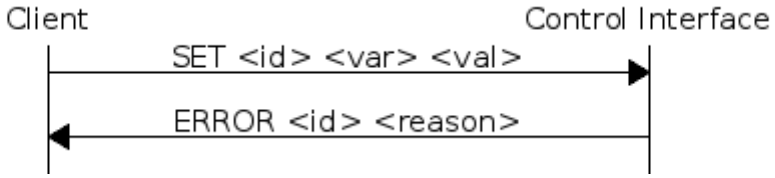


Figure 11: Control Interface SET operation (unsuccessful outcome)

TRAP operation

The program can at any time issue a trap. The term is used in the spirit of SNMP.

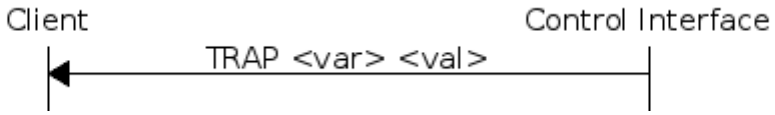


Figure 12: Control Interface TRAP operation

Common variables

There are several variables which are common to all the programs using control interface. They are described in the following table.

Table 12: Variables available over control interface

Name	Access	Value	Comment
counter.*	RO		Get counter value.
rate_ctr.*	RO		Get list of rate counter groups.
rate_ctr.IN.GN.GI.name	RO		Get value for interval IN of rate counter name which belong to group named GN with index GI.

Those read-only variables allow to get value of arbitrary counter using its name.

For example "rate_ctr.per_hour.bsc.0.handover:timeout" is the number of handover timeouts per hour.

Of course for that to work the program in question have to register corresponding counter names and groups using libosmocore functions.

In the example above, "bsc" is the rate counter group name and "0" is its index. It is possible to obtain all the rate counters in a given group by requesting "rate_ctr.per_sec.bsc.*" variable.

The list of available groups can be obtained by requesting "rate_ctr.*" variable.

The rate counter group name have to be prefixed with interval specification which can be any of "**per_sec**", "**per_min**", "**per_hour**", "**per_day**" or "**abs**" for absolute value.

The old-style counters available via "counter.*" variables are superseded by "rate_ctr.abs" so its use is discouraged. There might still be some applications not yet converted to rate_ctr.

Control Interface python examples

In the osmo-python-tests repository, there is an example python script called scripts/osmo_ctrl.py which implements the Osmocom control interface protocol.

You can use this tool either stand-alone to perform control interface operations against an Osmocom program, or you can use it as a reference for developing your own python software talking to the control interface.

Another implementation is in scripts/osmo_rate_ctr2csv.py which will retrieve performance counters for a given Osmocom program and output it in csv format. This can be used to periodically (using systemd timer for example) retrieve data to build KPI and evaluate how it changes over time.

Internally it uses "rate_ctr.*" variable described in [?] to get the list of counter groups and than request all the counters in each group. Applications interested in individual metrics can request it directly using rate_ctr2csv.py as an example.

Getting rate counters

Example: Use rate_ctr2csv.py to get rate counters from OsmoBSC

```
$ ./scripts/osmo_rate_ctr2csv.py --header
Connecting to localhost:4249...
Getting rate counter groups info...
"group","counter","absolute","second","minute","hour","day"
"e1inp.0","hdlc:abort","0","0","0","0","0"
"e1inp.0","hdlc:bad_fcs","0","0","0","0","0"
"e1inp.0","hdlc:overrun","0","0","0","0","0"
"e1inp.0","alarm","0","0","0","0","0"
"e1inp.0","removed","0","0","0","0","0"
"bsc.0","chreq:total","0","0","0","0","0"
"bsc.0","chreq:no_channel","0","0","0","0","0"
...
"msc.0","call:active","0","0","0","0","0"
"msc.0","call:complete","0","0","0","0","0"
"msc.0","call:incomplete","0","0","0","0","0"
Completed: 44 counters from 3 groups received.
```

Setting a value

Example: Use osmo_ctrl.py to set the short network name of OsmoBSC

```
$ ./osmo_ctrl.py -d localhost -s short-name 32C3
Got message: SET_REPLY 1 short-name 32C3
```


Getting a value

Example: Use osmo_ctrl.py to get the mnc of OsmoBSC

```
$ ./osmo_ctrl.py -d localhost -g mnc
Got message: GET_REPLY 1 mnc 262
```

Listening for traps

You can use `osmo_ctrl.py` to listen for traps the following way:

Example: Using osmo_ctrl.py to listen for traps:

```
$ ./osmo_ctrl.py -d localhost -m
❶
```

❶ the command will not return and wait for any TRAP messages to arrive

Control interface

The actual protocol is described in Section 21, the variables common to all programs using it are described in Section 21.2. Here we describe variables specific to OsmoBSC. The commands starting with prefix "bts.N." are specific to a certain BTS so N have to be replaced with BTS number when issuing command e. g. "bts.1.channel-load". Similarly the TRX-specific commands are additionally prefixed with TRX number e. g. "bts.1.trx.2.arfcn".

Table 13: Variables available over control interface

Name	Access	Trap	Value	Comment
msc_connection_status	RO	Yes	"connected", "disconnected"	Indicate the status of connection to MSC.
bts_connection_status	RO	Yes	"connected", "disconnected"	Indicate the status of connection to BTS.
location	RW	Yes	"<unixtime>,(invalid|fix2d|fix3d),<lat>,<lon>,<height>"	Set/Get location data.
timezone	RW	No	"<hours>,<mins>,<dst>", "off"	-19 <= hours <= 19, mins in {0, 15, 30, 45}, and 0 <= dst <= 2
apply-configuration	WO	No	"restart"	Restart all BTSes.
mnc	RW	No	"<mnc>"	Set/Get MNC (value between (0, 999)).
mcc	RW	No	"<mcc>"	Set/Get MCC (value between (1, 999)).
short-name	RW	No	"<name>"	Set/Get network's short name.
long-name	RW	No	"<name>"	Set/Get network's long name.
mcc-mnc-apply	WO	No	"<mcc>,<mnc>"	Apply new MCC/MNC values if different from currently used one.
notification	WO	Yes	Arbitrary value	See Section 22.1 for details.
inform-msc-v1	WO	Yes	Arbitrary value	See Section 22.2 for details.

Table 13: (continued)

Name	Access	Trap	Value	Comment
rf_locked	RW	No	"0","1"	See Section 22.6 for details.
number-of-bts	RO	No	"<num>"	Get number of configured BTS.
bts.N.location-area-code	RW	No	"<lac>"	Set/Get LAC (value between (0, 65535)).
bts.N.cell-identity	RW	No	"<id>"	Set/Get Cell Identity (value between (0, 65535)).
bts.N.apply-configuration	WO	No	Ignored	Restart BTS via OML.
bts.N.send-new-system-informations	WO	No	Ignored	Regenerate System Information messages for given BTS.
bts.N.channel-load	RO	No	"<name>,<used>,<total>"	See Section 22.3 for details.
bts.N.oml-connection-state	RO	No	"connected", "disconnected", "degraded"	Indicate the status of OML connection of BTS.
bts.N.oml-uptime	RO	No	<uptime>	Return OML link uptime in seconds.
bts.N.gprs-mode	RW	No	"<mode>"	See Section 22.4 for details.
bts.N.rf_state	RO	No	"<oper>,<admin>,<pol>"	See Section 22.5 for details.
bts.N.trx.M.arfcn	RW	No	"<arfcn>"	Set/Get ARFCN (value between (0, 1023)).
bts.N.trx.M.max-power-reduction	RW	No	"<mpr>"	See Section 22.7 for details.

notification

Setting this variable initiate TRAP "notification" to all the clients connected to control interface with the value supplied in SET operation. This is not intended to be used outside of local systems.

inform-msc-v1

Setting this variable initiate TRAP "inform-msc-v1" to all connected MSCs control interfaces with the value supplied in SET operation.

channel-load

Obtain channel load for given BTS. Returns concatenated set of triplets ("<name>,<used>,<total>") for all channel types configured on the BTS. The "<name>" is the channel type. The "<used>" is the number of channels of that type currently in use. The "<total>" is the number of channels of that type configured on the BTS.

gprs-mode

Set/Get the GPRS mode of the BTS. One of the following is accepted/returned: "none", "gprs", "egprs".

rf_state

Following triplet is returned: "<oper>,<admin>,<pol>". The "<oper>" might be "operational" or "inoperational" representing different operational states. The "<admin>" might be "locked" or "unlocked" representing administrative status. The "<pol>" might be "off", "on", "grace" or "unknown" representing different RF policies.

rf_locked

Set/Get RF locked status. The GET operation will return either "0" or "1" depending on the RF lock status. The SET operation will set RF lock status if RF Ctrl is enabled in the BSC Configuration.

max-power-reduction

Set/Get the value of maximum power reduction. Even values between 0 and 22 are accepted.

FIXME: add variables defined in src/ctrl/control_if.c?

Osmux

`Osmux` is a protocol aimed at multiplexing and transmitting voice and signalling traffic from multiple sources in order to reduce the overall bandwidth consumption. This feature becomes specially meaningful in case of satellite based GSM systems, where the transmission cost on the back-haul is relatively expensive. In such environment, even seemingly small protocol optimizations, eg. message batching and trunking, can result in significant cost reduction.

Full reference document for the osmux protocol can be found here: http://ftp.osmocom.org/docs/latest/osmux-reference.pdf

In Osmocom satellite based GSM networks, the satellite link is envisioned to be in between the BSS and the core network, that is, between the BSC and the MSC (or BSC-NAT). Hence, Osmocom components can make use of `Osmux` protocol to multiplex payload audio streams from call legs between OsmoBSC and OsmoMSC (or OsmoBSCNAT). The MGW attached those components need of course also be aware of `Osmux` existence in order to properly set up the audio data plane.

Osmux and NAT

It is quite usual for satellite based links to use NATs, which means any or both of the two components at each side of the satellite link (BSC and MSC/BSC-NAT) may end up being behind a NAT and being unable to provide the real public address to its peer on the other side of the satellite.

As a result, upon call parameter negotiation (RTP/Osmux IP address and port), those parameters won't be entirely useful and some specific logic needs to be introduced into the network components to circumvent the NAT under those cases.

For instance, if the BSC and its co-located MGW (BSC/MGW from now on) is under a NAT, it may end up providing its private address and port as RTP/Osmux parameters to the MSC/MGW through GSM protocols, but MSC will fail to send any message to that tuple because of the NAT or routing issues (due to IP address being a private address). In that scenario, MSC/MGW needs to be aware that there's a NAT and wait until an RTP/Osmux message arrives from the BSC/MGW host. It then can, from that message source IP address and port (and CID in case of Osmux), discover the real public IP address and port of the peer (BSC/MGW). From that point on, the BSC/MGW punched a hole in the NAT (its connection table is updated) and MSC/MGW is able to send data back to it on that connection.

Moreover, NATs tend to drop connections from their connection tables after some inactivity time, meaning a peer may receive notice about the other end not being available while it actually is. This means the GSM network needs to be configured in a way to ensure inactivity periods are short enough that this cannot occur. That's the reason why OsmoMGW provides the `osmux dummy` VTY command to enable sending dummy packets from time to time to keep the connections alive.

CID allocation

Each peer (BSC/MGW and MSC/MGW) allocates its own *recvCID*, and receives from the peer through the used GSM protocol the peer's *recvCID*, which becomes the local *sendCID* for that connection.

```
BSC/MGW(recvCID=Y,sendCID=?)<-X--MSC/MGW(recvCID=X,sendCID=?)
BSC/MGW(recvCID=Y,sendCID=X)--Y->MSC/MGW(recvCID=X,sendCID=Y)
```

This way each peer is responsible for allocating and managing their own local address (CID) space. This is basically the same that happens with regular IP address and port in the RTP case (and those also apply when Osmux is used, but an extra identifier, the CID, is allocated).

In an ideal scenario, without NAT, each BSC/MGW would have a public, differentiated and unique IP and port set tuple, And MSC/MGW should be able to identify messages coming from them by easily matching source IP address, port (and CID in Osmux case) against the parameters negotiated during call set up.

In this kind of scenario, MSC/MGW could easily open and manage one Osmux socket per BSC (based on SDP IPaddr and port parameters), with same `<localIPaddr, localPort>` tuple, allowing for 256 Osmux CIDs per BSC and hence call legs per BSC. Each of the peers could actually have more than one Osmux socket towards the other peer, by using a pool of ports or IP addresses, so there's really not limit if required as long as there's a way to infer the initially negotiated `<srcIP, srcPort, dstIP, dstPort, sendCID>` tuple from the received audio packets.

However, due to some constrains from in between NATs explained in section above, BSC/MGW IP address and port are not a priory known, and could change between different connections coming from it. As a result, it is difficult to infer the entire tuple, so for now MGW needs to allocate its Osmux *recvCID* in a clever way, in order to be able to identify the full tuple from it.

Hence, currently OsmoMGW CID allocation implementation shares CID between all connections, which means it can only handle up to 256 concurrent Osmux connections (call legs).

Future work in OsmoMGW (OS#4092) plans to use a set of local ports for Osmux sockets instead of only 1 currently used. This way local ports can be matched against specific `<remoteIP, remotePort>` tuples and have an entire 256 Osmux CID space per `<remoteIP, remotePort>` (aka per peer).

3GPP AoIP network setup with Osmux

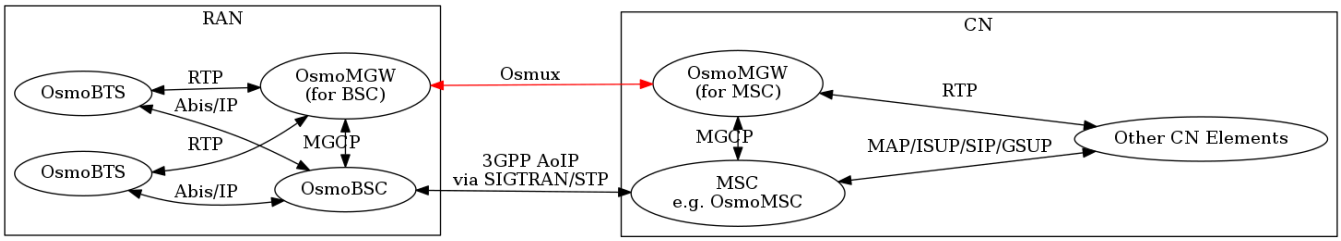


Figure 13: Sample node diagram of a 3GPP AoIP network with Osmux enabled

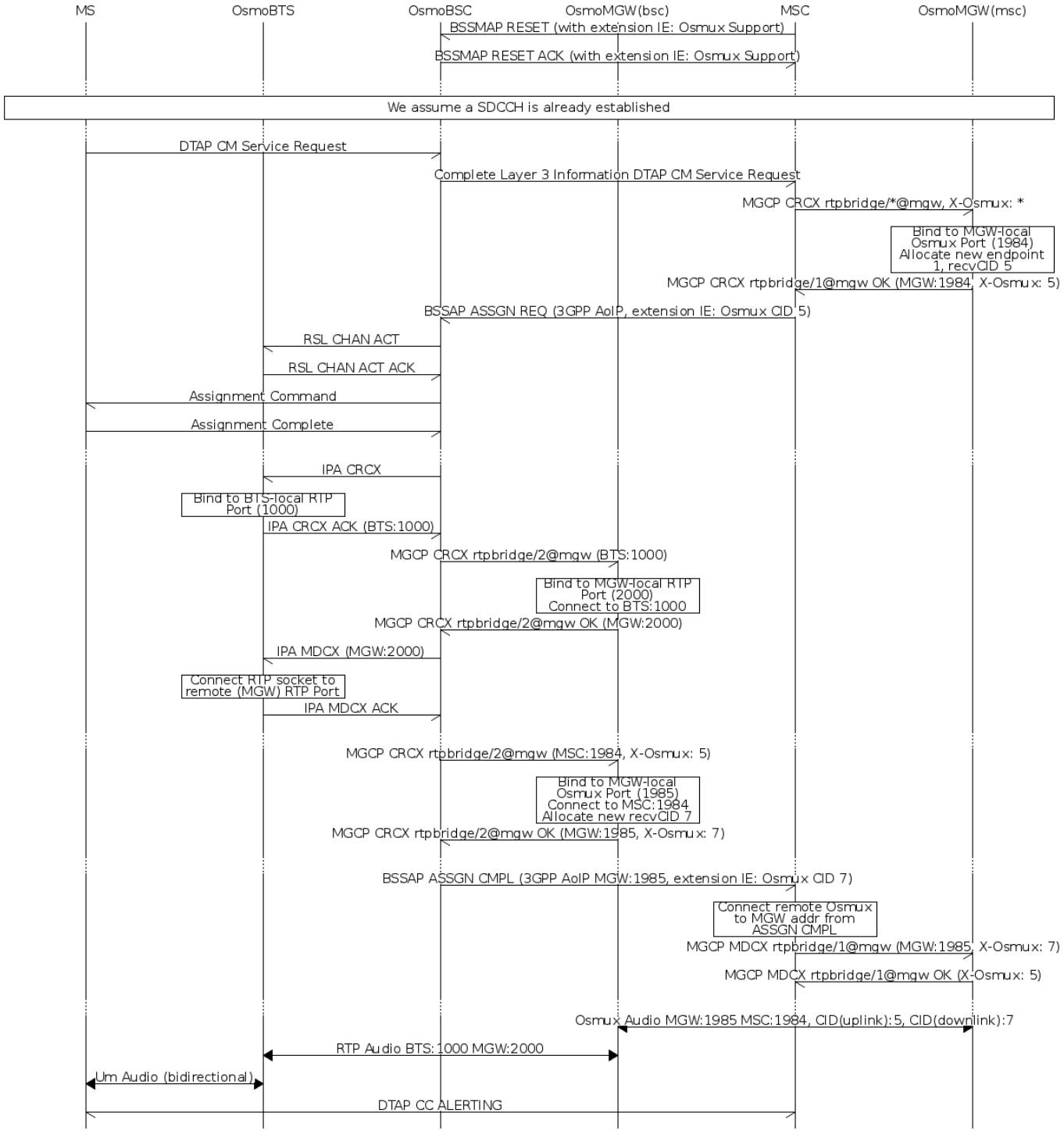


Figure 14: MO-call with Osmux enable on 3GPP AoIP

SCCPLite network setup with Osmux

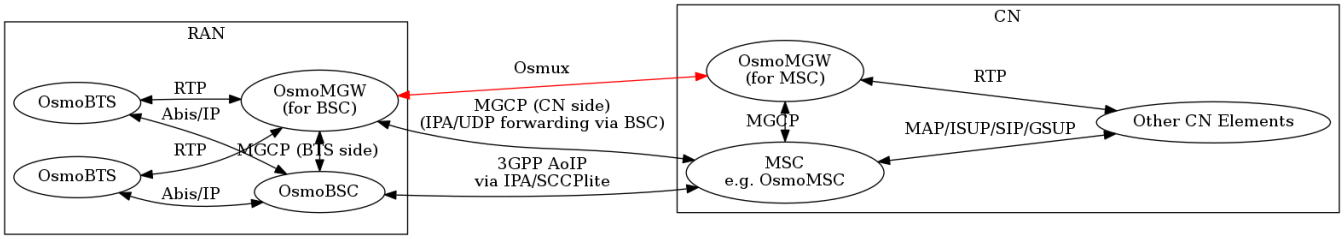


Figure 15: Sample node diagram of a 3GPP AoIP using A/IP with IPA/SCCPlite network with Osmux enabled

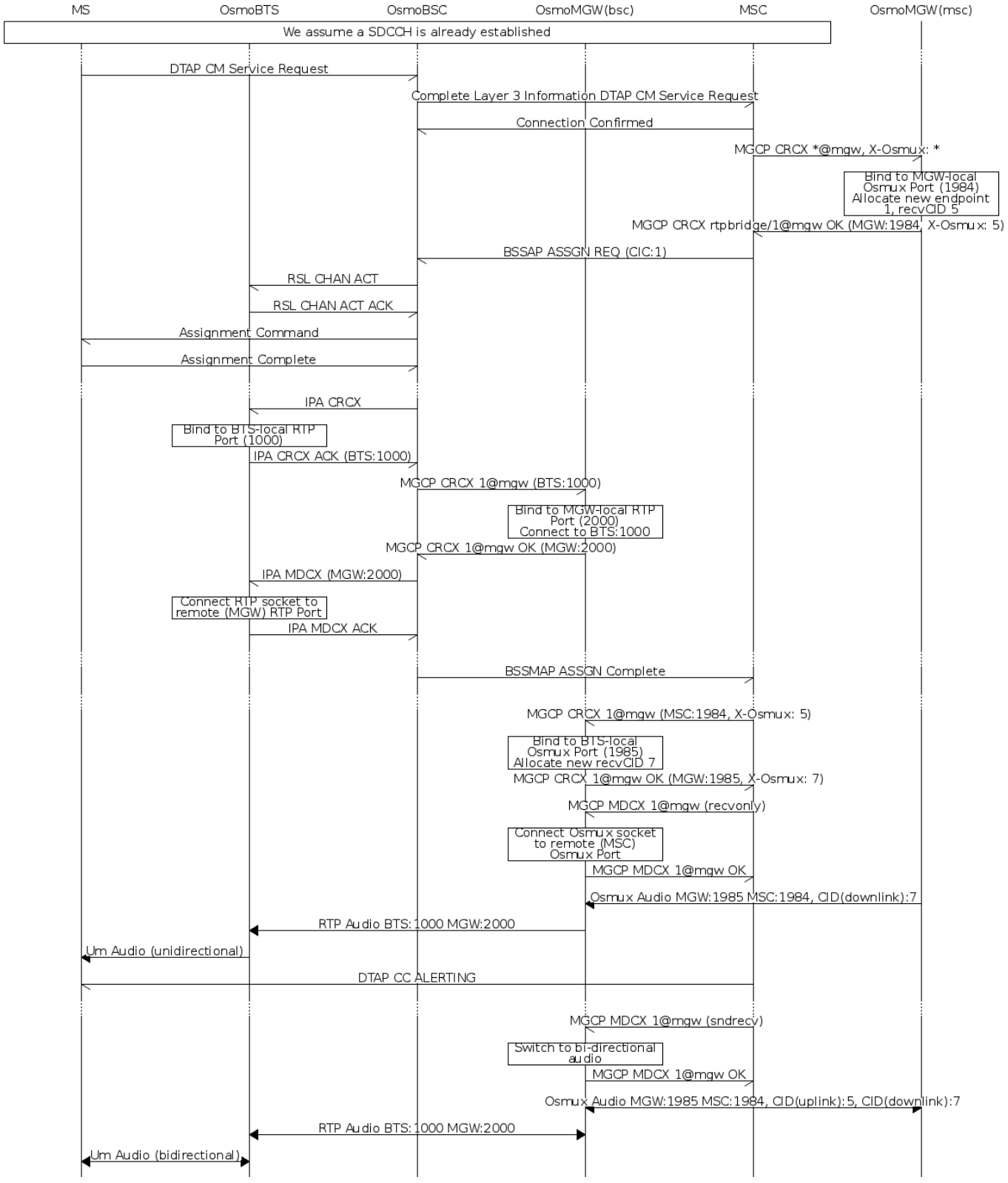


Figure 16: MO-call with Osmux enable on 3GPP AoIP using A/IP with IPA/SCCPlite

SCCPLite network setup with Osmux + BSC-NAT

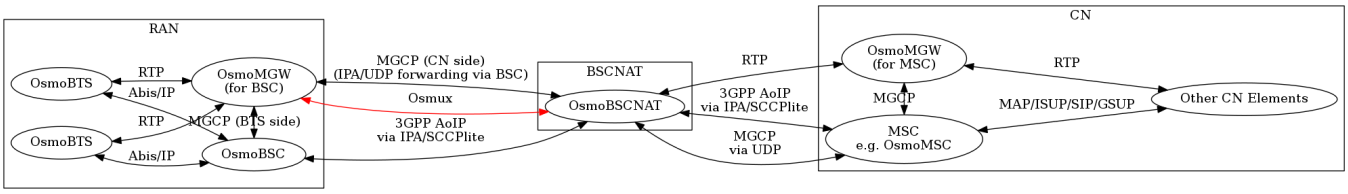
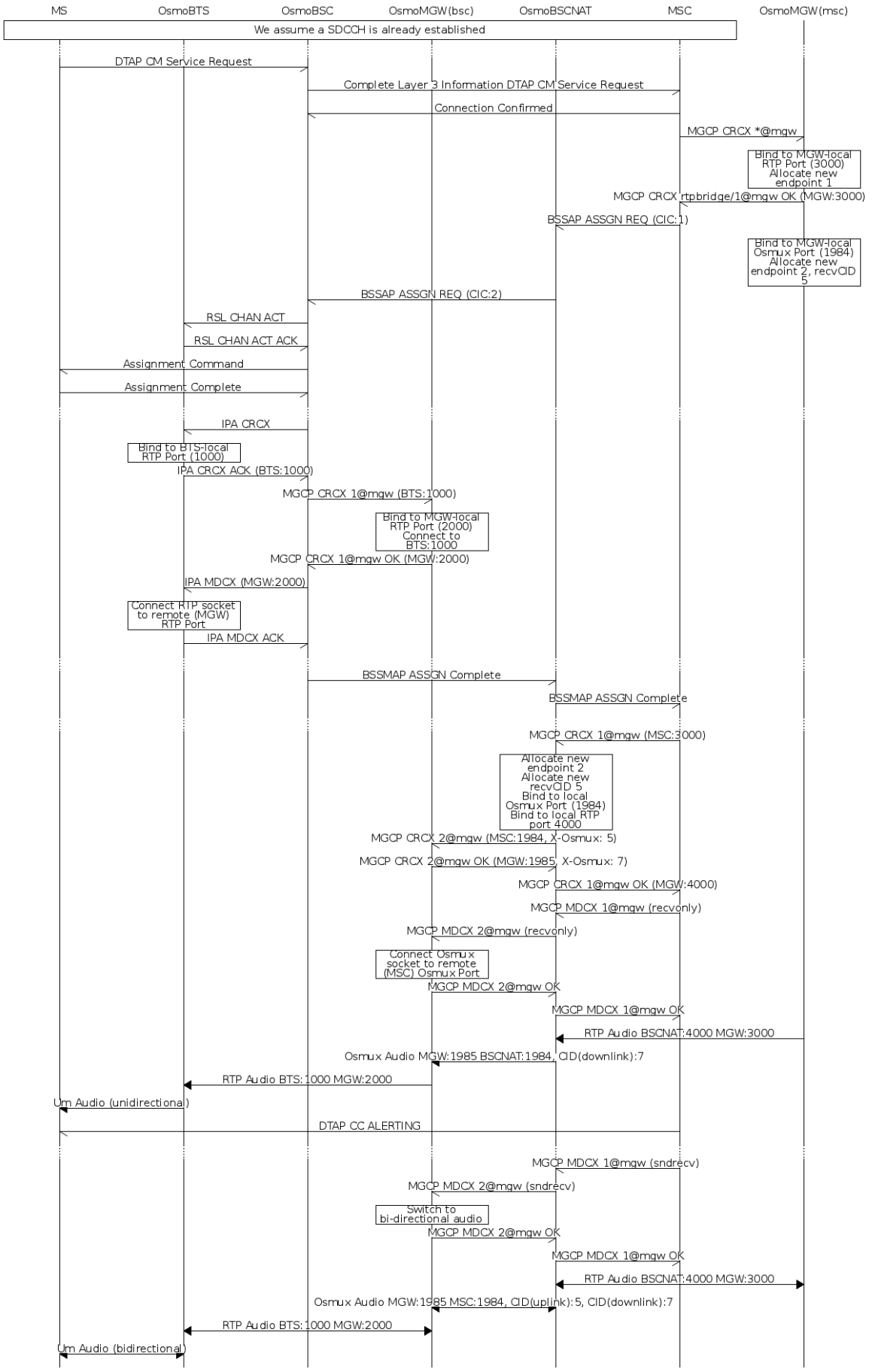


Figure 17: Sample node diagram of a 3GPP AoIP using A/IP with IPA/SCCPlite network and BSC-NAT with Osmux enabled



~~Figure 18: MO-call with Osmux enable on 3GPP AoIP using A/IP with IPA/SCCPlite with a BSC-NAT between BSC and MSC~~

Osmux and MGCP

`X-Osmux` indicates to OsmoMGW that a given connection of an `rtpbridge` endpoint has to be configured in order to handle Osmux frames instead of RTP messages on the data plane.

`X-Osmux` Format

The value part of `X-Osmux` must be one integer in range [0..255], or alternatively only on request messages, an asterisk (*) if the value is not yet known.

`X-Osmux` must be issued in the MGCP header section (typically as its last item), before the SDP section starts.

`X-Osmux` can be included inside `CRCX` and `MDCX` request messages, as well as their respective response messages.

In request messages, the value part of `X-Osmux` specifies the CID to be used by OsmoMGW to *send* Osmux frames to the remote peer for that connection, also known as *sendCID*.

In response messages, the value part of `X-Osmux` specifies the CID where OsmoMGW expect to *receive* Osmux frames from the remote peer for that connection, also known as *recvCID*.

Example: X-Osmux format with a known CID 3.

```
X-Osmux: 3
```

Example: X-Osmux format with a wildcard (not yet known) CID.

```
X-Osmux: *
```

`X-Osmux` Considerations

If the MGCP client is willing to use Osmux for a given connection, it shall specify so during `CRCX` time, and not later. If at `CRCX` time the MGCP client doesn't yet know the *sendCID*, it can use an astersik (*) and provide *sendCID* later within `MDCX` messages.

All subsequent `MDCX` messages sent towards an Osmux connection must contain the original *sendCID* sent during `CRCX`. The same way, all `MDCX` response shall contain the *recvCID* sent during `CRCX`.

The other required connection address parameters, such as IP address, port, and codecs, are negotiated through MGCP and SDP as usual, but in this case the IP address and port specific the Osmux socket IP address and port to use, that together with the Osmux CID conform the entire tuple identifying a Osmux stream.

Since Osmux only supports AMR codec payloads, the SDP must specify use of AMR codec.

Example: CRCX message that instructs OsmoMGW to create an Osmux connection

```
CRCX 189 rtpbridge/1@mgw MGCP 1.0
C: 36
M: sendrecv
X-Osmux: 2

v=0
o=- 36 23 IN IP4 172.18.2.20
s=-
c=IN IP4 1.2.3.4
t=0 0
m=audio 2342 RTP/AVP 112
a=fmtp:112
a=rtpmap:112 AMR/8000/1
a=ptime:20
```

Example: response to CRCX containing the


```
200 189 OK
I: 07E41584
X-Osmux: 2
Z: rtpbridge/1@mgw

v=0
o=- foo 21 IN IP4 172.18.1.20
s=-
c=IN IP4 172.18.1.20
t=0 0
m=audio 11002 RTP/AVP 112
a=rtpmap:112 AMR/8000
a=ptime:20
```

`X-Osmux` Support

`X-Osmux` is known to be supported by OsmoMGW on the MGCP server side, and by OsmoBSC as well as OsmoMSC on the MGCP client side (through libosmo-mgcp-cli). No other programs supporting this feature are known or envisioned at the time of writing this document.

In OmoMGW, Osmux support is managed through VTY.

Example: Sample config file section with Osmux configuration

```
mgcp
 ...
 osmux on ❶
 osmux bind-ip 172.18.1.20 ❷
 osmux port 1984 ❸
 osmux batch-factor 4 ❹
 osmux dummy on ❺
```

❶ Allow clients to set allocate Osmux connections in `rtpbridge` endpoints, while still allowing RTP connections

❷ Bind the Osmux socket to the provided IP address

❸ Bind the Osmux socket to the provided UDP port

❹ Batch up to 4 RTP payloads of the same stream on each Osmux frame

❺ Periodically send Osmux dummy frames, useful to punch a hole in NATs and maintain connections opened.

Osmux Support in OsmoBSC

OsmoBSC in a A/IP with IPA/SCCPlite network setup

In this kind of setup, Osmux is transparent to OsmoBSC and no specific configuration is required here, since the CN-side of the BSC-attached MGW is managed directly by the MSC.

So, in this case, only MSC and MGW (both for MSC-attached one and BSC-attached one) need to be configured explicitly.

OsmoBSC in a 3GPP AoIP network setup

Osmux usage in OsmoBSC in managed through the VTY command `osmux (on|off|only)`. Once enabled (`on` or `only`), OsmoBSC will start appending the vendor specific *Osmux Support* IE in *BSSMAP RESET* and *BSSMAP RESET-ACK* message towards the MSC in order to announce it supports Osmux. This way, the MSC can decide whether to use Osmux or not based on this information when setting up a call (this time using *Osmux CID* IE). It should be noted that this option should not be enabled

unless MSC managing OsmoBSC supports handling this extension IE (like OsmoMSC), a 3rd-party MSC might otherwise refuse the related *RESET*/*RESET-ACK* messages.

OsmoBSC will behave differently during call set up based on the VTY command presented above:

- `off`: If *BSSMAP Assign Request* from MSC contains *Osmux CID* IE, meaning MSC wants to use Osmux for this call, then OsmoBSC will reject the assignment and the call set up will fail.
- `on`: BSC will support and accept both Osmux and non-Osmux (RTP) upon call set up. If *BSSMAP Assign Request* from MSC contains *Osmux CID* IE, OsmoBSC will instruct its MGW to set up an Osmux connection on the CN-side of the MGCP endpoint, and will provide the MSC with its *recvCID* through the extension IE *Osmux CID* appended to the *BSSMAP Assign Complete* message. On the other hand, if *BSSMAP Assign Request* doesn't contain an *Osmux CID* IE, OsmoBSC will instruct its MGW to set up a regular RTP connection on the CN-side of the MGCP endpoint.
- `only`: Same as per `on`, except that OsmoBSC will accept only Osmux calls on the CN-side, this is, if *BSSMAP Assign Request* from MSC doesn't contain an *Osmux CID* IE, it will reject the assignment and the call set up will fail.

VTY Process and Thread management

Most Osmocom programs provide, some support to tune some system settings related to the running process, its threads, its scheduling policies, etc.

All of these settings can be configured through the VTY, either during startup by means of usual config files or through direct human interaction at the telnet VTY interface while the process is running.

Scheduling Policy

The scheduler to use as well as some of its properties (such as realtime priority) can be configured at any time for the entire process. This sort of functionality is useful in order to increase priority for processes running time-constrained procedures, such as those acting on the Um interface, like *osmo-trx* or *osmo-bts*, where use of this feature is highly recommended.

Example: Set process to use RR scheduler

```
cpu-sched
 policy rr 1 ❶
```

❶ Configure process to use *SCHED_RR* policy with real time priority 1

CPU-Affinity Mask

Most operating systems allow for some sort of configuration on restricting the amount of CPUs a given process or thread can run on. The procedure is sometimes called as *cpu-pinning* since it allows to keep different processes pinned on a subset of CPUs to make sure the scheduler won't run two CPU-hungry processes on the same CPU.

The set of CPUs where each thread is allowed to run on is expressed by means of a bitmask in hexadecimal representation, where the right most bit relates to CPU 0, and the Nth most significant bit relates to CPU *N-1*. Setting the bit means the process is allowed to run on that CPU, while clearing it means the process is forbidden to run on that CPU.

Hence, for instance a cpu-affinity mask of *0x00* means the thread is not allowed on any CPU, which will cause the thread to stall until a new value is applied. A mask of *0x01* means the thread is only allowed to run on the 1st CPU (CPU 0). A mask of *0xff00* means CPUs 8-15 are allowed, while 0-7 are not.

For single-threaded processes (most of Osmocom are), it is usually enough to set this line in VTY config file as follows:

```
cpu-sched
 cpu-affinity self 0x01 ❶
```


❶ Allow main thread (the one managing the VTY) only on CPU 0

Or otherwise:

```
cpu-sched
 cpu-affinity all 0x01 ❶
```

❶ Allow all threads only on CPU 0

For multi-threaded processes, it may be desired to run some threads on a subset of CPUs while another subset may run on another one. In order to identify threads, one can either use the TID of the thread (each thread has its own PID in Linux), or its specific Thread Name in case it has been set by the application.

The related information on all threads available in the process can be listed through VTY. This allows identifying quickly the different threads, its current cpu-affinity mask, etc.

Example: Get osmo-trx Thread list information from VTY

```
OsmoTRX> show cpu-sched threads
Thread list for PID 338609:
 TID: 338609, NAME: 'osmo-trx-uhd', cpu-affinity: 0x3
 TID: 338610, NAME: 'osmo-trx-uhd', cpu-affinity: 0x3
 TID: 338611, NAME: 'osmo-trx-uhd', cpu-affinity: 0x3
 TID: 338629, NAME: 'osmo-trx-uhd', cpu-affinity: 0x3
 TID: 338630, NAME: 'osmo-trx-uhd', cpu-affinity: 0x3
 TID: 338631, NAME: 'osmo-trx-uhd', cpu-affinity: 0x3
 TID: 338634, NAME: 'UHDAsyncEvent', cpu-affinity: 0x3
 TID: 338635, NAME: 'TxLower', cpu-affinity: 0x3
 TID: 338636, NAME: 'RxLower', cpu-affinity: 0x3
 TID: 338637, NAME: 'RxUpper0', cpu-affinity: 0x3
 TID: 338638, NAME: 'TxUpper0', cpu-affinity: 0x3
 TID: 338639, NAME: 'RxUpper1', cpu-affinity: 0x3
 TID: 338640, NAME: 'TxUpper1', cpu-affinity: 0x3
```

At runtime, one can change the cpu-affinity mask for a given thread identifying it by either TID or name:

Example: Set CPU-affinity from VTY telnet interface

```
OsmoTRX> cpu-affinity TxLower 0x02 ❶
OsmoTRX> cpu-affinity TxLower 0x03 ❷
```

❶ Allow thread named *TxLower* (*338635*) only on CPU 1

❷ Allow with TID *338636* (*RxLower*) only on CPU 0 and 1

Since thread names are set dynamically by the process during startup or at a later point after creating the thread itself, One may need to specify in the config file that the mask must be applied by the thread itself once being configured rather than trying to apply it immediately. To specify so, the *delay* keyword is using when configuring in the VTY. If the *delay* keyword is not used, the VTY will report and error and fail at startup when trying to apply a cpu-affinity mask for a yet-to-be-created thread.

Example: Set CPU-affinity from VTY config file

```
cpu-sched
 cpu-affinity TxLower 0x01 delay ❶
```

❶ Allow thread named *TxLower* (*338635*) only on CPU 1. It will be applied by the thread itself when created.

Glossary

2FF
2nd Generation Form Factor; the so-called plug-in SIM form factor

3FF
3rd Generation Form Factor; the so-called microSIM form factor

3GPP
3rd Generation Partnership Project

4FF
4th Generation Form Factor; the so-called nanoSIM form factor

A Interface
Interface between BTS and BSC, traditionally over E1 (*3GPP TS 48.008* [3gpp-ts-48-008])

A3/A8
Algorithm 3 and 8; Authentication and key generation algorithm in GSM and GPRS, typically COMP128v1/v2/v3 or MILENAGE are typically used

A5
Algorithm 5; Air-interface encryption of GSM; currently only A5/0 (no encryption), A5/1 and A5/3 are in use

Abis Interface
Interface between BTS and BSC, traditionally over E1 (*3GPP TS 48.058* [3gpp-ts-48-058] and *3GPP TS 52.021* [3gpp-ts-52-021])

ACC
Access Control Class; every BTS broadcasts a bit-mask of permitted ACC, and only subscribers with a SIM of matching ACC are permitted to use that BTS

AGCH
Access Grant Channel on Um interface; used to assign a dedicated channel in response to RACH request

AGPL
GNU Affero General Public License, a copyleft-style Free Software License

ARFCN
Absolute Radio Frequency Channel Number; specifies a tuple of uplink and downlink frequencies

AUC
Authentication Center; central database of authentication key material for each subscriber

BCCH
Broadcast Control Channel on Um interface; used to broadcast information about Cell and its neighbors

BCC
Base Station Color Code; short identifier of BTS, lower part of BSIC

BTS
Base Transceiver Station

BSC
Base Station Controller

BSIC
Base Station Identity Code; 16bit identifier of BTS within location area

BSSGP
Base Station Subsystem Gateway Protocol (*3GPP TS 48.018* [3gpp-ts-48-018])

BVCI
BSSGP Virtual Circuit Identifier

CBCH
Cell Broadcast Channel; used to transmit Cell Broadcast SMS (SMS-CB)

CC
Call Control; Part of the GSM Layer 3 Protocol

CCCH
Common Control Channel on Um interface; consists of RACH (uplink), BCCH, PCH, AGCH (all downlink)

Cell
A cell in a cellular network, served by a BTS

CEPT
Conférence européenne des administrations des postes et des télécommunications; European Conference of Postal and Telecommunications Administrations.

CGI
Cell Global Identifier comprised of MCC, MNC, LAC and BSIC

CSFB
Circiut-Switched Fall Back; Mechanism for switching from LTE/EUTRAN to UTRAN/GERAN when circuit-switched services such as voice telephony are required.

dB
deci-Bel; relative logarithmic unit

dBm
deci-Bel (milliwatt); unit of measurement for signal strength of radio signals

DHCP
Dynamic Host Configuration Protocol (*IETF RFC 2131* [ietf-rfc2131]

downlink
Direction of messages / signals from the network core towards the mobile phone

DSP
Digital Signal Processor

dvnixload
Tool to program UBL and the Bootloader on a sysmoBTS

EDGE
Enhanced Data rates for GPRS Evolution; Higher-speed improvement of GPRS; introduces 8PSK

EGPRS
Enhanced GPRS; the part of EDGE relating to GPRS services

EIR
Equipment Identity Register; core network element that stores and manages IMEI numbers

ESME
External SMS Entity; an external application interfacing with a SMSC over SMPP

ETSI
European Telecommunications Standardization Institute

FPGA
Field Programmable Gate Array; programmable digital logic hardware

Gb
Interface between PCU and SGSN in GPRS/EDGE network; uses NS, BSSGP, LLC

GERAN
GPRS/EDGE Radio Access Network

GFDL
GNU Free Documentation License; a copyleft-style Documentation License

GGSN
GPRS Gateway Support Node; gateway between GPRS and external (IP) network

GMSK
Gaussian Minimum Shift Keying; modulation used for GSM and GPRS

GPL
GNU General Public License, a copyleft-style Free Software License

Gp
Gp interface between SGSN and GGSN; uses GTP protocol

GPRS
General Packet Radio Service; the packet switched 2G technology

GPS
Global Positioning System; provides a highly accurate clock reference besides the global position

GSM
Global System for Mobile Communications. ETSI/3GPP Standard of a 2G digital cellular network

GSMTAP
GSM tap; pseudo standard for encapsulating GSM protocol layers over UDP/IP for analysis

GSUP
Generic ubscriber Update Protocol. Osmocom-specific alternative to TCAP/MAP

GT
Global Title; an address in SCCP

GTP
GPRS Tunnel Protocol; used between SGSN and GGSN

HLR
Home Location Register; central subscriber database of a GSM network

HNB-GW
Home NodeB Gateway. Entity between femtocells (Home NodeB) and CN in 3G/UMTS.

HPLMN
Home PLMN; the network that has issued the subscriber SIM and has his record in HLR

IE
Information Element

IMEI
International Mobile Equipment Identity; unique 14-digit decimal number to globally identify a mobile device, optionally with a 15th checksum digit

IMEISV
IMEI software version; unique 14-digit decimal number to globally identify a mobile device (same as IMEI) plus two software version digits (total digits: 16)

IMSI
International Mobile Subscriber Identity; 15-digit unique identifier for the subscriber/SIM; starts with MCC/MNC of issuing operator

IP
Internet Protocol (*IETF RFC 791* [?])

IPA
ip.access GSM over IP protocol; used to multiplex a single TCP connection

Iu
Interface in 3G/UMTS between RAN and CN

IuCS
Iu interface for circuit-switched domain. Used in 3G/UMTS between RAN and MSC

IuPS
Iu interface for packet-switched domain. Used in 3G/UMTS between RAN and SGSN

LAC
Location Area Code; 16bit identifier of Location Area within network

LAPD
Link Access Protocol, D-Channel (*ITU-T Q.921* [itu-t-q921])

LAPDm
Link Access Protocol Mobile (*3GPP TS 44.006* [3gpp-ts-44-006])

LLC
Logical Link Control; GPRS protocol between MS and SGSN (*3GPP TS 44.064* [3gpp-ts-44-064])

Location Area
Location Area; a geographic area containing multiple BTS

LU
Location Updating; can be of type IMSI-Attach or Periodic. Procedure that indicates a subscriber's physical presence in a given radio cell.

M2PA
MTP2 Peer-to-Peer Adaptation; a SIGTRAN Variant (*RFC 4165* [ietf-rfc4165])

M2UA
MTP2 User Adaptation; a SIGTRAN Variant (*RFC 3331* [ietf-rfc3331])

M3UA
MTP3 User Adaptation; a SIGTRAN Variant (*RFC 4666* [ietf-rfc4666])

MCC
Mobile Country Code; unique identifier of a country, e.g. 262 for Germany

MFF
Machine-to-Machine Form Factor; a SIM chip package that is soldered permanently onto M2M device circuit boards.

MGW
Media Gateway

MM
Mobility Management; part of the GSM Layer 3 Protocol

MNC
Mobile Network Code; identifies network within a country; assigned by national regulator

MNCC
Mobile Network Call Control; Unix domain socket based Interface between MSC and external call control entity like osmo-sip-connector

MNO
Mobile Network Operator; operator with physical radio network under his MCC/MNC

MO
Mobile Originated. Direction from Mobile (MS/UE) to Network

MS
Mobile Station; a mobile phone / GSM Modem

MSC
Mobile Switching Center; network element in the circuit-switched core network

MSC pool
A number of redundant MSCs serving the same core network, which a BSC / RNC distributes load across; see also the "MSC Pooling" chapter in OsmoBSC's user manual [userman-osmobsc] and *3GPP TS 23.236* [3gpp-ts-23-236]

MSISDN
Mobile Subscriber ISDN Number; telephone number of the subscriber

MT
Mobile Terminated. Direction from Network to Mobile (MS/UE)

MTP
Message Transfer Part; SS7 signaling protocol (*ITU-T Q.701* [itu-t-q701])

MVNO
Mobile Virtual Network Operator; Operator without physical radio network

NCC
Network Color Code; assigned by national regulator

NITB
Network In The Box; combines functionality traditionally provided by BSC, MSC, VLR, HLR, SMSC functions; see OsmoNITB

NRI
Network Resource Indicator, typically 10 bits of a TMSI indicating which MSC of an MSC pool attached the subscriber; see also the "MSC Pooling" chapter in OsmoBSC's user manual [userman-osmobsc] and *3GPP TS 23.236* [3gpp-ts-23-236]

NSEI
NS Entity Identifier

NVCI
NS Virtual Circuit Identifier

NWL
Network Listen; ability of some BTS to receive downlink from other BTSs

NS
Network Service; protocol on Gb interface (*3GPP TS 48.016* [3gpp-ts-48-016])

OCXO
Oven Controlled Crystal Oscillator; very high precision oscillator, superior to a VCTCXO

OML
Operation & Maintenance Link (ETSI/*3GPP TS 52.021* [3gpp-ts-52-021])

OpenBSC
Open Source implementation of GSM network elements, specifically OsmoBSC, OsmoNITB, OsmoSGSN

OpenGGSN
Open Source implementation of a GPRS Packet Control Unit

OpenVPN
Open-Source Virtual Private Network; software employed to establish encrypted private networks over untrusted public networks

Osmocom
Open Source MObile COMmunications; collaborative community for implementing communications protocols and systems, including GSM, GPRS, TETRA, DECT, GMR and others

OsmoBSC
Open Source implementation of a GSM Base Station Controller

OsmoNITB
Open Source implementation of a GSM Network In The Box, combines functionality traditionally provided by BSC, MSC, VLR, HLR, AUC, SMSC

OsmoSGSN
Open Source implementation of a Serving GPRS Support Node

OsmoPCU
Open Source implementation of a GPRS Packet Control Unit

OTA
Over-The-Air; Capability of operators to remotely reconfigure/reprogram ISM/USIM cards

PC
Point Code; an address in MTP

PCH
Paging Channel on downlink Um interface; used by network to page an MS

PCU
Packet Control Unit; used to manage Layer 2 of the GPRS radio interface

PDCH
Packet Data Channel on Um interface; used for GPRS/EDGE signalling + user data

PIN
Personal Identification Number; a number by which the user authenticates to a SIM/USIM or other smart card

PLMN
Public Land Mobile Network; specification language for a single GSM network

PUK
PIN Unblocking Code; used to unblock a blocked PIN (after too many wrong PIN attempts)

RAC
Routing Area Code; 16bit identifier for a Routing Area within a Location Area

RACH
Random Access Channel on uplink Um interface; used by MS to request establishment of a dedicated channel

RAM
Remote Application Management; Ability to remotely manage (install, remove) Java Applications on SIM/USIM Card

RF
Radio Frequency

RFM
Remote File Management; Ability to remotely manage (write, read) files on a SIM/USIM card

Roaming
Procedure in which a subscriber of one network is using the radio network of another network, often in different countries; in some countries national roaming exists

Routing Area
Routing Area; GPRS specific sub-division of Location Area

RR
Radio Resources; Part of the GSM Layer 3 Protocol

RSL
Radio Signalling Link (*3GPP TS 48.058* [3gpp-ts-48-058])

RTP
Real-Time Transport Protocol (*IETF RFC 3550* [ietf-rfc3550]); Used to transport audio/video streams over UDP/IP

SACCH
Slow Associate Control Channel on Um interface; bundled to a TCH or SDCCH, used for signalling in parallel to active dedicated channel

SCCP
Signaling Connection Control Part; SS7 signaling protocol (*ITU-T Q.711* [itu-t-q711])

SDCCH
Slow Dedicated Control Channel on Um interface; used for signalling and SMS transport in GSM

SDK
Software Development Kit

SGs
Interface between MSC (GSM/UMTS) and MME (LTE/EPC) to facilitate CSFB and SMS.

SGSN
Serving GPRS Support Node; Core network element for packet-switched services in GSM and UMTS.

SIGTRAN
Signaling Transport over IP (*IETF RFC 2719* [ietf-rfc2719])

SIM
Subscriber Identity Module; small chip card storing subscriber identity

Site
A site is a location where one or more BTSs are installed, typically three BTSs for three sectors

SMPP
Short Message Peer-to-Peer; TCP based protocol to interface external entities with an SMSC

SMSC
Short Message Service Center; store-and-forward relay for short messages

SS7
Signaling System No. 7; Classic digital telephony signaling system

SS
Supplementary Services; query and set various service parameters between subscriber and core network (e.g. USSD, 3rd-party calls, hold/retrieve, advice-of-charge, call deflection)

SSH
Secure Shell; *IETF RFC 4250* [ietf-rfc4251] to 4254

SSN
Sub-System Number; identifies a given SCCP Service such as MSC, HLR

STP
Signaling Transfer Point; A Router in SS7 Networks

SUA
SCCP User Adaptation; a SIGTRAN Variant (*RFC 3868* [ietf-rfc3868])

syslog
System logging service of UNIX-like operating systems

System Information
A set of downlink messages on the BCCH and SACCH of the Um interface describing properties of the cell and network

TCH
Traffic Channel; used for circuit-switched user traffic (mostly voice) in GSM

TCP
Transmission Control Protocol; (*IETF RFC 793* [ietf-rfc793])

TFTP
Trivial File Transfer Protocol; (*IETF RFC 1350* [ietf-rfc1350])

TRX
Transceiver; element of a BTS serving a single carrier

TS
Technical Specification

u-Boot
Boot loader used in various embedded systems

UBI
An MTD wear leveling system to deal with NAND flash in Linux

UBL
Initial bootloader loaded by the TI Davinci SoC

UDP
User Datagram Protocol (*IETF RFC 768* [ietf-rfc768])

UICC
Universal Integrated Chip Card; A smart card according to *ETSI TR 102 216* [etsi-tr102216]

Um interface
U mobile; Radio interface between MS and BTS

uplink
Direction of messages: Signals from the mobile phone towards the network

USIM
Universal Subscriber Identity Module; application running on a UICC to provide subscriber identity for UMTS and GSM networks

USSD
Unstructured Supplementary Service Data; textual dialog between subscriber and core network, e.g. **100 → Your extension is 1234*

VCTCXO
Voltage Controlled, Temperature Compensated Crystal Oscillator; a precision oscillator, superior to a classic crystal oscillator, but inferior to an OCXO

VLR
Visitor Location Register; volatile storage of attached subscribers in the MSC

VPLMN
Visited PLMN; the network in which the subscriber is currently registered; may differ from HPLMN when on roaming

VTY
Virtual TeletYpe; a textual command-line interface for configuration and introspection, e.g. the OsmoBSC configuration file as well as its telnet link on port 4242

Osmocom TCP/UDP Port Numbers

The Osmocom GSM system utilizes a variety of TCP/IP based protocols. The table below provides a reference as to which port numbers are used by which protocol / interface.

Table 14: TCP/UDP port numbers

L4 Protocol	Port Number	Purpose	Software
UDP	2427	MGCP GW	osmo-bsc_mgcp, osmo-mgw
TCP	2775	SMPP (SMS interface for external programs)	osmo-nitb
TCP	3002	A-bis/IP OML	osmo-bts, osmo-bsc, osmo-nitb
TCP	3003	A-bis/IP RSL	osmo-bts, osmo-bsc, osmo-nitb
TCP	4236	Control Interface	osmo-trx
TCP	4237	telnet (VTY)	osmo-trx
TCP	4238	Control Interface	osmo-bts
TCP	4239	telnet (VTY)	osmo-stp
TCP	4240	telnet (VTY)	osmo-pcu
TCP	4241	telnet (VTY)	osmo-bts
TCP	4242	telnet (VTY)	osmo-nitb, osmo-bsc, cellmgr-ng
TCP	4243	telnet (VTY)	osmo-bsc_mgcp, osmo-mgw
TCP	4244	telnet (VTY)	osmo-bsc_nat
TCP	4245	telnet (VTY)	osmo-sgsn
TCP	4246	telnet (VTY)	osmo-gbproxy
TCP	4247	telnet (VTY)	OsmocomBB
TCP	4249	Control Interface	osmo-nitb, osmo-bsc
TCP	4250	Control Interface	osmo-bsc_nat
TCP	4251	Control Interface	osmo-sgsn
TCP	4252	telnet (VTY)	sysmobts-mgr
TCP	4253	telnet (VTY)	osmo-gtphub
TCP	4254	telnet (VTY)	osmo-msc
TCP	4255	Control Interface	osmo-msc
TCP	4256	telnet (VTY)	osmo-sip-connector
TCP	4257	Control Interface	osmo-ggsn, ggsn (OpenGGSN)
TCP	4258	telnet (VTY)	osmo-hlr
TCP	4259	Control Interface	osmo-hlr
TCP	4260	telnet (VTY)	osmo-ggsn
TCP	4261	telnet (VTY)	osmo-hnbgw
TCP	4262	Control Interface	osmo-hnbgw
TCP	4263	Control Interface	osmo-gbproxy
TCP	4264	telnet (VTY)	osmo-cbc
TCP	4265	Control Interface	osmo-cbc
TCP	4266	D-GSM MS Lookup: mDNS serve	osmo-hlr
TCP	4267	Control Interface	osmo-mgw
TCP	4268	telnet (VTY)	osmo-uecups
SCTP	4268	UECUPS	osmo-uecups
TCP	4269	telnet (VTY)	osmo-e1d
TCP	4271	telnet (VTY)	osmo-smlc
TCP	4272	Control Interface	osmo-smlc
UDP	4729	GSMTAP	Almost every osmocom project
TCP	5000	A/IP	osmo-bsc, osmo-bsc_nat
UDP	23000	GPRS-NS over IP default port	osmo-pcu, osmo-sgsn, osmo-gbproxy

Bibliography / References

References

[1] [osmobts-abis-spec] Neels Hofmeyr & Harald Welte. OsmoBTS Abis Protocol Specification. http://ftp.osmocom.org/docs/latest/osmobts-abis.pdf

[2] [userman-osmobts] Osmocom Project: OsmoBTS User Manual. http://ftp.osmocom.org/docs/latest/osmobts-usermanual.pdf

[3] [vty-ref-osmobts] Osmocom Project: OsmoBTS VTY Reference Manual. http://ftp.osmocom.org/docs/latest/-osmobts-vty-reference.pdf

[4] [userman-osmobsc] Osmocom Project: OsmoBSC User Manual. http://ftp.osmocom.org/docs/latest/osmobsc-usermanual.pdf

[5] [vty-ref-osmobsc] Osmocom Project: OsmoBSC VTY Reference Manual. http://ftp.osmocom.org/docs/latest/-osmobsc-vty-reference.pdf

[6] [userman-osmomsc] Osmocom Project: OsmoMSC User Manual. http://ftp.osmocom.org/docs/latest/-osmomsc-usermanual.pdf

[7] [vty-ref-osmomsc] Osmocom Project: OsmoMSC VTY Reference Manual. http://ftp.osmocom.org/docs/-latest/osmomsc-vty-reference.pdf

[8] [userman-osmohlr] Osmocom Project: OsmoHLR User Manual. http://ftp.osmocom.org/docs/latest/osmohlr-usermanual.pdf

[9] [vty-ref-osmohlr] Osmocom Project: OsmoHLR VTY Reference Manual. http://ftp.osmocom.org/docs/latest/-osmohlr-vty-reference.pdf

[10] [userman-osmopcu] Osmocom Project: OsmoPCU User Manual. http://ftp.osmocom.org/docs/latest/osmopcu-usermanual.pdf

[11] [vty-ref-osmopcu] Osmocom Project: OsmoPCU VTY Reference Manual. http://ftp.osmocom.org/docs/latest/-osmopcu-vty-reference.pdf

[12] [userman-osmonitb] Osmocom Project: OsmoNITB User Manual. http://ftp.osmocom.org/docs/latest/-osmonitb-usermanual.pdf

[13] [vty-ref-osmonitb] Osmocom Project: OsmoNITB VTY Reference Manual. http://ftp.osmocom.org/docs/-latest/osmonitb-vty-reference.pdf

[14] [userman-osmosgsn] Osmocom Project: OsmoSGSN User Manual. http://ftp.osmocom.org/docs/latest/-osmosgsn-usermanual.pdf

[15] [vty-ref-osmosgsn] Osmocom Project: OsmoSGSN VTY Reference Manual. http://ftp.osmocom.org/docs/-latest/osmonitb-vty-reference.pdf

[16] [userman-osmoggsn] Osmocom Project: OpenGGSN User Manual. http://ftp.osmocom.org/docs/latest/-osmoggsn-usermanual.pdf

[17] [vty-ref-osmoggsn] Osmocom Project: OsmoGGSN VTY Reference Manual. http://ftp.osmocom.org/docs/-latest/osmoggsn-vty-reference.pdf

[18] [3gpp-ts-23-048] 3GPP TS 23.048: Security mechanisms for the (U)SIM application toolkit; Stage 2 http://www.3gpp.org/DynaReport/23048.htm

[19] [3gpp-ts-23-236] 3GPP TS 23.236: Intra-domain connection of Radio Access Network (RAN) nodes to multiple Core Network (CN) nodes http://www.3gpp.org/DynaReport/23236.htm

[20] [3gpp-ts-24-007] 3GPP TS 24.007: Mobile radio interface signalling layer 3; General Aspects http://www.3gpp.org/DynaReport/24007.htm

[21] [3gpp-ts-24-008] 3GPP TS 24.008: Mobile radio interface Layer 3 specification; Core network protocols; Stage 3. http://www.3gpp.org/dynareport/24008.htm

[22] [3gpp-ts-31-101] 3GPP TS 31.101: UICC-terminal interface; Physical and logical characteristics http://www.3gpp.org/DynaReport/31101.htm

[23] [3gpp-ts-31-102] 3GPP TS 31.102: Characteristics of the Universal Subscriber Identity Module (USIM) application http://www.3gpp.org/DynaReport/31102.htm

[24] [3gpp-ts-31-103] 3GPP TS 31.103: Characteristics of the IMS Subscriber Identity Module (ISIM) application http://www.3gpp.org/DynaReport/31103.htm

[25] [3gpp-ts-31-111] 3GPP TS 31.111: Universal Subscriber Identity Module (USIM) Application Toolkit (USAT) http://www.3gpp.org/DynaReport/31111.htm

[26] [3gpp-ts-31-115] 3GPP TS 31.115: Secured packet structure for (Universal) Subscriber Identity Module (U)SIM Toolkit applications http://www.3gpp.org/DynaReport/31115.htm

[27] [3gpp-ts-31-116] 3GPP TS 31.116: Remote APDU Structure for (U)SIM Toolkit applications http://www.3gpp.org/DynaReport/31116.htm

[28] [3gpp-ts-35-205] 3GPP TS 35.205: 3G Security; Specification of the MILENAGE algorithm set: General

[29] [3gpp-ts-35-206] 3GPP TS 35.206: 3G Security; Specification of the MILENAGE algorithm set: Algorithm specification http://www.3gpp.org/DynaReport/35206.htm

[30] [3gpp-ts-44-006] 3GPP TS 44.006: Mobile Station - Base Station System (MS - BSS) interface; Data Link (DL) layer specification http://www.3gpp.org/DynaReport/44006.htm

[31] [3gpp-ts-44-018] 3GPP TS 44.018: Mobile radio interface layer 3 specification; Radio Resource Control (RRC) protocol http://www.3gpp.org/DynaReport/44018.htm

[32] [3gpp-ts-44-064] 3GPP TS 44.064: Mobile Station - Serving GPRS Support Node (MS-SGSN); Logical Link Control (LLC) Layer Specification http://www.3gpp.org/DynaReport/44064.htm

[33] [3gpp-ts-48-008] 3GPP TS 48.008: Mobile Switching Centre - Base Station system (MSC-BSS) interface; Layer 3 specification http://www.3gpp.org/DynaReport/48008.htm

[34] [3gpp-ts-48-016] 3GPP TS 48.016: General Packet Radio Service (GPRS); Base Station System (BSS) - Serving GPRS Support Node (SGSN) interface; Network service http://www.3gpp.org/DynaReport/48016.htm

[35] [3gpp-ts-48-018] 3GPP TS 48.018: General Packet Radio Service (GPRS); Base Station System (BSS) - Serving GPRS Support Node (SGSN); BSS GPRS protocol (BSSGP) http://www.3gpp.org/DynaReport/48018.htm

[36] [3gpp-ts-48-056] 3GPP TS 48.056: Base Station Controller - Base Transceiver Station (BSC - BTS) interface; Layer 2 specification http://www.3gpp.org/DynaReport/48056.htm

[37] [3gpp-ts-48-058] 3GPP TS 48.058: Base Station Controller - Base Transceiver Station (BSC - BTS) Interface; Layer 3 specification http://www.3gpp.org/DynaReport/48058.htm

[38] [3gpp-ts-51-011] 3GPP TS 51.011: Specification of the Subscriber Identity Module - Mobile Equipment (SIM-ME) interface

[39] [3gpp-ts-51-014] 3GPP TS 51.014: Specification of the SIM Application Toolkit for the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface http://www.3gpp.org/DynaReport/51014.htm

[40] [3gpp-ts-52-021] 3GPP TS 52.021: Network Management (NM) procedures and messages on the A-bis interface http://www.3gpp.org/DynaReport/52021.htm

[41] [etsi-tr102216] ETSI TR 102 216: Smart cards http://www.etsi.org/deliver/etsi_tr/102200_102299/102216/03.00.00_60/tr_102216v030000p.pdf

[42] [etsi-ts102221] ETSI TS 102 221: Smart Cards; UICC-Terminal interface; Physical and logical characteristics http://www.etsi.org/deliver/etsi_ts/102200_102299/102221/13.01.00_60/ts_102221v130100p.pdf

[43] [etsi-ts101220] ETSI TS 101 220: Smart Cards; ETSI numbering system for telecommunication application providers http://www.etsi.org/deliver/etsi_ts/101200_101299/101220/12.00.00_60/ts_101220v120000p.pdf

[44] [ietf-rfc768] IETF RFC 768: Internet Protocol https://tools.ietf.org/html/rfc791

[45] [ietf-rfc793] IETF RFC 793: Transmission Control Protocol https://tools.ietf.org/html/rfc793

[46] [ietf-rfc1035] IETF RFC 1035: Domain Names - Implementation and Specification https://tools.ietf.org/html/-rfc1035

[47] [ietf-rfc1350] IETF RFC 1350: Trivial File Transfer Protool https://tools.ietf.org/html/rfc1350

[48] [ietf-rfc2131] IETF RFC 2131: Dynamic Host Configuration Protocol https://tools.ietf.org/html/rfc2131

[49] [ietf-rfc2719] IETF RFC 2719: Signal Transport over IP https://tools.ietf.org/html/rfc2719

[50] [ietf-rfc3331] IETF RFC 3331: Message Transfer Part 2 User Adaptation Layer https://tools.ietf.org/html/-rfc3331

[51] [ietf-rfc3550] IETF RFC 3550: RTP: A Transport protocol for Real-Time Applications https://tools.ietf.org/-html/rfc3550

[52] [ietf-rfc3596] IETF RFC 3596: DNS Extensions to Support IP Version 6 https://tools.ietf.org/html/rfc3596

[53] [ietf-rfc3868] IETF RFC 3868: SCCP User Adaptation Layer https://tools.ietf.org/html/rfc3868

[54] [ietf-rfc4165] IETF RFC 4165: Message Transfer Part 2 Peer-to-Peeer Adaptation Layer https://tools.ietf.org/-html/rfc4165

[55] [ietf-rfc4251] IETF RFC 4251: The Secure Shell (SSH) Protocol Architecture https://tools.ietf.org/html/-rfc4251

[56] [ietf-rfc4666] IETF RFC 4666: Message Transfer Part 3 User Adaptation Layer https://tools.ietf.org/html/-rfc4666

[57] [ietf-rfc5771] IETF RFC 5771: IANA Guidelines for IPv4 Multicast Address Assignments https://tools.ietf.org/html/rfc5771

[58] [itu-t-q701] ITU-T Q.701: Functional Description of the Message Transfer Part (MTP) https://www.itu.int/rec/-T-REC-Q.701/en/

[59] [itu-t-q711] ITU-T Q.711: Functional Description of the Signalling Connection Control Part https://www.itu.int/rec/T-REC-Q.711/en/

[60] [itu-t-q713] ITU-T Q.713: Signalling connection control part formats and codes https://www.itu.int/rec/T-REC-Q.713/en/

[61] [itu-t-q714] ITU-T Q.714: Signalling connection control part procedures https://www.itu.int/rec/T-REC-Q.714/en/

[62] [itu-t-q921] ITU-T Q.921: ISDN user-network interface - Data link layer specification https://www.itu.int/rec/-T-REC-Q.921/en

[63] [smpp-34] SMPP Develoepers Forum. Short Message Peer-to-Peer Protocol Specification v3.4 http://docs.nimta.com/SMPP_v3_4_Issue1_2.pdf

[64] [gnu-agplv3] Free Software Foundation. GNU Affero General Public License. http://www.gnu.org/licenses/-agpl-3.0.en.html

[65] [freeswitch_pbx] FreeSWITCH SIP PBX https://freeswitch.org

GNU Free Documentation License

Version 1.3, 3 November 2008

PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Section whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not Transparent is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary

formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, Title Page means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section Section C.4.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

a. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

b. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

c. State on the Title Page the name of the publisher of the Modified Version, as the publisher.

d. Preserve all the copyright notices of the Document.

e. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

f. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

g. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

h. Include an unaltered copy of this License.

i. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

j. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

k. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

l. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

m. Delete any section Entitled "Endorsements". Such a section may not be included in the [?].

n. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Sections.

o. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Section and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (c)  YEAR  YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.3
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled ``GNU
Free Documentation License''.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with…Texts." line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

