|  |
| --- |
|  |
| rsys_logo |
| **LTE TotaleNodeB Common Platform**  **User Guide**  **1222464 1.0** |
|  |
|  |
|  |

© 1998-2013 by RadiSys Corporation. All rights reserved.

Radisys, Network Service-Ready Platform, Quick!Start, TAPA, Trillium, Trillium+plus, Trillium Digital Systems, Trillium On Board, TAPA, and the Trillium logo are trademarks or registered trademarks of RadiSys Corporation. All other trademarks, registered trademarks, service marks, and trade names are the property of their respective owners.

**Contents**

[1. Preface 4](#_Toc346639982)

[1.1 Objective 4](#_Toc346639983)

[1.2 Audience 4](#_Toc346639984)

[1.3 Document Organization 4](#_Toc346639985)

[1.4 Release History 5](#_Toc346639986)

[2. Introduction 6](#_Toc346639987)

[2.1 Product Description 6](#_Toc346639988)

[2.2 Definitions and Acronyms 7](#_Toc346639989)

[3. Functional Architecture of Platform Components 8](#_Toc346639990)

[4. Software Requirements 9](#_Toc346639991)

[5. Functionality Support 10](#_Toc346639992)

[6. Open Source Components Used 11](#_Toc346639993)

[7. Steps to Download and Compile the Open Source Component 12](#_Toc346639994)

[7.1. Pre-conditions 12](#_Toc346639995)

[7.2. Autoconf Compilation 12](#_Toc346639996)

[7.3. libxml2-2.7.2 13](#_Toc346639997)

[7.4. libsoap 13](#_Toc346639998)

[7.5. Boost C++ source 15](#_Toc346639999)

[7.6. Openssl 15](#_Toc346640000)

[8. Building of TeNB with Common Platform Components 17](#_Toc346640001)

[9. Execution of Common Platform Components 18](#_Toc346640002)

[9.1. Execution of Common Platform Components 18](#_Toc346640003)

[9.2. OAM Libraries Cleanup 18](#_Toc346640004)

[10. CLI Usage and Commands 19](#_Toc346640005)

[10.1. DRX Feature Settings from CLI 20](#_Toc346640006)

[11. Building Common Platform Components Separately 21](#_Toc346640007)

[12. Appendix 22](#_Toc346640008)

[12.1 ACSLite Setup 22](#_Toc346640009)

[13. References 23](#_Toc346640010)

# Preface

## Objective

This document provides a usage description of the LTE TotaleNodeB common platform components designed by Radisys. This document describes the procedure to setup, configure, and demonstrate a single call using common platform components.

## Audience

The target audiences for this document are:

* Developers involved in TotaleNodeB solution
* System Validation of TotaleNodeB solution
* System Engineers of TotaleNodeB solution
* Architects of the TotaleNodeB solution

## Document Organization

This document contains the following sections:

|  |  |
| --- | --- |
| **Section** | **Description** |
| 1. Preface | Provides the objective and release details. |
| 1. Introduction | Provides an overview of the product, including the product description and features. |
| 1. Functional Architecture of Platform Components | Describes the Platform Component Architecture. |
| 1. Software Requirements | Describes software required for platform components. |
| 1. Functionality Support | Describes the features supported. |
| 1. Open Source Components Used | Describes the OSS component used in the OAM. |
| 1. Steps to Download and Compile the Open Source Items | Describes the steps to download and compile the OSS items. |
| 1. Building of TeNB with Common Platform Components | Describes the building of common platform components. |
| 1. Execution of Common Platform Components | Describes the execution of common platform components. |
| 1. CLI usage and Commands | Describes the usage of CLI. |
| 1. Building Common Platform Components Separately | Describes how to build common platform components individually. |
| 1. References | Lists the reference documents. |

## Release History

The following table lists the history of changes in successive revisions to this document.

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author (s)** | **Description** |
| 1.0 | January 22, 2013 | Rajaram | Conforms to TotaleNodeB release, version 1.0 |

# Introduction

## Product Description

The LTE TotaleNodeB common platform components support the following features:

**HeMS**

* Support for HeMS discovery, HeNB initialization and registration.

**Configuration Management**

* As part of configuration management, TotaleNodeB supports the Broadband Forum defined standard interfaces TR-069 between HeMS and HeNB.
* Support for TR 196, TR 98, TR 157, TR 181, and TR 262 data models.
* Support for dynamic updating of parameters.

**Performance Management**

* Supports various Key Performance Indicators / counters and configurations related to collection intervals, collection periods (TR 262), and so on.
* Support for collection and storing of performance counters.
* Support for reporting of performance counters.
* Support for checking the counters through CLI.

**Log Management**

* Support for logging/trace feature that writes trace information to local files and console.

## Definitions and Acronyms

|  |  |
| --- | --- |
| **Acronym** | **Description** |
| HeMS | Home eNodeB Management System |
| OAM | Operations, Administration and Maintenance |
| PM | Performance Management |
| CLI | Command line interface |
| RAN | Radio Access Network |
| REM | Radio Environment Map |
| RRM | Radio Resource Management |
| SSI | System Services Interface |
| IPsec | IP Security |
| SON | Self-Organizing Networks |
| MIB | Management Information Base |
| NTP | Network Time Protocol |
| FTP | File Transfer Protocol |
| TCP | Transmission Control Protocol |
| TCP/IP | Transmission Control Protocol / Internet Protocol |
| UDP | User Datagram Protocol |
| PHY | Physical Layer |
| FSM | Finite State Machine |
| NAS | Non-Access-Stratum |
| KPI | Key Performance Indicators |
| HeNB | Home eNodeB |
| CM | Configuration Management |
| FM | Fault Management |

# Functional Architecture of Platform Components

The following diagram describes the functional architecture of the common platform components.

**FTP server interactions**

**(KPI, logs)**

**NTP interactions**

**TR-069 interactions with HeMS**

**OAM**

**CLI**

eNodeB Application

**NTP**

**IPSec Controller**

SSI

**MIB**

Messenger

KPI Management framework

Trace/logger

MIB Cache framework

**Stack**

**Manager**

**TR-069**

eNodeB

Protocol Stack

eNodeB Convergence Layer

UDP/IP Stack

**Common Platform Components**

PHY

**REM /SON**

**FTP**

**3rd party IPSec Stack**

Common Platform components comprises of:

* Common Framework and Infrastructure components
* OAM FSM and control module
* TR-069 client module
* Command line interface module
* IPSec controller module
* NTP client
* Management Information Base module (Data Model repository)
* REM controller and SON module.

**Note**: The greyed items are not part of this release.

For details, see Common Platform Software Architecture document.

# Software Requirements

The common platform components are compiled on 32-bit machines.

For building the platform components, the following development tools must be installed:

* sun-java6 (-bin, -jre and -jdk)
* gcc, g++, gdb, binutils, cpp
* libcppunit (-1.x, -dev and -doc)
* libncurses5-dev
* php5-cli (required for fsmc)
* xsltproc (required for applying xslt stylesheets to xml docs)
* subversion

# Functionality Support

This section describes the different functions supported.

|  |  |  |
| --- | --- | --- |
| **Feature** | **Support** | **Remarks** |
| Static Configuration | Yes | Parameters mentioned in the Radisys\_Data\_model.pdf are supported (green colored items in the embedded document). |
| Parameter Validation | Yes | Parameters mentioned in the Radisys\_Data\_model.pdf are supported (green colored items in the embedded document). |
| First time bootup (BOOTSTRAP) procedure | Partial | Factory Reset is not supported. |
| BOOT Procedure | Partial | Rebooting procedure of OAM and UARM are supported. For LARM rebooting MSPD support required. |
| Dynamic Updating | No | Planned in Phase-2. |
| Active/Passive Notification | Yes | -- |
| Command line interface (CLI) | Yes | Parameters mentioned in the Radisys\_Data\_model.pdf are supported (green colored items in the embedded document). |
| Performance Counters | Partial | Checking through CLI is present.  File uploading mechanism is planned in Phase-2.  For supported counters list, see the RadisysKPI\_Supported.xlsx embedded document. |

 

# Open Source Components Used

This section describes the different open source components used.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Reference Item** | **Reference Source** | **Objective of Usage** | **License** | **Comments** |
| Boost C++ | [http://www.boost.org](http://www.boost.org/) | Used by various components like OAM | <http://www.boost.org/users/license.html> |  |
| libCSOAP/nanoHTTP | [http://csoap.sourceforge.net](http://csoap.sourceforge.net/) | Used by TR069/TR196 modules | GNU LGPLv2 |  |
| libXML | <http://xmlsoft.org/> | Used by TR069/TR196 modules | MIT |  |
| OpenSSL | [http://www.openssl.org](http://www.openssl.org/) | Used as a crypto library plugin for strongswan (IPsec client) | OpenSSL and original SSLeay | This is not part of the current delivery. |
| libgmp | [*http://gmplib.org*](http://gmplib.org) | Required by strongswan during compilation of strongswan related binaries | GNU LGPLv2 | This is not part of the current delivery. |
| strongswan | [www.**strongswan**.org](http://www.strongswan.org/) | Version used is strongswan-4.3.4  TotaleNodeB components tenpin and ike-tunnel-ind use strongswan to establish the IPsec tunnel | GNU LGPLv2 | This is not part of the current delivery. |

# Steps to Download and Compile the Open Source Component

The steps to download and compilation of open source components are given in this section.

Please follow the instructions in this section, if the open source code is not provided or if the open source component is provided as a separate tar ball with a few exceptions as explained in detail below.

In the following sections, "TotaleNodeB” is the current directory where the source code is extracted.

If the third party source code is delivered, then,

1. cp <*TotaleNodeB>-*thirdparty.tar.gz <*TotaleNodeB>/tenb\_commonplatform/software/thirdparty/*
2. *cd* <*TotaleNodeB>/tenb\_commonplatform/software/thirdparty/*
3. *tar –zxvf* <*TotaleNodeB>-*thirdparty.tar.gz
4. Go directly to Section 7.3

## Pre-conditions

1. It is always advised to have the latest (compatible) autoconf version. However, this is not mandatory.
2. If errors occur during libxml2-2.7.2 compilation **(Section 7.3)**,thensteps mentioned in **Section 7.2** must beexecuted, followed by steps mentioned in **Section 7.3.**
3. Set the arm-compiler path in PATH environment variable:

**For example:** export PATH=$PATH:/root/arm/bin

## Autoconf Compilation

1. Open the download link: <http://ftp.gnu.org/gnu/autoconf/>
2. Download version: latest autoconf-latest.tar.gz
3. Execute the following command to copy the tar file to <*TotaleNodeB>/tenb\_commonplatform/software/thirdparty* directory

* cp –rf autoconf-latest.tar

<*TotaleNodeB> /tenb\_commonplatform/software/thirdparty/*

* cd <*TotaleNodeB> /tenb\_commonplatform/software/thirdparty/*

1. Extract the contents by executing the command mentioned as follows:

* *tar –xvf autoconf-latest.tar*

1. Copy the script from

*<TotaleNodeB>/tenb\_commonplatform/scripts/compilation\_scripts/configs\_autoconf/configscript\_autoconf*

to

*<TotaleNodeB>/tenb\_commonplatform/software/thirdparty/autoconf-latest/* by executing the following command:

* cp –rf

*<TotaleNodeB>/tenb\_commonplatform/scripts/compilation\_scripts/configs\_autoconf/configscript\_autoconf* *TotaleNodeB/tenb\_commonplatform/software/thirdparty/autoconf-latest/*

1. Go to autoconf-latest directory

* cd <*TotaleNodeB>/tenb\_commonplatform/software/thirdparty/autoconf-latest/*

1. Execute the script with the following command:

* *dos2unix configscript\_autoconf*
* *./configscript\_autoconf*

## libxml2-2.7.2

1. If the <TotaleNodeB>-thirdparty.tar.gz source code is delivered, then go to step 6 directly else go to step 2
2. Open the download link: <ftp://xmlsoft.org/libxml2/>
3. Download version: libxml2-2.7.2.tar.gz
4. Copy the tar file to

*<TotaleNodeB>/tenb\_commonplatform/software/thirdparty*

* *cp –rf libxml2-2.7.2.tar <TotaleNodeB>/tenb\_commonplatform/software/thirdparty*
* *cd <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/*

1. Extract the contents by executing the following command:

* *tar –xvf libxml2-2.7.2.tar*

1. Copy the script from <*TotaleNodeB>/tenb\_commonplatform/scripts/compilation\_scripts/configs\_libxml2/configscript\_libxml2*

to

<*TotaleNodeB>/tenb\_commonplatform/software/thirdparty/libxml2-2.7.2/*

* *cp –rf <TotaleNodeB>/tenb\_commonplatform/scripts/compilation\_scripts/configs\_libxml2/configscript\_libxml2* <*TotaleNodeB>/tenb\_commonplatform/software/thirdparty/libxml2-2.7.2/*

1. Go to the libxml2-2.7.2/ directory

* *cd <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/libxml2-2.7.2/*
* *find –exec dos2unix {} \;*

1. Execute the script:

* *./configscript\_libxml2*

## libsoap

1. If the <TotaleNodeB>-thirdparty.tar.gz source code is delivered, then go to step 9 directly else go to step 2
2. Open the download link: <http://csoap.sourceforge.net/>
3. Download version: libsoap-1.1.0.tar.gz
4. Copy libsoap-1.1.0.tar.gz to <*TotaleNodeB>/tenb\_commonplatform/software/thirdparty/*

* *cp –rf libsoap-1.1.0.tar.gz <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/*

1. Extract the contents by executing the following command:

* *tar –xvf libsoap-1.1.0.tar.gz*

1. Create a “csoap” directory

* *mkdir <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/csoap/*
* *cd <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/csoap/*

1. Copy the contents of libsoap-1.1.0 to <*TotaleNodeB>/tenb\_commonplatform/software/thirdparty/csoap/*

* *cp –rf libsoap-1.1.0/\* <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/csoap/*

1. Remove ONLY the following 3 lines of code at the end of the following file: <*TotaleNodeB>/tenb\_commonplatform/software/thirdparty/csoap/nanohttp/****nanohttp-logging.h***

*#ifdef \_\_cplusplus*

*}*

*#endif*

1. Copy the scripts from

*<TotaleNodeB>/tenb\_commonplatform/scripts/compilation\_scripts/ configs\_csoap/\**

to

*<TotaleNodeB>/tenb\_commonplatform/software/thirdparty/csoap/*

* *cp –rf <TotaleNodeB>/tenb\_commonplatform/scripts/compilation\_scripts/ configs\_csoap/\* <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/csoap/*
* *cd <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/csoap/*
* *find -exec dos2unix {} \;*

1. Execute the scripts with the following command:

* *./configscript\_csoap\_1*
* *./configscript\_csoap\_2*

1. Edit the generated config.h file:

* change the HAVE\_MALLOC define from 0 to 1 on line#39
* comment out the malloc define on line#202

1. Execute the script:

* ./configscript\_csoap\_build
* ./configscript\_csoap\_3

1. Again edit the generated config.h file:

* change the HAVE\_MALLOC define from 0 to 1 on line#39
* comment out the malloc define on line#202

1. Again execute the script:

* *./configscript\_csoap\_build*

## Boost C++ source

1. If the <TotaleNodeB>-thirdparty.tar.gz source code is delivered, then go to step 7 directly else go to step 2
2. Open the download link: <http://sourceforge.net/projects/boost/files/boost/1.52.0/boost_1_52_0.tar.gz/download>
3. Download version: boost\_1\_52\_0.tar.gz
4. Copy the tar file to

*<TotaleNodeB>/tenb\_commonplatform/software/thirdparty/*

* *cp –rf boost\_1\_52\_0.tar.gz <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/*
* *cd <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/*

1. Extract the contents, by executing the following command:

* *tar –xvf boost\_1\_52\_0.tar.gz*

1. Create a boost directory in <*TotaleNodeB>/tenb\_commonplatform/software/libs/common/include/*

* *mkdir <TotaleNodeB>/tenb\_commonplatform/software/libs/common/include/boost*

Copy the contents of *boost\_1\_52\_0/boost/* to *<TotaleNodeB>/tenb\_commonplatform/software/libs/common/include/boost/*

*cp –rf boost\_1\_52\_0/boost/\* <TotaleNodeB>/tenb\_commonplatform/software/libs/common/include/boost/*

**SKIP steps 7 and 8**

1. Create a boost directory in <*TotaleNodeB>/tenb\_commonplatform/software/libs/common/include/*

*mkdir <TotaleNodeB>/tenb\_commonplatform/software/libs/common/include/boost/*

1. *Copy the* ***CONTENTS*** *of <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/boost/ to <TotaleNodeB>/tenb\_commonplatform/software/libs/common/include/boost/*

* *cp –rf <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/boost/\* <TotaleNodeB>/tenb\_commonplatform/software/libs/common/include/boost/*
* *cd <TotaleNodeB>/tenb\_commonplatform/software/libs/common/include/boost/*
* *find -exec dos2unix {} \;*

## Openssl

1. If the <TotaleNodeB>-thirdparty.tar.gz source code is delivered, then SKIP this section else go to step 2
2. Create a directory by name “origopenssl”

mkdir <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/origopenssl

1. Open the download link:

<http://www.openssl.org/source/openssl-0.9.8r.tar.gz>

1. Download version: openssl-0.9.8r.tar.gz
2. Copy the tar file to,

*<TotaleNodeB>/tenb\_commonplatform/software/thirdparty/*origopenssl

* *cp –rf openssl-0.9.8r.tar.gz <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/*origopenssl
* *cd <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/*origopenssl

1. Extract the contents by executing the following command:

* *tar –xvf openssl-0.9.8r.tar.gz*

**Note**: The contents do not get extracted on a mounted device like VmWare, as softlinks are not created. Hence, use a proper Linux machine.

* **mv** *openssl-0.9.8r/\* ../*
* *cd <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/*origopenssl

1. Copy the script from

*<TotaleNodeB>/tenb\_commonplatform/scripts/compilation\_scripts/configs\_openssl/ configscript\_openssl* to *TotaleNodeB/tenb\_commonplatform/software/thirdparty/*origopenssl by executing the following command:

* *cp –rf <TotaleNodeB>/tenb\_commonplatform/scripts/compilation\_scripts/configs\_openssl/configscript\_openssl <TotaleNodeB>/tenb\_commonplatform/software/thirdparty/*origopenssl/
* *cd*

*<TotaleNodeB>/tenb\_commonplatform/software/thirdparty/*origopenssl/

1. Execute the script by executing the following command:

* *./configscript\_openssl*

**Note:** Nano http libraries which are built as part of libsoap compilation procedures do not support cookie based authentication.

# Building of TeNB with Common Platform Components

This section describes the compilation, build and execution of common platform components.

1. **Build Options in Linux**

* Use Linux server which supports the software requirements mentioned in Section 4 for compilation.
* Go to ***enbapp/build*** folder and execute one of the below commands based on the requirement.

|  |  |
| --- | --- |
| **Compilation Command** | **Description** |
| make lnxe2e | Linux compilation |
| make lnxe2e\_oam | Linux compilation with OAM feature enabled |
| make lnxclean | Clean LNX object files |

1. **Build Options in Mindspeed**

* Go to ***enbapp/build*** folder and execute one of the following commands based on the requirement.

|  |  |
| --- | --- |
| **Compilation Command** | **Description** |
| make mse2e | ARM (MSPD) compilation |
| make mse2e\_oam | ARM (MSPD) compilation with OAM feature enabled |
| make msclean | Clean MSPD object files |

* After compilation, go to enbapp/build directory and execute the *TeNBCP\_build* script.

Usage:

./TeNBCP\_build mspd

The above command creates the following directory structures:

*rsys/bin*

*rsys/config*

*rsys/lib*

rsys/nmmscripts

rsys/setup

* Please tar the binaries before taking into any server/setup.

# Execution of Common Platform Components

## Execution of Common Platform Components

The steps mentioned below describe the steps to be followed for executing common platform components.

1. Customize the TeNBCP\_env script to suit the setup, which is available under “rsys/setup” path
2. Set the date and time in board (date -s "2 OCT 2012 18:00:00")
3. Execute the script **“setup\_TeNB”**

**Note***:* Use “. setup\_TeNB” and DO NOT use ./ or sh as the env path is not set otherwise. This is ONE time only.

1. Since the environment files are shell specific, to execute the binaries in a different shell, execute the TeNBCP\_env file in that shell also. Verify the environment variables set via TeNBCP\_env file.
2. Run “. start\_TeNB" script present in rsys/setup directory.

**Note**: Please check whether customization is required for start\_TeNB script.

1. Run ./cli to change the values required for configuration in another shell after executing TeNBCP\_env

## OAM Libraries Cleanup

Go to the following directory – enbapp/build

Execute the respective clean command for Linux or ARM.

Linux:  *make lnxclean\_oam*

ARM:  *make msclean\_oam*

# CLI Usage and Commands

This section describes the CLI usage. CLI is used for the following purposes:

1. Setting the LTE parameters

Command: *oam.set <param\_name> <value>* - Set the parameter

1. Getting the LTE parameters

Command: *oam.getwild <Parameter Name>*

For example,to get the current admin state the command must be as given as follows:

*oam.getwild LTE\_FAP\_ADMIN\_STATE*

To get all the parameters following command is used:

*oam.getwild LTE\_\**

1. Retrieving the Performance Counter

Command: *oam.pollkpis interval*

*Counters are redirected to OAM-sm\* trace file, which is put into /opt/trace/*

1. Retrieving PM files

Command: *tr69.gen.pm.file <filename>*

File is generated in the place where CLI is running.

The list of LTE parameters supported is embedded here.



Following mandatory list of parameters to be set via CLI:

1. MANAGEMENT\_SERVER
2. FEMTO\_GATEWAY\_IP\_ADDRESS\_1ST
3. FEMTO\_GATEWAY\_SERVER\_1
4. IKE\_IPSEC\_ENABLE
5. LTE\_PLMNID\_1 (primary PLMN id )
6. LTE\_PRIMARY\_PLMN\_1
7. LTE\_SIGLINK\_SERVER\_LIST (MME IP)

## MME IP Updating from CLI

This section describes the configuration of MME IP from CLI.

**For Single MME:**

oam.set LTE\_SIGLINK\_SERVER\_LIST **"**172.27.4.216**"**

**For Multiple MMEs:**

oam.set LTE\_S1CON\_MODE **All**

oam.set LTE\_SIGLINK\_SERVER\_LIST **"**172.27.4.216**","**172.27.4.219**","**172.27.4.220**”**

## DRX Feature Settings from CLI

This section describes the mapping of the values to be set from CLI for the DRX feature.

For example:

If LongDrxCycleGbr value to be given is 1, then as per TR-196, the value to be given from CLI is 20.

The mapping is as below:-

                        <range minInclusive="10" maxInclusive="10" />               enum value = 0

                        <range minInclusive="20" maxInclusive="20" />               enum value = 1

                        <range minInclusive="32" maxInclusive="32" />               enum value = 2

                        <range minInclusive="40" maxInclusive="40" />               enum value = 3

                        <range minInclusive="64" maxInclusive="64" />               enum value = 4

                        <range minInclusive="80" maxInclusive="80" />               enum value = 5

                        <range minInclusive="128" maxInclusive="128" />          enum value = 6

                        <range minInclusive="160" maxInclusive="160" />          enum value = 7

                        <range minInclusive="256" maxInclusive="256" />          enum value = 8

                        <range minInclusive="320" maxInclusive="320" />          enum value = 9

                        <range minInclusive="512" maxInclusive="512" />          enum value = 10

                        <range minInclusive="640" maxInclusive="640" />          enum value = 11

                        <range minInclusive="1024" maxInclusive="1024" />     enum value = 12

                        <range minInclusive="1280" maxInclusive="1280" />     enum value = 13

                        <range minInclusive="2048" maxInclusive="2048" />     enum value = 14

                        <range minInclusive="2560" maxInclusive="2560" />     enum value = 15

**Default value of LongDrxCycleGbr is 1 (which is 20 when viewed from CLI)**

**Default value of LongDrxCycleNonGbr is 3 (which is 40 when viewed from CLI)**

# Building Common Platform Components Separately

To build platform code on Linux, the software mentioned in Section 4must be installed.

The compilation is done only on 32-bit for now.

**To clean the directories:**

*make clean PRODUCT=generic HARDWARE=generic BUILD=i686-linux*

**To build individual components:**

*make PRODUCT=generic HARDWARE=generic BUILD=i686-linux*

**To build all the components:**

*cd Master\_MSPD\_LIB/tenb\_commonplatform/software/products/fap/hbs2/hbs2-iu/*

*make distro PRODUCT=generic HARDWARE=generic BUILD=i686-linux*

After building the libraries and binaries, all these are fetched at one shot with the script **“TeNBCP\_linux\_load/TeNBCP\_arm\_load”** which is found in the

Master\_MSPD\_LIB/tenb\_commonplatform/scripts/ directory.

After placing the config files, libraries and binaries in a common path, execute the “setup\_TeNB” script once.

**Note**: Convert all the config files and scripts using dos2unix command.

setup\_TeNB script MUST be executed only once in a lifetime (until the test machine or the laptop on which it is executing is changed).

For arm compilation, use the following command instead of the Linux command:

*make PRODUCT=hbs2-iu HARDWARE=hbs2-4 BUILD=arm-none-linux-gnueabi*

# Appendix

## 12.1 ACSLite Setup

ACSLite is used as a HeMs simulator for end-to-end testing. The steps mentioned below describe the steps to be followed for ACSLite setup.

1. Through “vncviewer” login to 172.27.4.244:6

Give the password as, *lte@sqa1*

**Note**: Also *“:7” (password:lte@sqa2) and “:8” (password: lte@sqa3) can be used.*

1. Once logged in, from an internet browser connect to the following mentioned URL:

<http://172.27.4.244/management>

1. Use the following user name and password to log in:

User Name: *lte.sqa1*

Password: *lte@sqa1*

**Note 1***:* Also *“lte.sqa2”* is used as the user name and password as “*lte@sqa2”*.

**Note 2***:* The 172.27.4.244 IP must be pingable from the eNodeB (add required routes at eNodeB as well as ACSLite).

1. In “Managed Devices” tab at ACSLite,
   * The following username and password is supported and must be used.

Username: *cpe*

Password: *cpe*

* + Under *“HTTP Authentication Method*”, select only *“BASIC”.*
  + The *“HTTPS Authentication Method”,* must be left as *“DEFAULT”.*

1. Refer to the ACSLite usage documents embedded here for details.



**Note***:* ACSLite is used as a simulator for HeMS, by Radisys internally*.*

# References

Refer to the following documents for more information.

1. Radisys\_LTE\_Common\_Platform\_SwArch, Software Architecture of Common Platform in TeNB, 11th September 2012.
2. LTE\_TotaleNodeB\_Solution\_User\_Guide\_05\_10.



[**www.radisys.com**](http://www.radisys.com)