User Guide

**Control Plane, User Plane Gateways;**

**LTE EPC;**

**Installation Guide;**

**Troubleshooting Guide;**

**(Release 1.9)**

Table of Contents

[1](#_heading=h.1fob9te) Acronyms and Definitions 4

[2](#_heading=h.3znysh7) Introduction 4

[3](#_heading=h.2et92p0) Background 5

[3.1](#_heading=h.tyjcwt) Source Repository 5

[3.1.1](#_heading=h.3dy6vkm) Ngic-rtc 6

[3.1.2](#_heading=h.2s8eyo1) Epctools 6

[3.1.3](#_heading=h.17dp8vu) Freediameter 6

[3.1.4](#_heading=h.26in1rg) Libpfcp 6

[3.1.5](#_heading=h.lnxbz9) Libgtpv2c 6

[3.2](#_heading=h.44sinio) License 6

[3.3](#_heading=h.2jxsxqh) Specifications Referred 6

[3.4](#_heading=h.3j2qqm3) Release Updates 6

[3.4.1](#_heading=h.1y810tw) Release Details 7

[3.4.1.1](#_heading=h.2xcytpi) New Features 7

[3.4.1.2](#_heading=h.1ci93xb) Not Supported E-UTRAN Features 7

[4](#_heading=h.3as4poj) Deployment Options 8

[4.1](#_heading=h.ihv636) Combined SGW-PGW Architecture 8

[4.1.1](#_heading=h.2grqrue) Combined GW Deployment and Terminology 8

[4.1.2](#_heading=h.vx1227) Configuration Settings 8

[4.2](#_heading=h.4f1mdlm) Split GW Architecture 9

[4.2.1](#_heading=h.19c6y18) Split GW Deployment and Terminology 9

[4.2.2](#_heading=h.3tbugp1) Configuration Settings 9

[5](#_heading=h.28h4qwu) Downloading, Building and Launching 10

[5.1](#_heading=h.nmf14n) Downloading 10

[5.2](#_heading=h.48pi1tg) Installation of Control Plane and Data Plane 10

[5.2.1](#_heading=h.haapch) Control Plane Installation 10

[5.2.2](#_heading=h.319y80a) Data Plane Installation 12

[5.3](#_heading=h.1gf8i83) Building 13

[5.3.1](#_heading=h.40ew0vw) Runtime Flags 13

[5.4](#_heading=h.2fk6b3p) Dependencies 14

[5.5](#_heading=h.upglbi) Configuration 15

[5.5.1](#_heading=h.3ep43zb) Editing Control plane Configuration 15

[5.5.2](#_heading=h.1tuee74) Editing Date Plane Configuration 17

[5.5.3](#_heading=h.4du1wux) Editing Gx App Configuration 18

[5.5.4](#_heading=h.2szc72q) DNS server setup for UPF selection 19

[5.5.4.1](#_heading=h.184mhaj) Basic DNS Server Installation 19

[5.5.4.2](#_heading=h.3s49zyc) Update /etc/hostname and /etc/hosts 21

[5.5.5](#_heading=h.279ka65) User level packet copying components setup 21

[5.6](#_heading=h.meukdy) Launching 23

[5.6.1](#_heading=h.36ei31r) Running Control Plane, Data Plane and Gx Applications 24

[5.6.2](#_heading=h.1ljsd9k) Bring Up the Data Plane Path 24

[5.6.3](#_heading=h.45jfvxd) Bring Up User Level Packet Copying Components 24

[6](#_heading=h.2koq656) Monitoring 24

[6.1](#_heading=h.zu0gcz) Using CLI 24

[6.2](#_heading=h.3jtnz0s) Logging 25

[6.2.1](#_heading=h.1yyy98l) c3pocli for Logging Configuration 26

[6.3](#_heading=h.4iylrwe) DP pcap generation framework Commands 27

[6.4](#_heading=h.1d96cc0) ULPC pcap generation commands 27

[7](#_heading=h.4anzqyu) Troubleshooting 28

[7.1](#_heading=h.2pta16n) Binding PCI Interface to dpdk Driver 28

[7.2](#_heading=h.14ykbeg) Setting Environment for Control and User Planes 29

[7.3](#_heading=h.3oy7u29) Data Not Passing Through User Plane 29

[7.4](#_heading=h.243i4a2) Giving Permission to Shell Script File 29

[7.5](#_heading=h.j8sehv) Memory Issue in Section 5.3 Control/User Plane build 29

[8](#_heading=h.338fx5o) References 30

# Acronyms and Definitions

**Table 1. Acronyms and Definitions**

|  |  |
| --- | --- |
| **Acronym** | **Definition** |
| 3GPP | 3rd Generation Partnership Project |
| APN | Access Point Name |
| C3PO | Clean CUPS Core for Packet Optimization –a code repository under OMEC-project |
| CDF | Charging Data Function |
| CDR | Charging Data Record |
| CLI | Command Line Interface |
| CTF | Charging Trigger Function |
| CUPS | Control and User Plane Separation – Provides architecture enhancements for the separation of control and user functionality in EPC |
| DNS | Domain Name System |
| EPC | Evolved Packet Core |
| LTE | Long Term Evolution |
| MME | Mobile Management Entity |
| NGIC | Next Generation Infrastructure Core –a code repository under OMEC-project |
| OMEC | Open Mobile Evolved Core – the first full-featured, scalable, high performance open source EPC |
| ONF | Open Networking Foundation –non-profit operator led consortium driving transformation of network infrastructure and carrier business models |
| OSS | Operations Support System |
| PCRF | Policy and Charging Rules Function |
| PDN | Packet Data Network |
| PGW-C | Packet Data Network Gateway – Control Plane – an EPC gateway controller that processes control signal messages for PGW-Us. |
| PGW-U | Packet Data Network Gateway – User Plane – an EPC gateway that provides connectivity from the UE to external packet data networks |
| rtc | Run to completion |
| SAE | System Architecture Evolution |
| SAEGW-C | System Architecture Evolution Gateway – Control Plane – an EPC gateway that is a combination of S-GW and P-GW control plane nodes |
| SAEGW-U | System Architecture Evolution Gateway – User Plane – an EPC gateway that is a combination of S-GW and P-GW user plane nodes. |
| SGW-C | Serving Gateway – Control plane – an EPC gateway controller that processes, routes, and forwards control signal packets for SGW-Us |
| SGW-U | Serving Gateway – User plane – an EPC gateway that routes and forwards user data packets |
| SGX | Software Guard Extensions – Intel’s protected environment that contains the code and data of a security-sensitive computation |
| TST | Technical Steering Team |
| UE | User Equipment |
| UPF | User Plane Function |
| ULPC | User Level Packet Copying |

# Introduction

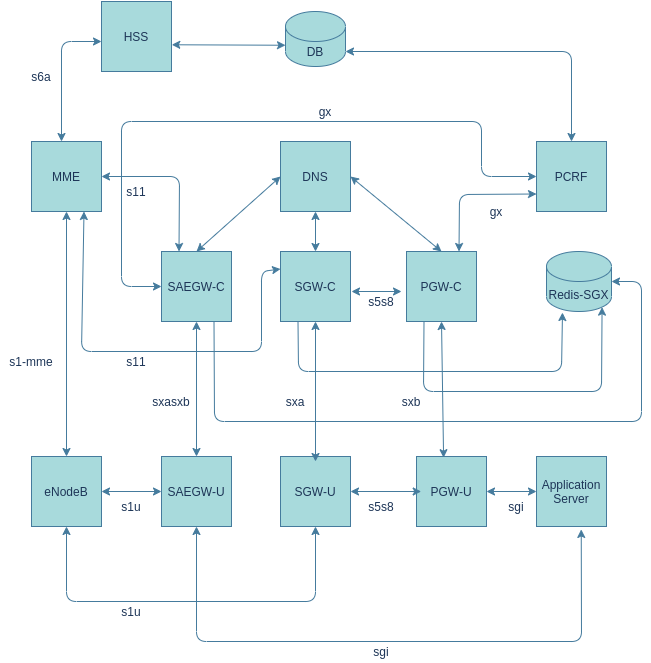
This user guide is for LTE EPC Gateways developers and users as part of OMEC (Open Mobile Enhanced Core). This guide is intended to help users build, configure and deploy OMEC with the supported Gateway combinations. This user guide also provides some technical insights into the architecture of the Gateway code and documents open issues and backlog.

# Background

The OMEC project is an open source LTE EPC development under Open Networking Foundation (ONF) consortium. Under OMEC, there are a suite of projects working together to create EPC solutions. The ‘ngic-rtc’ project provides Gateway components of the EPC.

Figure 1 shows the components under EPC. The ‘ngic-rtc’ project provides source code for the SGW-C, PGW-C, SGW-U and PGW-U components shown in Figure 1.

This document is a user guide for ‘ngic-rtc’ source code and Gateways combinations mentioned above. Throughout this document all gateway combinations (SGW-C, PGW-C, SGW-U, PGW-U, SAEGW-C and SAEGW-U) are commonly referred to as Gateways.

****

**Figure 1. EPC Architecture**

## Source Repository

The following are the source code repositories relevant for Gateway development, deployment and testing.

### Ngic-rtc

The public repository for the Gateways is hosted at the following location and is maintained by OMEC Technical Steering Team (TST) members.

<https://github.com/omec-project/ngic-rtc>.git

Branch: e-utran-features

### Epctools

The public repository for epctools is at https://github.com/omec-project/epctools.git

### Freediameter

The Public repository for freediameter is at https://github.com/omec-project/freeDiameter.git

Branch : gx

### Libpfcp

The Public repository for PFCP protocol library at <https://github.com/omec-project/libpfcp.git>

### Libgtpv2c

The Public repository for GTPV2C protocol library at <https://github.com/omec-project/libgtpv2c.git>

## License

The complete source code is under [**Apache 2.0**](https://www.apache.org/licenses/LICENSE-2.0) license. Any new open to be linked with must follow the license restriction and license contamination against Apache 2.0 has to be avoided.

The source code has gone through the following source control and license check process:

<License check process, contamination checks, blackduck etc.>

## Specifications Referred

All development was based on release 15 of the 3gpp specifications listed in Section 8.

## Release Updates

This document describes release version 1.9. The source code is available at the following source control:

<https://github.com/omec-project/ngic-rtc>

Branch: e-utran-features

### Release Details

#### New Features

The new features included in release 1.9 are listed in Table 2.

**Table 2. New Features**

|  |  |
| --- | --- |
| **Feature** | **Reference** |
| Network Triggered Service Request | TS 23.401 - 5.3.4.3 |
| [HSS Init Subscribed QoS Modification](https://docs.google.com/document/d/1xHoxv2zfli7KEUiJdf_bD0ZER4efzDF7/edit#heading=h.1ksv4uv) | TS [23.401 - 5.4.2.2](https://docs.google.com/document/d/1xHoxv2zfli7KEUiJdf_bD0ZER4efzDF7/edit#heading=h.1ksv4uv) |
| [dedicated bearer in attac](https://docs.google.com/document/d/1xHoxv2zfli7KEUiJdf_bD0ZER4efzDF7/edit#heading=h.44sinio)h | TS [23.401 - F-1](https://docs.google.com/document/d/1xHoxv2zfli7KEUiJdf_bD0ZER4efzDF7/edit#heading=h.44sinio) |
| [MO Data transport in NAS PDU](https://docs.google.com/document/d/1xHoxv2zfli7KEUiJdf_bD0ZER4efzDF7/edit#heading=h.26in1rg) | TS [23.401 - 5.3.4B.2](https://docs.google.com/document/d/1xHoxv2zfli7KEUiJdf_bD0ZER4efzDF7/edit#heading=h.26in1rg) |
| [MT Data transport in NAS PDU](https://docs.google.com/document/d/1xHoxv2zfli7KEUiJdf_bD0ZER4efzDF7/edit#heading=h.lnxbz9) | TS [23.401 - 5.3.4B.3](https://docs.google.com/document/d/1xHoxv2zfli7KEUiJdf_bD0ZER4efzDF7/edit#heading=h.lnxbz9) |
| [Establishment of S1-U bearer during Data Transport in Control Plane CIoT EPS Optimisation](https://docs.google.com/document/d/1xHoxv2zfli7KEUiJdf_bD0ZER4efzDF7/edit#heading=h.35nkun2) | TS [23.401 - 5.3.4B.4](https://docs.google.com/document/d/1xHoxv2zfli7KEUiJdf_bD0ZER4efzDF7/edit#heading=h.35nkun2) |
| [S1 based Handover](https://docs.google.com/document/d/1xHoxv2zfli7KEUiJdf_bD0ZER4efzDF7/edit#heading=h.2jxsxqh) | TS 23.401 - 5.5.1.2 |
| [TAU procedure with Serving GW change and data forwarding](https://docs.google.com/document/d/1xHoxv2zfli7KEUiJdf_bD0ZER4efzDF7/edit#heading=h.z337ya) | TS 23.401 - 5.3.3.1A |
| [Tracking Area Update With SGW Change flow](https://docs.google.com/document/d/1xHoxv2zfli7KEUiJdf_bD0ZER4efzDF7/edit#heading=h.3j2qqm3) | TS 23.401 - 5.3.3.0, 5.3.3.1 |
| IPv6 interfaces and applications support in Deployment models | TS 23.401 - 5.3.1 |
| IPv6, IPv4v6 support in Mobility Management | TS 23.401 - 5.3.1 |

#### Not Supported E-UTRAN Features

The not supported E-UTRAN features are listed in Table 3.

**Table 3. Not Supported E-UTRAN Features**

|  |  |
| --- | --- |
| **Feature** | **Reference** |
| SGSN-Initiated Detach Procedure with ISR activated | TS 23.401 - 5.3.8.3A |
| UE-initiated Detach procedure for GERAN/UTRAN with ISR activated | TS 23.401 - 5.3.8.2.2 |
| PDN GW Pause of charging procedure | TS 23.401 - 5.3.6A |
| E-UTRAN to UTRAN Iu mode Inter RAT handover | TS 23.401 - 5.5.2.1 |
| Support for Non-IP Data Delivery (NIDD) | TS 23.401 - 4.3.17.8 |
| Multimedia Priority Service | TS 23.401 - 4.3.18 |
| UTRAN Iu mode to E-UTRAN Inter RAT handover | TS 23.401 - 5.5.2.2 |
| E-UTRAN to GERAN A/Gb mode Inter RAT handover | TS 23.401 - 5.5.2.3 |
| GERAN A/Gb mode to E-UTRAN Inter RAT handover | TS 23.401 - 5.5.2.4 |
| Interoperation procedures | TS 23.401 - D.3 |
| QoS Control | TS 29.244 - 5.4.4 |

# Deployment Options

The ngic-rtc software can be configured to run in two deployment modes, Combined SGW-PGW Architecture and Split GW Architecture, as described below.

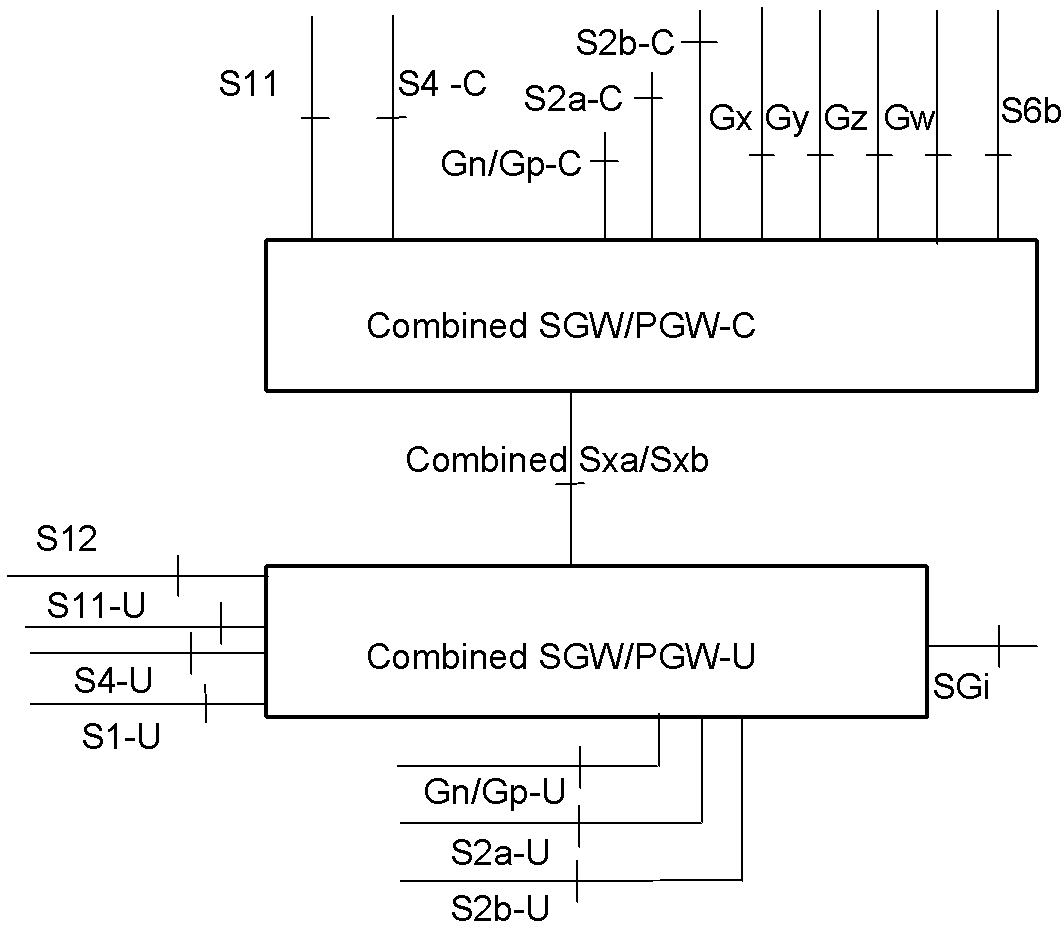
## Combined SGW-PGW Architecture

In the Combined SGW-PGW Architecture, a single instance of the control gateway operates as a combination of SGW and PGW for the control plane, and a single instance of user gateway operates as a combination of SGW and PGW for the user plane.

TS 23.214 section 4.2.2 describes the combined gateway architecture.

### Combined GW Deployment and Terminology

The deployment diagram, taken from TS 23.214, Section 4.2.2, is shown in Figure 2. The control plane shown as “Combined SGW/PGW-C” in the diagram is referred to as “SAEGW-C” throughout this document. The user plane shown as “Combined SGW/PGW-U” in the diagram is referred to as “SAEGW-U” throughout this document.

****

**Figure 2. Deployment diagram**

### Configuration Settings

The following configuration runtime setting is used to run ngic-rtc as SAEGW-C.

CP\_TYPE = 03

For the details of the setup configuration refer to Section 5.5

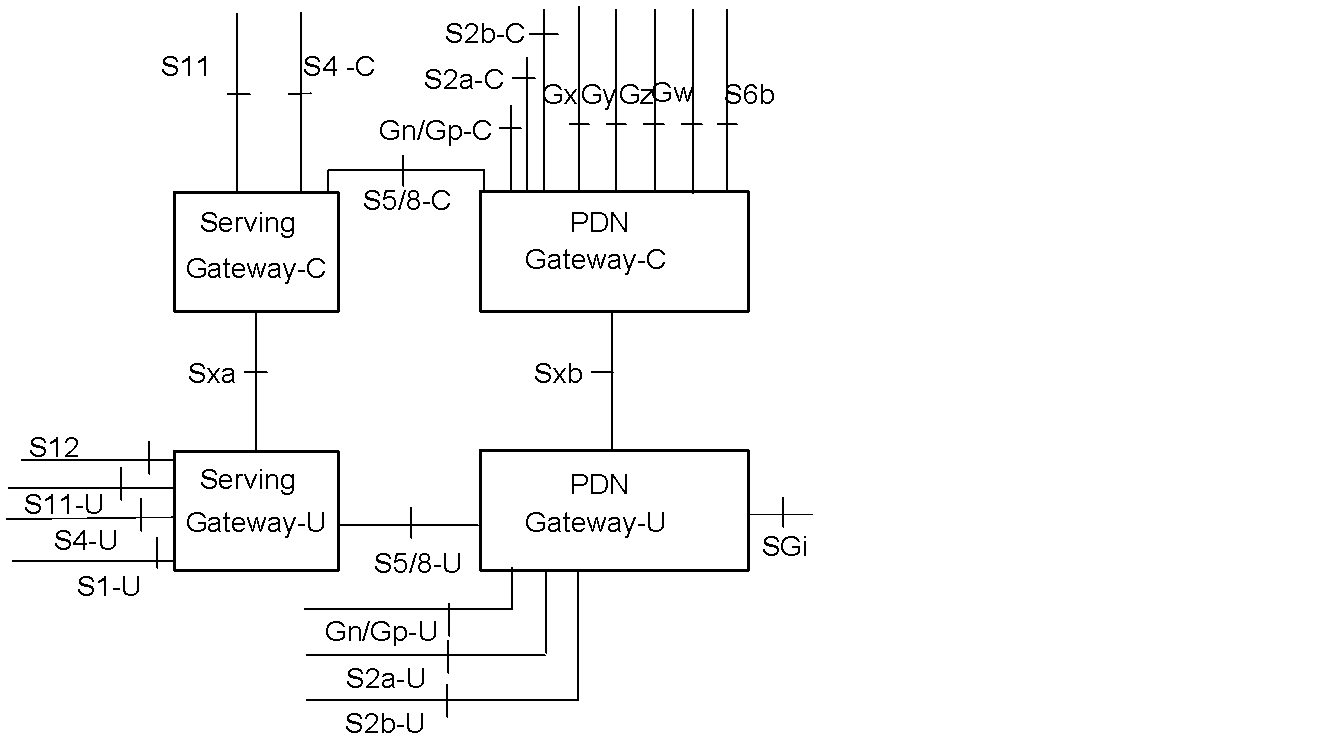
## Split GW Architecture

Operating as a split GW in the CUPS architecture, the ngic-rtc software can be deployed to run as four different types of instances: SGW-C, PGW-C, SGW-U and PGW-U.

TS 23.214 section 4.2.1 describes the split gateway architecture.

### Split GW Deployment and Terminology

The deployment diagram, taken from TS 23.214, Section 4.2.1, is shown in Figure 3. The control plane shown as “Serving Gateway-C” in the diagram is referred to as “SGW-C” throughout this document. The user plane shown as “Serving Gateway-U” in the diagram is referred to as “SGW-U” throughout this document. The control plane shown as “PDN Gateway-C” in the diagram is referred to as “PGW-C” throughout this document. The user plane shown as “PDN Gateway-U” in the diagram is referred to as “PGW-U” throughout this document.

****

**Figure 3. Deployment diagram**

### Configuration Settings

The following runtime configuration setting is used to run ngic-rtc as SGW-C.

CP\_TYPE = 01

The following configuration setting is used to run ngic-rtc as PGW-C.

CP\_TYPE = 02

Please refer to Section 5.5 for the details of overall configuration.

# Downloading, Building and Launching

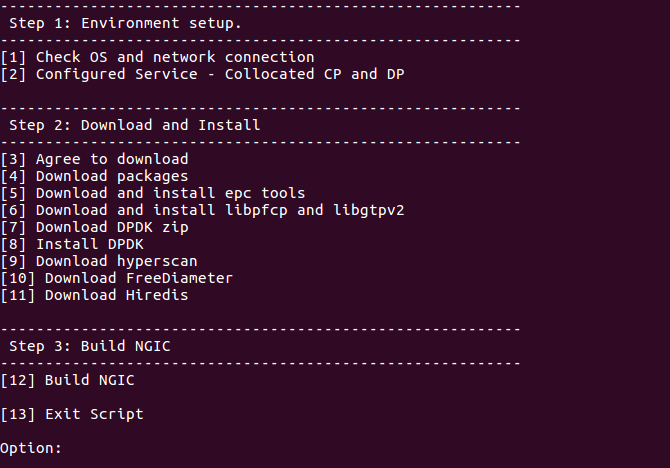
## Downloading

To download from github, execute the following.

*git clone* <https://github.com/omec-project/ngic-rtc>.git

## Installation of Control Plane and Data Plane

To install a control plane or user plane, run the *install.sh* script from the ngic-rtc folder. The *install.sh* script will provide the menu shown in Figure 4.



**Figure 4. Install.sh Menu**

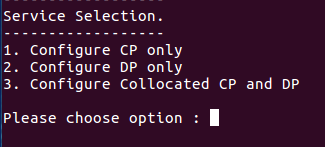
### Control Plane Installation

Follow the steps below to install the control plane.

1. Enter “1” to obtain information about network connectivity and OS information
2. Enter “2” to configure control plane

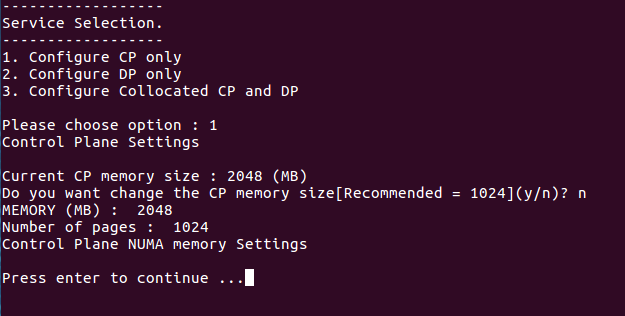
Note: The configuration of the Collocated CP and DP Architecture is not supporting.

After selecting option 2, the new selection menu will appear as shown in Figure 5.

****

**Figure 5. Service Selection Menu**

After entering “1” for control plane installation, the selection menu shown in Figure 6 will appear. Note that “n” has already been selected in the figure. If you want to change memory then enter “y”, otherwise enter “n”.

****

**Figure 6. Control Plane Setting Selection**

1. Enter “3” and agree to download dependent libraries or packages. Enter “y” after selecting option “3”.
2. Enter “4” to download packages.
3. Enter “5” to download and install the epctools.
4. Enter “6” to download and install the PFCP and GTPV2C protocol library.
5. Enter “7” to download the DPDK zip file.
6. Enter “8” to install the Data Plane Development Kit.
7. Enter “9” to download the FreeDiameter library.
8. Enter “10” to download the hiredis library.
9. Enter “11” to build NGIC. Enter “y” to build NGIC in debug mode otherwise enter “n”.

This option builds and links all the libraries used to make the final binary.

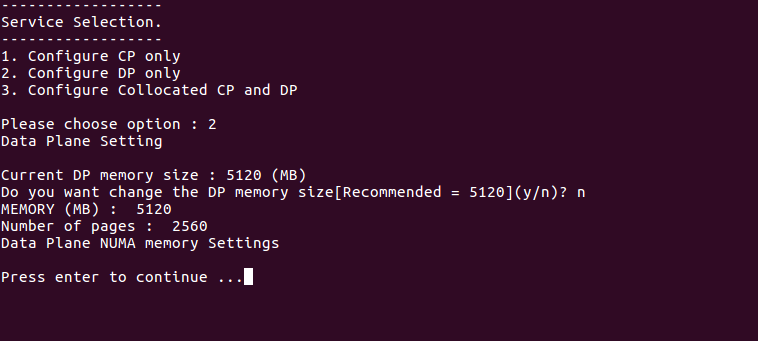
### Data Plane Installation

Follow the steps below to install the user plane.

1. Enter “1” to obtain information about network connectivity and OS.
2. Enter “2” to configure the data plane server.

After selecting option 2, the new selection menu will appear as shown in Figure 5. Enter “2” for the data plane.

Note: The configuration of the Collocated CP and DP Architecture is not supporting.



**Figure 7. Data Plane Setting Selection**

After entering “1” for control plane installation, the selection menu shown in Figure 6 will appear. Note that “n” has already been selected in the figure. If you want to change memory then enter “y”, otherwise enter “n”.

After entering “2” for data plane installation, the selection menu shown in Figure 7 will appear. Note that “n” has already been selected in the figure. If you want to change memory then enter “y”, otherwise enter “n”.

1. Enter “3” and agree to download dependent libraries or packages. Enter “y” after selecting option “3”.
2. Enter “4” to download packages.
3. Enter “5” to download and install the epctools.
4. Enter “6” to download and install the PFCP Protocol library.
5. Enter “7” to download the DPDK zip file.
6. Enter “8” to install the Data Plane Development Kit.
7. Enter “9” to download and install hyperscan packages.
8. Enter “10” to Build NGIC. Enter “y” to build NGIC in debug mode otherwise enter “n”.

This option builds and links all the libraries used to make the final binary.

## Building

The build options below help the user build the code manually without using *./install.sh* script every time. Once setup is complete as section 4, if the user made code changes then build steps can be followed.

1. Control plane

Path: *ngic-rtc/cp*

Command: *make clean; make;*

1. Data plane

Path: *ngic-rtc/dp*

Command: *make clean; make;*

1. Epctools

Path: *ngic-rtc/third\_party/epctools*

Command: *./configure; make; make install;*

Note: After this step, go to step 1 or 2.

1. Build libpfcp library

Path: *ngic-rtc/third\_party/libpfcp*

Command: *make clean; make;*

Note: After this step, go to step 1 or 2 according to the requirements of the server.

1. Build libgtpv2c library

Path: *ngic-rtc/third\_party/libgtpv2*

Command: *make clean; make;*

Note: After this step, go to step 1 or 2 according to the requirements of the server.

6) Build libepcadapter for CLI, DNS and ULPC

Path: ngic-rtc/oss\_adapter/libepcadapter

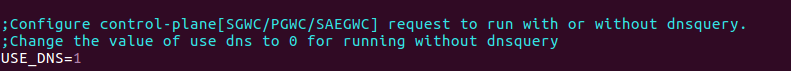
Command: *make clean; make; make install;*

Note: After this step, go to step 1 or 2 according to the requirements of the server.

### Runtime Flags

1. DNS Flag

Path: *ngic-rtc/config/cp.cfg*

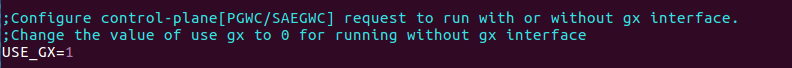


Function: It enables/disables the DNS functionality.

Flag Type: Run Time

1. GX Flag

Path: *ngic-rtc/config/cp.cfg*

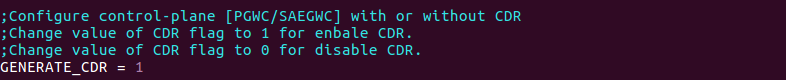


Function: It enables/disables the Gx functionality.

Flag Type: Run Time

1. CDR Flag

Path: *ngic-rtc/config/cp.cfg*

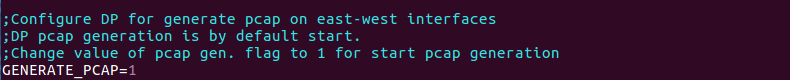


Function: It enables/disables the CDR functionality.

Flag Type: Run Time

1. Generate Pcap Flag

Path: *ngic-rtc/config/dp.cfg*



Function: It enables/disables the Pcap Generation functionality.

Flag Type: Run Time

Logs Path: *ngic-rtc/dp/logs*

## Dependencies

1. **Ubuntu 16.04 LTS** should be installed. The Ubuntu 16.04 LTS image can be downloaded from [Ubuntu Downloads](https://www.ubuntu.com/download/alternative-downloads).
2. The User account on the server must have **root privileges**.
3. Peer components MME, DNS, Redis, PCRF, eNB, and SGi-AS, or their simulators must be installed and configured.
4. The following packages must be installed by install.sh:

* DPDK version 18.02
* build-essential
* linux-headers-generic
* git
* unzip
* libpcap-dev
* make
* hyperscan
* curl
* openssl-dev
* freediameter
* Pistache
* rapidjson
* spdlog
* cpp-driver
* c-ares
* automake
* cmake
* libgcrypt-dev
* flex
* bison
* gnutls-dev
* libidn11-dev
* and any other library dependencies

## Configuration

### Editing Control plane Configuration

The parameters in the *ngic-rtc/config/cp.cfg* file for the control plane are shown in Table 4.

**Table 4. Control Plane Parameters**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Description** | **Value** |
| CP\_TYPE | This value is per gateway It defines type of deployment for the gateway | SGWC = 01, PGWC = 02, SAEGWC = 03 |
| S11\_IP/S11\_IP\_V6 | SGW-C/SAEGW-C S11 interface address | IPV4/IPV6 |
| S11\_PORT | GTPV2 port number | 2123 |
| S5S8\_IP/S5S8\_IP\_V6 | SGW-C s5s8 interface address if SGW and PGW-C s5s8 interface address if PGW | IPV4/IPV6 |
| S5S8\_PORT | GTPV2 port number | 2123 |
| PFCP\_IP/ PFCP\_IP\_V6 | SX interface address of SGW-C/SAEGW-C/PGW-C | IPV4/IPV6 |
| PFCP\_PORT | PFCP protocol port | 8805 |
| UPF\_PFCP\_IP/UPF\_PFCP\_IP\_V6 | Select User Plane path statically when DNS is disable | IPV4/IPV6 |
| UPF\_PFCP\_PORT | PFCP protocol port | 8805 |
| REDIS\_IP | Redis Server address | IPV4/IPV6 |
| REDIS\_PORT | Redis port | 6379 |
| CP\_REDIS\_IP | Redis Source address | IPV4/IPV6 |
| REDIS\_CERT\_PATH | Redis Certificate path | ngic-rtc/config/redis\_cert |
| DDF2\_IP | DDF2 Server address | IPV4/IPV6 |
| DDF2\_PORT | DDF2 port | 8888 |
| DDF2\_LOCAL\_IP | DDF2 source address | IPV4/IPV6 |
| DADMF\_IP | DADMF Server address | IPV4/IPV6 |
| DADMF\_PORT | DADMF port | 8881 |
| DADMF\_LOCAL\_IP | DADMF Source address | IPV4/IPV6 |
| SUGGESTED\_PKT\_COUNT | Select packet count for network service request | Default value for suggested pkts count for N/W service request |
| LOW\_LEVEL\_ARP\_PRIORITY | Select ARP priority for network service request | Default value for low level ARP priority for N/W service request |
| TRANSMIT\_TIMER | Check peer status if periodic timer expires. | Values in seconds |
| PERIODIC\_TIMER | Check peer status periodically. | Values in seconds |
| TRANSMIT\_COUNT | Number of times transmit timer sends a request. | Values in Integer |
| REQUEST\_TIMEOUT | Wait time for response from peer | Value in milliseconds 1 to 1800000 |
| REQUEST\_TRIES | Max Request Tries if peer is not responding | 1 to 20 |
| USE\_DNS | Enables/Disables DNS | 1 or 0 |
| CP\_DNS\_IP | DNS Source address | IPv4/IPV6 |
| CLI\_REST\_IP | CLI REST address for http requests | IPV4/IPV6 |
| CLI\_REST\_PORT | CLI REST port | Configurable |
| GENERATE\_CDR | Enables/Disables CDR | 1 or 0 |
| GENERATE\_SGW\_CDR | Activates SGW-C mode to check charging characteristics value from create session request | Value should be 3 to check charging characteristics from CSR. |
| SGW\_CC | SGW-C charging characteristics | Home/visiting/roaming [3,4,5] |
| ADD\_DEFAULT\_RULE | Add default rule if default bearer QOS is missing in CCA. | 0, 1, 2 |
| IP\_ALLOCATION\_MODE | IP address allocation mode | static/dynamic [0,1] |
| IP\_TYPE\_SUPPORTED | IP type supported selection settings | IPV4/IPV6/Priority based/dual mode [0,1,2,3] |
| IP\_TYPE\_PRIORITY | Select IP type priority | IPV4/IPv6 [0,1] |
| USE\_GX | Enables/Disables GX | 1 or 0 |
| APN | Add apn Configuration value | APN |
| URR\_DEFAULT | Set Usage Report Settings | Trigger Type/Volume Threshold for uplink and downlink/Time Threshold |
| IP\_POOL\_CONFIG | Add ip pool configuration for UE ip allocation | IPV4/IPV6 |
| APP nameserver | IP address of APP Server for DNS query | IPV4/IPV6 |
| OPS nameserver | IP address of OPS Server for DNS query | IPV4/IPV6 |
| PERF\_FLAG | To disable unnecessary operation | 1 or 0 |

### Editing Date Plane Configuration

The parameters in the *ngic-rtc/config/dp.cfg* file for the data plane are shown in Table 5.

**Table 5. Data Plane Parameters**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Description** | **Value** |
| PFCP\_IPv4/ PFCP\_IPv6 | User Plane SX interface address | IPV4/IPV6 |
| PFCP\_PORT | PFCP protocol port | 8805 |
| WB\_IFACE | West Bound interface | WBdev |
| EB\_IFACE | East Bound interface | EBdev |
| WB\_IPv4/ WB\_IPv6 | West Bound address | IPV4/IPV6 |
| WB\_IPv4\_MASK | West Bound mask | IPV4 |
| WB\_MAC | West Bound Mac address | Value of mac as per interface bind to dpdk drivers. |
| EB\_IPv4/ EB\_IPv6 | East Bound address | IPV4/IPV6 |
| EB\_IPv4\_MASK | East Bound mask | IPV4 |
| EB\_MAC | East Bound Mac address | Value of mac as per interface bind to dpdk drivers. |
| NUMA | Enables/Disables NUMA | 1 or 0 |
| TRANSMIT\_TIMER | Check peer status if periodic timer expires. | Values in seconds |
| PERIODIC\_TIMER | Check peer status periodically. | Values in seconds |
| TRANSMIT\_COUNT | Number of times transmit timer sends a request. | Values in Integer |
| TEIDRI | Teid Range Indicator | 0 to 7 |
| TEIDRI\_TIMEOUT | Teid Range Indicator timeout | Values in milliseconds |
| GENERATE\_PCAP | Enables /Disables Pcap generation on User Plane | 1 or 0 |
| CLI\_REST\_IP | CLI REST address for http requests | IPV4/IPV6 |
| CLI\_REST\_PORT | CLI REST port | Configurable |
| DDF2\_IP | DDF2 Server address | IPV4/IPV6 |
| DDF2\_PORT | DDF2 Port | 8869 |
| DDF2\_LOCAL\_IP | DDF2 Source address | IPV4/IPV6 |
| DDF3\_IP | DDF3 Server address | IPV4/IPV6 |
| DDF3\_PORT | DDF3 Port | 8870 |
| DDF3\_LOCAL\_IP | DDF3 Source address | IPV4/IPV6 |
| PERF\_FLAG | To disable unnecessary operation | 1 or 0 |

Note: You must bind the data plane interfaces to the dpdk driver before running the server. See [DPDK Guide](https://doc.dpdk.org/guides/tools/devbind.html) or Troubleshooting Section 7.1

### Editing Gx App Configuration

Edit the configuration files to run Gx applications.

1. As shown in Figure 8 below, edit the gx.conf file available at:

*ngic-rtc/cp/gx\_app/gx.conf*

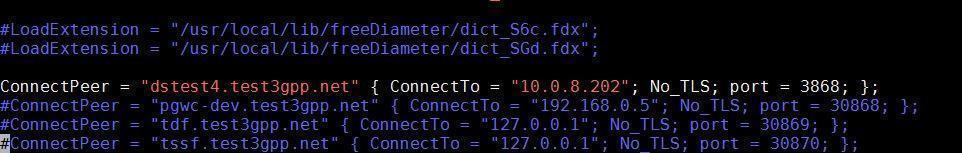
Hostname: pgwc6 (Modify in Identity,TLS\_Cred)

Realm: test3gpp.net (Identity, Realm)

****

**Figure 8. gx.conf File**

1. Modify the PCRF <Hostname>.<Realm> {connectTo= “PCRF IPV4/IPV6 ”; NO\_TLS,port = 3868} as shown in Figure 9

****

**Figure 9. gx.conf File**

1. As shown in Figure 10, generate the certificate file with command using

*./make\_certs.sh <hostname> <Realm>*

****

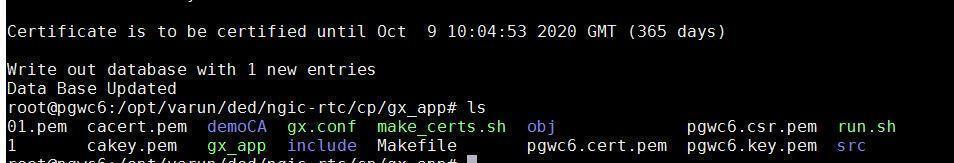
**Figure 10. Certificate Make Command**

The output will be:

<Hostname>.cert.pem

<Hostname>.csr.pem

<Hostname>.key.pem

****

**Figure 11. Certificate Make Console Result**

More information can be found at [Freediameter Configuration](http://www.freediameter.net/trac/wiki/Configuration).

### DNS server setup for UPF selection

#### Basic DNS Server Installation

Execute following steps on the host where DNS is to be installed.

1. Install the required DNS packages

*sudo apt-get install bind9 dnsutils bind9-doc*

1. Set bind by update the options parameter in */etc/default/bind9* as follows:

* For IPv4 mode

OPTIONS="-4 -u bind"

* For Ipv6 mode

OPTIONS="-6 -u bind"

* For IPv4 and Ipv6 mode

OPTIONS="-u bind"

1. Modify */etc/bind/named.conf*.options to define the DNS servers to forward unresolved DNS requests. For example, to forward unresolved DNS requests to Google, */etc/bind/named.conf.options* should look as follows:

options {

directory "/var/cache/bind";

// If there is a firewall between you and nameservers you want

// to talk to, you may need to fix the firewall to allow multiple

// ports to talk. See http://www.kb.cert.org/vuls/id/800113

// If your ISP provided one or more IP addresses for stable

// nameservers, you probably want to use them as forwarders.

// Uncomment the following block, and insert the addresses replacing

// the all-0's placeholder.

forwarders {

8.8.8.8;

};

//========================================================================

// If BIND logs error messages about the root key being expired,

// you will need to update your keys. See https://www.isc.org/bind-keys

//========================================================================

dnssec-validation auto;

auth-nxdomain no; # conform to RFC1035

listen-on-v6 { any; };

};

1. Configure Local File. Modify */etc/bind/named.conf.local* as follows:

* Add the forward zone with the following lines (substitute the zone name with your own):

zone "test3gpp.net" {

type master;

file "/etc/bind/db.test3gpp.net";

};

* Add the reverse zone by adding the following lines (note that the reverse zone name starts with 93.212.10 which is the opposite of 10.212.93.10).

zone "93.212.10in-addr.arpa" {

type master;

notify no;

file "/etc/bind/db.10";

};

1. Create the forward zone file using the file name referenced in */etc/bind/named.conf.local*. In this example the file is named */etc/bind/db.test3gpp.net* and the DNS server (ns1).

; BIND data file for local loopback interface

;

$TTL 604800

@ IN SOA ns1.test3gpp.net. root.ns1.test3gpp.net. (

2 ; Serial

604800 ; Refresh

86400 ; Retry

2419200 ; Expire

604800 ) ; Negative Cache TTL

;

@ IN NS ns1.test3gpp.net.

@ IN A 127.0.0.1

@ IN AAAA ::1

1. Create the reverse zone file using the file name referenced in etc*/bind/named.conf.local*. In this example the file is named */etc/bind/db.10*

;

; BIND reverse data file for local loopback interface

;

$TTL 604800

@ IN SOA ns1.test3gpp.net. root.ts1.test3gpp.net. (

1 ; Serial

604800 ; Refresh

86400 ; Retry

2419200 ; Expire

604800 ) ; Negative Cache TTL

;

@ IN NS ns1.test3gpp.net.

1. Start the DNS server.

*sudo service bind9 start*

1. Verify that the DNS service started successfully by opening /var/log/syslog and look for the following entry toward the end of the file:

Aug 10 13:20:29 ns1 named[29932]: all zones loaded

If this message cannot be located, locate the error message associated with the “named” service and make the necessary corrections.

#### Update /etc/hostname and /etc/hosts

1. Modify */etc/hostname* to reflect the unqualified host name of each server. In this example, the DNS server hostname should be:

ns1

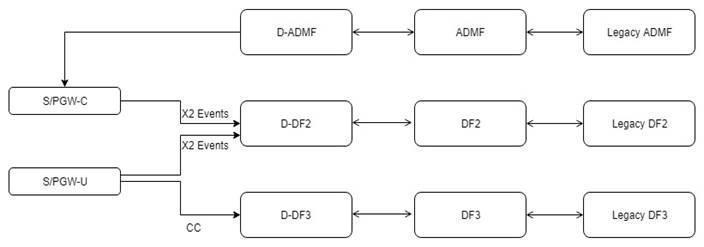
1. Ensure that */etc/hosts* has entries for 127.0.0.1, 127.0.1.1 and the IP address of the host with both the fully qualified hostname and the unqualified hostname.

127.0.0.1 localhost

127.0.1.1 ns1.test3gpp.net ns1

10.212.93.10 ns1.test3gpp.net ns1

### User level packet copying components setup



**Figure 12. User Level Packet Copying Components**

1. User level packet copying installation and configuration:

* Install the user level packet copying Servers

1. Run the install\_builddeps.sh from path *ngic-rtc/ulpc/*
2. Run the install\_build.sh from path *ngic-rtc/ulpc/*

Note : Install the ULPC components on different machines from control and user plane.

* Modify the DADMF configuration

Path: ngic-rtc/ulpc/d\_admf/config/d\_admf.json

"DAdmfApplication": {

"NodeName" : "dadmf",

"DAdmfIP" : "<DADMF server IPv4/IPv6>",

"DAdmfPort" : <DADMF port>,

"AdmfIP" : "<ADMF server IPv4/IPv6>",

"AdmfPort" : <ADMF port>,

"AckCheckTimeInMin" : 2

}

* Modify the ADMF configuration

Path:ngic-rtc/ulpc/admf/config/admf.json

"AdmfApplication": {

"Nodename" : "admf",

"AdmfIp" : "<ADMF server IPv4/IPv6>",

"AdmfPort" : <ADMF port>,

"DAdmfIp" : "<DADMF server IPv4/IPv6>",

"DAdmfPort" : <DADMF port>,

"LegacyInterfaceIp" : "<Legacy Interface server IPv4/IPv6>",

"LegacyAdmfIp" : "<Legacy ADMF server IPv4/IPv6>",

"LegacyAdmfPort" : <Legacy ADMF port>,

"LegacyAdmfIntfcPort" : <Legacy ADMF Interface port>,

"TransportProtocol" : "TCP"

}

* Modify the LEGACY ADMF configuration

Path:ngic-rtc/ulpc/legacy\_admf/config/legacy\_admf.json

"LegacyAdmfApp": {

"NodeName" : "legacy\_admf",

"ServerPort" : <Legacy ADMF port>,

"LegacyAdmfIntfcIp" : "<Legacy ADMF interface IPv4/IPv6>",

"LegacyAdmfIntfcPort" : <Legacy ADMF interface port>

}

* Modify the DDF configuration

### 

Path: ngic-rtc/ulpc/d\_df/config/d\_df.json

"DdfAppliction": {

"NodeName" : "<DDF2/DDF3>",

"DdfPort" : <DDF2/DDF3 port>,

"PcapFilePath" : "<Path on server to save pcaps>",

"DfIp" : "<DF2/DF3 server IPv4/IPv6>",

"DfPort" : <DF2/DF3 port>,

"StorageDirName" : "./db"

}

* Modify the DF configuration

Path: ngic-rtc/ulpc/df/config/df.json

"DfAppliction": {

"NodeName" : "<DF2/DF3>",

"DfPort" : DF2/DF3 port,

"StorageDirName" : "./db",

"LegacyCommMode" : "TCP",

"LegacyIp" : "<Legacy DF2/DF3 server IPv4/IPv6>",

"LegacyPort" : <Legacy DF2/DF3 port>

}

* Modify the LEGACY DF configuration

Path: ngic-rtc/ulpc/legacy\_df/config/legacy\_df.json

"DdfAppliction": {

"NodeName" : "Legacy DF",

"LegacyDfPort" : <Legacy DF2/DF3 port>,

"PcapFilePath" : "<Path on server to save pcaps>”

}

Note: Parameters marked in <> need to change as per requirement.

## Launching

Prerequisite for running the following component is that EPC setup is up and running with DNS, Redis and ULPC Servers as per configured in the Gateway configuration files.

1. Refer Redis Server configuration guide.

* [Graphene-SGX Installation](https://graphene.readthedocs.io/en/latest/building.html)
* [Build Redis on Graphene-SGX](https://github.com/oscarlab/graphene/blob/master/Examples/redis/README)

### Running Control Plane, Data Plane and Gx Applications

1. Control plane path: *ngic-rtc/cp*
2. Data plane path: *ngic-rtc/dp*
3. Gx app path: *ngic-rtc/cp/gx\_app*

Run the *run.sh* script in each path above to run all servers.

### Bring Up the Data Plane Path

Use the Kni script to bring up the data plane path.

Kni script path: - *ngic-rtc/kni\_ifcfg*

<kni-iface.sh> is used to bring up the data plane path for data packets.

### Bring Up User Level Packet Copying Components

1. Run *ngic-rtc/ulpc/legacy\_admf/run.sh* script.
2. Run *ngic-rtc/ulpc/d\_admf/run.sh* script.
3. Run *ngic-rtc/ulpc/admf/run.sh* script.
4. Run *ngic-rtc/ulpc/legacy\_df/run.sh* script.
5. Run *ngic-rtc/ulpc/df/run.sh* script.
6. Run *ngic-rtc/ulpc/d\_df/run.sh* script.

# Monitoring

The Control Plane and Data Plane can be controlled, monitored and tuned through the command line. Details are provided in the following sections.

## Using CLI

Run the CP or DP as SAEGW-C/U, SGW-C/U, PGW-C/U using *./run.sh* script in ngic-rtc folder.

Use the readme file to set c3pocli environment *ngic-rtc/oss\_adapter/cli/README.txt.*

Once CLI is set up with the above steps, the command *c3pocli* activates the CLI.

The c3pocli command can be used with several arguments as illustrated below:

1. c3pocli [http://127.0.0.1:12997](http://127.0.0.1:12997/) stats describe-stats-live

Gives live statistics of all peers in JSON format.

1. c3pocli http://127.0.0.1:12997 stats describe-stats-all

Gives live statistics of all peers in JSON format with all supported messages.

1. c3pocli http://127.0.0.1:12997 stats set-stats-reset

It reset the value of health parameters, last activity and messages stats.

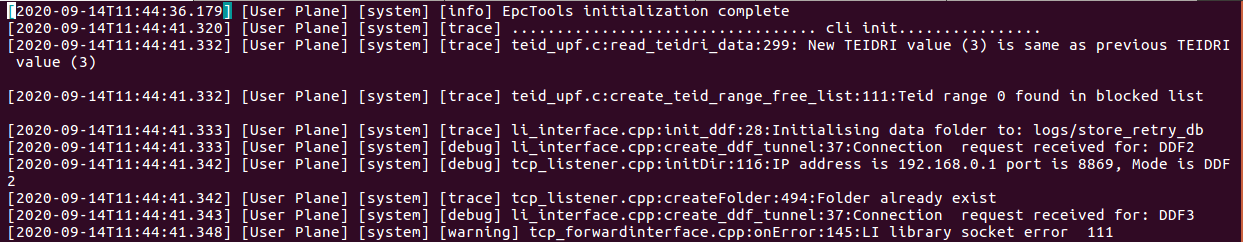
1. c3pocli http://127.0.0.1:12997 config describe-config-live

Displays the *ngic-rtc/config/cp.cf*g and *ngic-rtc/config/dp.cfg* configuration values in json format.

## Logging

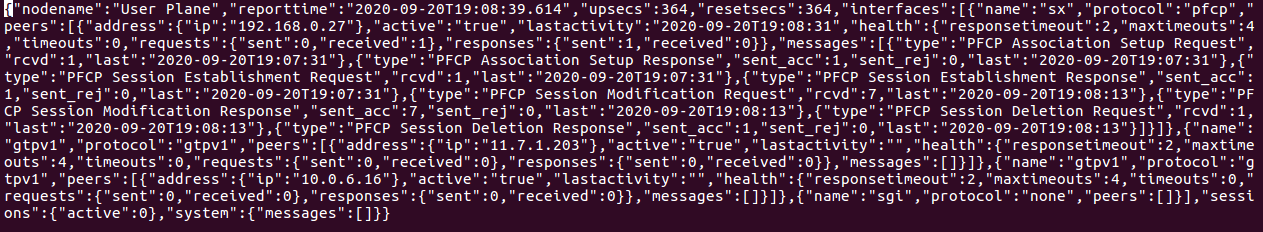
Logs are maintained in the following files under *ngic-rtc/cp/logs* folder or *ngic-rtc/dp/logs*.

1. Log messages are logged into the ngic.basic.log file as shown in Figure 13.



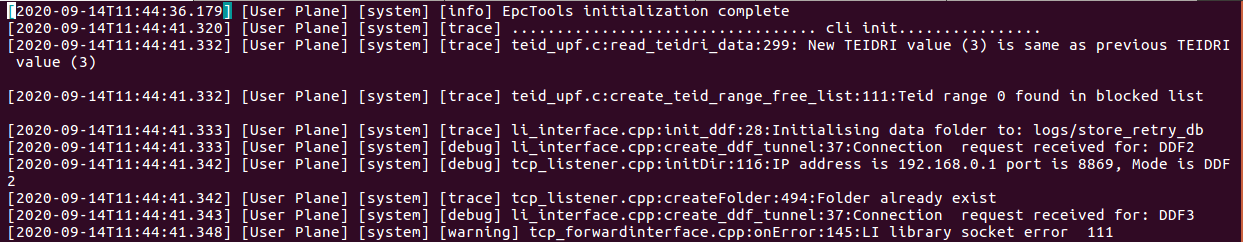
**Figure 13. ngic.basic.log file**

1. Stats after each 5 seconds (stat frequency set to 5 seconds by default) are logged into the ngic.stats.rotating.log file in JSON object format as shown in Figure 14.



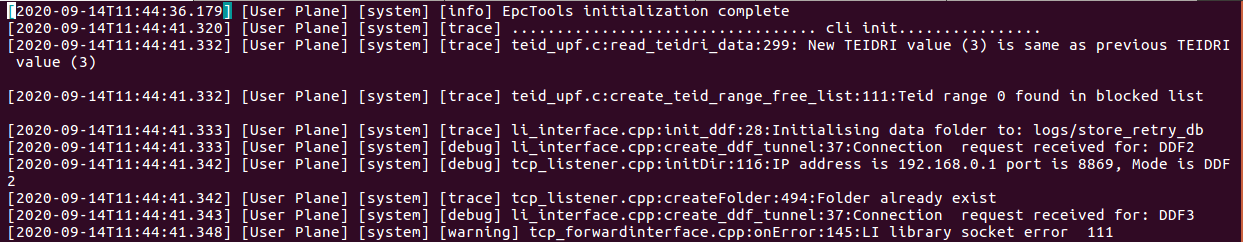
**Figure 14. ngic.stats.rotating.log File**

1. Log messages are logged into the ngic.daily\_2020-09-14.log as per date as shown in Figure 15.



**Figure 15. ngic.daily\_2020-09-14.log File**

1. Log messages are logged into the ngic.rotating.log and It stores rotating logs as shown in Figure 16.



**Figure 16. ngic.rotating.log File**

Note: Modify *ngic-rtc/config/log.json* file for logging initial settings

[Documentation reference for logging](https://brianwaters3.github.io/epctools/html/index.html#feature-overview-logging-configuration-json)

### c3pocli for Logging Configuration

1. c3pocli http://127.0.0.1:12997 stats set-stats-logging -n all

This command updates the stats logging mode in “all” level, Show all messages on each interface. Interfaces appear based on Gateway type. Ex. If SGW-C then “S11, S5S8 and Sxa” interface. By Default, “suppress” mode is configured.

1. c3pocli http://127.0.0.1:12997 stats set-stats-logging -n suppress

This command updates the stats logging mode to suppress mode for saving data in file. In “suppress”mode, we show messages which messages arrive on the interface (S11, S5S8 etc).

1. c3pocli http://127.0.0.1:12997 stats describe-stats-logging

Displays the stats logging mode to save data in the log file.

1. c3pocli [http://127.0.0.1:12997](http://127.0.0.1:12997/statfreq) stats describe-stats-frequency

This command shows the information about frequency of writing content in the ngic.stats.rotating.log file.

1. c3pocli [http://127.0.0.1:12997](http://127.0.0.1:12997/) stats set-stats-frequency -f 2000

This command sets the statistics frequency. The value is in milliseconds.

## DP pcap generation framework Commands

1. c3pocli http://127.0.0.1:12997 pcap describe-pcap-generation-status

Displays the information of the pcap generation framework status.

1. c3pocli http://127.0.0.1:12997 pcap set-pcap-generation -g start

It is used to start a pcap generation framework if it is in stop mode.

1. c3pocli http://127.0.0.1:12997 pcap set-pcap-generation -g stop

It is used to stop pcap generation framework.

1. c3pocli http://127.0.0.1:12997 pcap set-pcap-generation -g restart

It is used to restart pcap generation framework.

Note: (By default it is start in ngic-rtc/config/dp.cfg configuration)

## ULPC pcap generation commands

1. Add UE entry

curl -X POST -H "X-User-Name: YOUR\_NAME" -H "Content-Type: application/json" -d '{"uedatabase":[{"imsi":121014567892234,"signallingconfig":{"s11":2,"sgw-s5s8c":2,"pgw-s5s8c":2,"sx":[{"sxintfc":1, "type":3},{"sxintfc":2, "type":3},{"sxintfc":3, "type":3}]},"dataconfig":{"s1u\_content":2,"sgw\_s5s8u\_content":2,"pgw\_s5s8u\_content":2,"sgi\_content":2,"intfcconfig":[{"intfc":1, "direction":3},{"intfc":2, "direction":3},{"intfc":3, "direction":3},{"intfc":4, "direction":3}]},"forward":3,"timer":{"starttime":"2020-06-22T08:55:00Z","stoptime":"2020-06-25T09:33:00Z"}}]}' [http://](http://192.168.0.69)<D-ADMF IP>:<D-ADMF PORT>/addueentry

Above curl command can be used to add a new UE entry to perform ULPC. The command can be configured in different ways to enable copy of messages with or without header, downlink or uplink data and also for particular interfaces. This command returns a unique sequence ID which can be used further to update or delete the UE entry.

1. Update UE entry

curl -X POST -H "X-User-Name: YOUR\_NAME" -H "Content-Type: application/json" -d '{"uedatabase":[{"sequenceId":121014567548122,"imsi":121014567892234,"signallingconfig":{"sx":[{"sxintfc":3,"type":0}]},"dataconfig":{"s1u\_content":1,"sgi\_content":1,"intfcconfig":[{"intfc":1,"direction":1},{"intfc":4,"direction":0}]}}]}' [http://](http://192.168.0.69)<D\_ADMF IP>:<D\_ADMF PORT>/updateueentry

Above curl command can be used to update the configuration of the existing UE. Here sequence ID returned at the time adding the UE entry is used for identification purposes. Once the command is successful ULPC will get performed according to the new configurations.

1. Delete UE entry

curl -X POST -H "X-User-Name: YOUR\_NAME" -H "Content-Type: application/json" -d '{"uedatabase":[{"sequenceId":121014567890503,"imsi":121014567892234}]}' [http://](http://192.168.0.69)<D\_ADMF IP>:<D\_ADMF PORT>/deleteueentry

Above curl command can be used to remove the existing UE from performing ULPC. Here sequence ID returned at the time adding the UE entry is used for identification purposes. Once the command is successful the UE will no longer be performing ULPC.

# Troubleshooting

## Binding PCI Interface to dpdk Driver

Bind the S1u/Sgi port to DPDK drivers

1. Command to get PCI address is:

|  |
| --- |
| root@spgwu:/opt/ngic-rtc/third\_party/dpdk/usertools# lshw -c network -businfo  Bus info Device Class Description  ========================================================  pci@0000:00:03.0 ens3 network Virtio network device  pci@0000:00:04.0 ens4 network Virtio network device  pci@0000:00:08.0 ens5 network 82599ES 10-Gigabit SFI/SFP+ Network Connection  pci@0000:00:09.0 ens6 network 82599ES 10-Gigabit SFI/SFP+ Network Connection |

1. Bind the port using the PCI id

|  |
| --- |
| cd /opt/ngic-rtc/third\_party/dpdk/usertools/  ./dpdk-devbind.py -b igb\_uio 00:08.0 |

1. Lists ports

|  |
| --- |
| root@spgwu:/opt/ngic-rtc/dpdk/usertools# ./dpdk-devbind.py --status  Network devices using DPDK-compatible driver  ============================================  0000:00:08.0 '82599ES 10-Gigabit SFI/SFP+ Network Connection 10fb' drv=igb\_uio unused=ixgbe  Network devices using kernel driver  ===================================  0000:00:03.0 'Virtio network device 1000' if=ens3 drv=virtio-pci unused=igb\_uio \*Active\*  0000:00:04.0 'Virtio network device 1000' if=ens4 drv=virtio-pci unused=igb\_uio \*Active\*  0000:00:09.0 '82599ES 10-Gigabit SFI/SFP+ Network Connection 10fb' if=ens6 drv= ixgbe unused=igb\_uio |

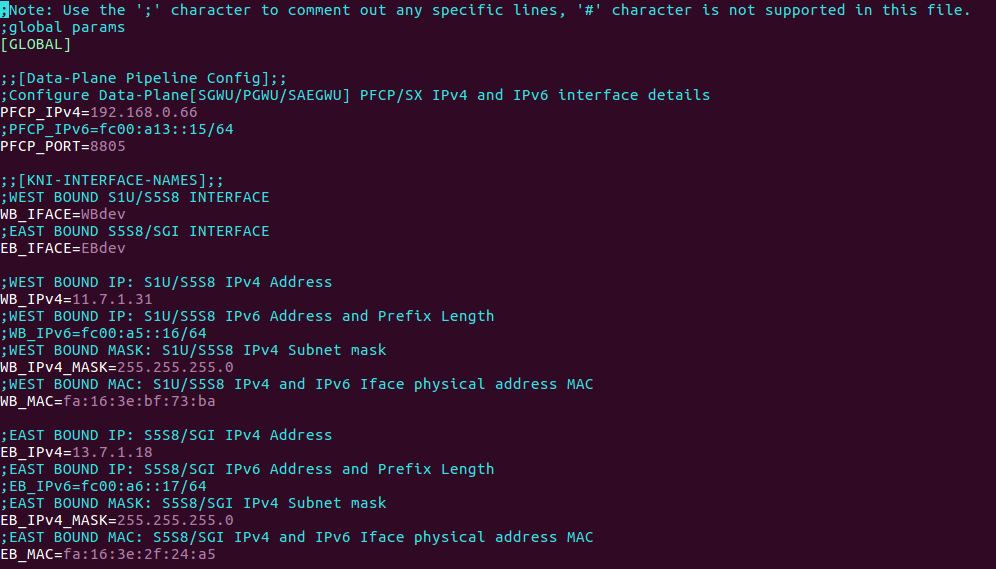
## Setting Environment for Control and User Planes

Path: cd *ngic-rtc/*

Command: *source setenv.sh*

## Data Not Passing Through User Plane

1. Check PGW-U/SAEGW-U/SGW-U for *ngic-rtc/config/dp.cfg* configuration file. See Figure 17.
2. Check the MAC address and IP Configuration for west and east bound.



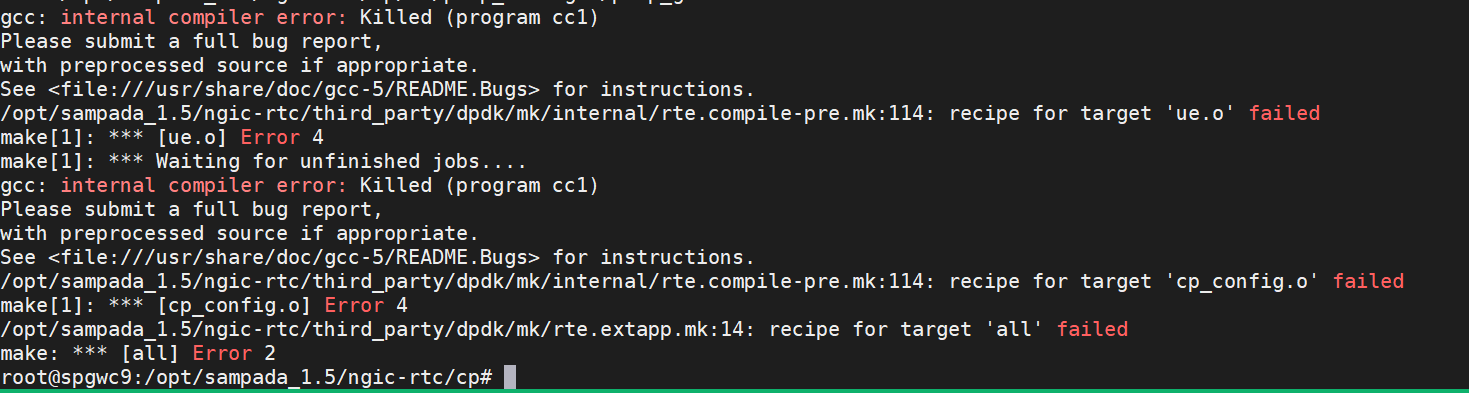
**Figure 17. dp.cfg File**

## Giving Permission to Shell Script File

Command: *chmod +x < filename >*

## Memory Issue in Section 5.3 Control/User Plane build

If memory issues arise during the control plane/user plane build described in Section 5.3 (See Figure 18), perform the steps below.

****

**Figure 18. Memory Issue**

1. Go to *ngic-rtc/*
2. Run the *source setenv.sh* command
3. *make -j5 build-cp or make -j5 build-dp*

# References

1. 3GPP specifications release 15.
   1. TS 23.401
   2. TS 23.214
   3. TS 23.272
   4. TS 23.003
   5. TS 23.007
   6. TS 29.212
   7. TS 29.244
   8. TS 29.303
   9. TS 29.212
   10. TS 29.213
2. [Open Networking Foundation](https://www.opennetworking.org/)