



**UNIVERSITAT POLITÈCNICA DE CATALUNYA  
BARCELONATECH**

---

**Escola Tècnica Superior d'Enginyeria  
de Telecomunicació de Barcelona**

**STUDY OF RAN SLICING IN A REAL SDN-NFV  
PLATFORM**

**A Master's Thesis**

**Submitted to the Faculty of the  
Escola Tècnica d'Enginyeria de Telecomunicació de  
Barcelona**

**Universitat Politècnica de Catalunya**

**by**

**Ester Muñoz Sánchez**

**In partial fulfilment  
of the requirements for the degree of  
MASTER IN TELECOMMUNICATIONS ENGINEERING**

**Advisor: Anna Umbert Juliana**

**Barcelona, February 2018**



UNIVERSITAT POLITÈCNICA  
DE CATALUNYA  
BARCELONATECH



**Title of the thesis:** Study of RAN slicing in a real SDN-NFV platform

**Author:** Ester Muñoz Sánchez

**Advisor:** Anna Umbert Juliana

## **Abstract**

The continuous increase of traffic demand, in the future, could congest and overflow current networks. New virtualization technologies that are capable of developing more heterogeneous, flexible and manageable networks, in which operating costs (OPEX) and investment (CAPEX) are reduced are being researched to fix this issue. Software Defined Networking (SDN) and Network Function Virtualization (NFV) are two virtualization technologies that provide network flexibility and better resource management, allowing resources to be dynamically shared and slicing the Radio Access Network (RAN). In this project, the study and implementation of RAN Slicing on a testbed based on the OpenAirInterface and 5G-EmPOWER platforms is carried out. For its implementation, the 2 platforms have been studied, configured and integrated and the RAN Slicing has been applied, with the creation of tenants. The study of RAN Slicing for Multi Operator Core Network (MOCN) has also been developed, with the creation of a new operator to analyse the connectivity of multiple operators to the same eNB. Throughout the project, several studies have been developed to verify the functioning of the network and the connectivity of one or multiple users to it.

## **Acknowledgements**

This project would not have been possible without the help and support of various people. Especially, I want to thank my director Anna Umbert Juliana for having offered me such an enriching project that has allowed me to discover a small part of the work that is done within the GRCM group. I want to thank her for all the dedication and the great help she has given me both in the project and at the time of writing the thesis, providing me with new knowledge, guiding me and resolving doubts whenever I have needed it. I would also like to thank Ali Esmaeily, who has accompanied me throughout the project process and has contributed favorably to its development; to Katerina Koutlia, for all the help she has provided me and for the interest she has shown in the project; and Sergio García, for the technical support. Finally, I want to thank my family and friends for their support and patience over all these years and for always trust in me.

## Revision history and approval record

Revision	Date	Purpose
0	22/12/2017	Document creation
1	26/01/2018	Document revision
2	04/02/2018	Document correction

Written by:		Reviewed and approved by:	
Date	26/01/2018	Date	04/02/2018
Name	Ester Muñoz Sánchez	Name	Anna Umbert Juliana
Position	Project Author	Position	Project Supervisor

## **Table of contents**

Abstract.....	1
Acknowledgements .....	2
Revision history and approval record.....	3
Table of contents .....	4
List of Figures .....	6
List of Tables .....	10
1. Introduction.....	11
2. State of the art.....	13
2.1. Software Defined Networking (SDN).....	13
2.2. Network Virtualization Function (NFV) .....	14
2.3. Long Term Evolution (LTE) .....	16
2.4. OpenAirInterface platform .....	20
2.5. 5G-EmPOWER platform.....	23
3. Methodology.....	26
4. Development and achieved results .....	30
4.1. Presentation of the workspace .....	30
4.2. OAI platform: Configuration and terminals connection .....	32
4.2.1. Network architectures.....	32
4.2.2. General network parameters.....	34
4.2.3. Network configuration for 2 and 1 PC .....	36
4.2.4. eNB configuration .....	45
4.2.5. Launch of the OAI platform .....	46
4.2.6. Connectivity tests .....	47
4.3. 5G-EmPOWER platform.....	50
4.3.1. 5G-EmPOWER web interface.....	51
4.4. Integration of OAI and 5G-EmPOWER platforms.....	54
4.5. Implementation of RAN Slicing and users connection.....	57
4.6. Implementation of Multiple Operator Core Network (MOCN) .....	63
4.6.1. Creation of a new operator.....	63
4.6.2. Network architecture .....	65
4.6.3. Configuration of the new EPC.....	68

4.6.4.	Creation of a new tenant .....	71
4.6.5.	Connectivity tests of the new operator.....	71
4.6.6.	Connectivity tests with 2 EPCs and the eNB .....	73
5.	Conclusions and future development .....	75
	Bibliography.....	77
	Annex 1. Configuration of the OAI platform .....	79
	Annex 2. Manufacturer SIM parameters .....	103
	Annex 3. Configuration of database.....	103
	Annex 4. Launching the network.....	104
	Glossary.....	106

## **List of Figures**

Figure 1. Architecture of a traditional network and a SDN network. Extracted from [3]. ...	13
Figure 2. Architecture of SDN approach. Extracted from [5]. .....	14
Figure 3. Network Virtualization approach. Extracted from [6]. .....	15
Figure 4. Relationship diagram between SDN and NFV. Extracted from [8]. .....	16
Figure 5. Overall LTE architecture. Extracted from [9]. .....	17
Figure 6. Detailed E-UTRAN (left) and EPC (right) architectures. Extracted from [9]. ....	17
Figure 7. OAI protocol stack scheme. Extracted from [17]. .....	20
Figure 8. Overall OAI platform architecture. Extracted from [19]. .....	21
Figure 9. USRP B210, Ettus Research. Extracted from [20]. .....	22
Figure 10. Complete architecture of the OAI platform. ....	23
Figure 11. EmPOWER network architecture. Extracted from [21]. .....	24
Figure 12. 5G-EmPOWER block diagram. ....	25
Figure 13. Overall architecture of the 5G-EmPOWER and OAI LTE network. ....	26
Figure 14. OAI LTE network configurations. ....	26
Figure 15. Integration of 5G-EmPOWER and OAI platforms. ....	27
Figure 16. Final network with RAN Slicing implemented. ....	28
Figure 17. Final network with Multi Operator Core Networks (MOCN). ....	29
Figure 18. Project workspace. ....	30
Figure 19. Scheme of the OAI LTE network configurations. ....	31
Figure 20 . User Equipments (UEs). ....	31
Figure 21. Sysmocom SimCard. Extracted from [26]. ....	32
Figure 22. Configuration of the OAI LTE network for 2 PCs. ....	32
Figure 23. Configuration of the OAI LTE network for 1 PC. ....	33
Figure 24. PLMN Id definition. ....	35
Figure 25. GUMMEI definition. ....	35
Figure 26. TAI List definition. ....	35
Figure 27. Representation of TAs. Extracted from [28]. ....	36
Figure 28. Configuration files OAI platform. ....	36
Figure 29. Configuration of hss.conf for 1 and 2 PCs configurations. ....	37
Figure 30. Configuration parameters of hss_fd.conf for 1 and 2 PCs configurations. ....	38
Figure 31. Configuration of acl.conf for 1 and 2 PCs configurations. ....	38
Figure 32. MySQL console interface. ....	39



Figure 33. phpMyAdmin database web interface. ....	40
Figure 34. Configuration of users table.....	41
Figure 35. Configuration of pdn table. ....	42
Figure 36. Configuration of mmeidentity table for 2 PCs configuration.....	42
Figure 37. Configuration of mmeidentity table for 1 PC configuration. ....	42
Figure 38. Configuration of GUMMEI and TAI List for 1 and 2 PCs configurations.....	42
Figure 39. Configuration of network interfaces for 1 PC configuration. ....	43
Figure 40. Configuration of network interfaces for 2 PCs configuration. ....	43
Figure 41. Configuration of mme_fd.conf for 1 PC configuration. ....	43
Figure 42. Configuration of mme_fd.conf for 2 PCs configuration. ....	43
Figure 43. Configuration of spgw.conf for 1 PC configuration.....	44
Figure 44. Configuration of spgw.conf for 2 PCs configuration.....	44
Figure 45. Configuration of network parameters for 1 and 2 PCs configuration.....	45
Figure 46. Configuration of MME IP address and network interfaces for 1 PC configuration.....	45
Figure 47. Configuration of MME IP address and network interfaces for 2 PCs configuration.....	45
Figure 48. Launching the OAI LTE network in 2 PCs configuration. HSS (top left), MME (top right), SPGW (bottom left) and eNB (bottom right). ....	46
Figure 49. Launching the OAI LTE network in 1 PC configuration. HSS (top left), MME (top right), SPGW (bottom left) and eNB (bottom right). ....	47
Figure 50. eNB is ready for users connection for 1 and 2 PCs configuration. ....	47
Figure 51. RFBENCHMARK: Availability of 21491 network.....	48
Figure 52. Qualipoc: Analysis of the 21491 network. ....	48
Figure 53. Configuration of APN in Samsung Galaxy SIII and Telenor Mobile Partner program. ....	49
Figure 54. SIM card 214910000009916 introduced in Dongle is detected in HSS as a client. ....	49
Figure 55. Authentication and authorization messages in eNB. ....	49
Figure 56. Scheme of the 5G-EmPOWER complete system.....	50
Figure 57. Initialization of the controller. ....	51
Figure 58. 5G-EmPOWER web interface. ....	51
Figure 59. Main window login as administrator. Tenants tab. ....	52
Figure 60. Requests tab.....	52
Figure 61. VBSes tab. ....	52
Figure 62. UEs tab. ....	53

Figure 63. IMSI to MAC address mapping tab. ....	53
Figure 64. Users tab.....	53
Figure 65. Main window login as regular user. Tenants tab.....	53
Figure 66. Tab to create of new tenants. ....	54
Figure 67. Pending requests tab. ....	54
Figure 68. OAI and 5G-EmPOWER platforms before to be integrated.....	54
Figure 69. Initialization controller messages to identify VBSP port.....	55
Figure 70. Integration of the OAI and 5G-EmPOWER platforms. ....	56
Figure 71. Creating VBS. ....	56
Figure 72. eNB is associated to the controller.....	57
Figure 73. Creation of a new tenant.....	58
Figure 74. Accepting the new tenant. ....	58
Figure 75. The new tenant is active. ....	58
Figure 76. Association of VBS with the new tenant.....	59
Figure 77. IMSI to MAC address mapping.....	59
Figure 78. eNB ready to connect users. ....	60
Figure 79. UE authentication and connection.....	60
Figure 80. VBS detected by the controller. ....	60
Figure 81. Controller waiting for UE connection. ....	61
Figure 82. UE (214910000009916) detected by the controller. ....	61
Figure 83. Join messages of UE to the tenant. ....	61
Figure 84. HSS identifies the UE connected: 214910000009916.....	61
Figure 85. UE (214910000009916) associated to the tenant. ....	61
Figure 86. Authentication and connection of 2 UEs. ....	62
Figure 87. Identification of 2 UEs, 214910000009915 and 214910000009916, in the HSS. .....	62
Figure 88. Detection and association of 2 UEs, 214910000009915 and 214910000009916. ....	63
Figure 89. PLMN Id definition.....	64
Figure 90. GUMMEI definition.....	64
Figure 91. TAI List definition. ....	64
Figure 92. Overall architecture with MOCN. ....	65
Figure 93. Configuration of mme.conf for 21491 network. ....	66
Figure 94. Configuration of mme_fd.conf for 21491 network. ....	67
Figure 95. Configuration of spgw.conf for 21491 network.....	67

Figure 96. Changes in eNB configuration file to connect with new EPC. ....	67
Figure 97. Changes in eNB configuration file to connect with 2 EPCs. ....	68
Figure 98. Configuration of users table.....	69
Figure 99. Configuration of pdn table. ....	69
Figure 100. Configuration of mmeidentity table.....	69
Figure 101. Configuration of hss.conf for the 21412 network. ....	69
Figure 102. Configuration of mme.conf for the 21412 network. ....	70
Figure 103. Configuration of mme_fd.conf for the 21412 network. ....	70
Figure 104. Configuration of spgw.conf for 21412 network.....	71
Figure 105. Creation of a new tenant and VBS association.....	71
Figure 106. Controller establish connection with eNB.....	72
Figure 107. VBS is active.....	72
Figure 108. eNB is ready for the UE connection. ....	72
Figure 109. HSS has identified the UE connected. ....	72
Figure 110. UE authentication and connection.....	73
Figure 111. Controller detects de UE and associates with the new tenant. ....	73
Figure 112. UE associated to the new tenant.....	73
Figure 113. MME from 21491 is considered none existing. ....	74
Figure 114. Error in MME of 21491 network: No common PLMN Id with eNB. ....	74
Figure 115. Manufacturer SIM card parameters.....	103
Figure 116. MySQL interface. ....	103

## **List of Tables**

Table 1. Interfaces of the E-UTRAN. ....	18
Table 2. Components of the Core Network. ....	19
Table 3. Characteristics of the physical layer, E-UTRAN and Core Network. ....	21
Table 4. Architecture of the 5G-EmPOWER platform. ....	25
Table 5. Allocation of IP addresses for 2 PCs configuration. ....	33
Table 6. Allocation of IP addresses for 1 PC configuration. ....	34
Table 7. Assignment of MNC codes by operator in Spain. Extracted from [27]. ....	34
Table 8. Configuration parameters hss.conf. ....	37
Table 9. Configuration parameters of hss_fd.conf. ....	38
Table 10. Configuration parameter of acl.conf ....	39
Table 11. Configuration parameters of SIM cards. ....	40
Table 12. Configuration parameter of new SIM card. ....	65
Table 13. Allocation of IP addresses for 21491 and 21412 networks. ....	66

## 1. Introduction

In the last years, the data traffic in the wireless networks has increased notably due to the generalised use of Internet and mobile applications. Thanks to the development of new communication standards, such as Long Term Evolution (LTE) and Long Term Evolution Advanced (LTE-A), this levels of traffic have been easily managed contributing to the communication with 230% more efficiency, faster data transmission rates and better quality than older standards like Global System for Mobile communications (GSM) or Universal Mobile Telecommunications System (UMTS). Nevertheless, a higher global traffic demand is expected by 2020, which will reach the 30.6 exabytes monthly [1]. This issue is worrying manufacturers and operators of wireless networks since the increase in the traffic demand will originate congestion and consequently overflow of the fourth generation networks. Moreover, this increase might cause the rise in the equipment inversion and operation costs. Thus, new approaches and technologies are being studied in order to deploy denser and heterogeneous networks, which will be able to provide a flexible management and an efficient operation while maintaining the Capital (CAPEX) and Operating (OPEX) Expenditures low.

One of these new approaches is the virtualization that permits decoupling the software applications from the underlying hardware. Software Defined Networks (SDN) and Network Function Virtualization (NFV) are two virtualization technologies that together can provide flexibility in the network and management of the resources. These features allow sharing dynamically the resources and slicing the Radio Access Network (RAN) basing on the requested specifications.

Nowadays, some research groups in mobile communications area are investigating in this field. In particular, in the UPC the Mobile Communications Research Group (GRCM) is collaborating with the Future Networks (FuN) research Unit, which belongs to the Fondazione Bruno Kessler Create-Net (FBK.Create-Net) international research center in Trento, in the study of the RAN Slicing concept on a real-time test platform based on the 5G-EmPOWER. This platform implements Wi-Fi and LTE networks following the SDN paradigm and using virtualization functionalities.

RAN Slicing [2] can deal with the continuous increasing of the traffic demand and with the heterogeneous patterns of traffic in the wireless networks. In fact, an important challenge is that each Mobile Virtual Network Operator (MVNO) or tenant can have different requests in a specific node (Wi-Fi Access Point or LTE eNodeB, among others) according to the traffic variations in the network. Thus, slicing the RAN can reduce the costs in the deployment and operation of several logical networks over a common physical network infrastructure in such a way that each network could be personalized to accomplish with the necessities of specific applications and/or communications service providers. Besides, each tenant can manage its own slice of resources while it is providing the desired set of services. Nevertheless, slicing the RAN can be a bit difficult to implement because of the inherently shared nature of the radio channel and the considerable influence of any transmitter over any receiver.

Up to now both groups were working together in the RAN Slicing for the Wi-Fi standard, in which the slicing have been successfully achieved, but for some time now they started to study the RAN Slicing for the LTE standard in which this project is enrolled.

The main objective of this project is the study and implementation of RAN Slicing on a real-time platform based on the 5G-EmPOWER software, designed by the research center FDK-Create-Net, for its later integration with the OpenAirInterface (OAI) LTE network, provided by the OpenAirInterface Software Alliance (OSA). With this study we want to contribute to the investigation of new technologies that allow to evolve to 5th generation networks in which more complex and flexible networks are sought, which will be capable of handling very high levels of traffic and providing a better quality of service according to the service that are providing and the users requirements.

The thesis consists of the following sections. In Section 2, the context of the project will be defined by the explanation of several concepts such as the SDN paradigm, NFV, the LTE standard, the OpenAirInterface platform and the EmPOWER testbed which will clarify the bases of this project. In Section 3, the methodology followed to carry out the work will be briefly explained. In Section 4, all the procedures developed to reach the main objective and the results obtained will be exposed. Finally in Section 5, the conclusions of the project and the future work will be presented.

## 2. State of the art

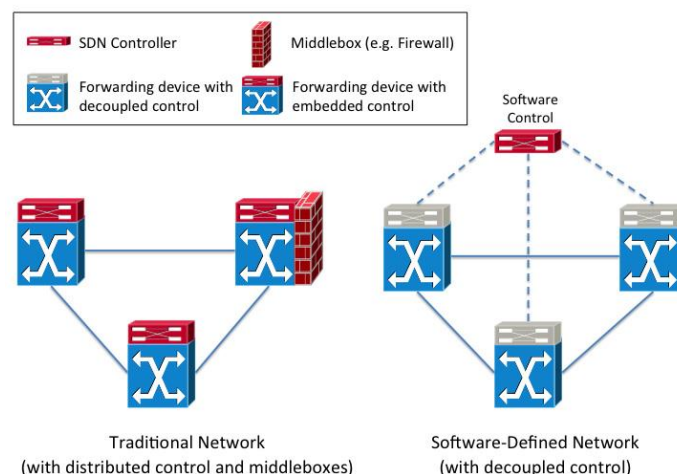
In this section, the context in which the project is carried out is presented. First, the different technologies and standards (such as SDN, NFV and LTE) that fix the basis of this work will be described. Then, the platforms OAI LTE network and 5G-EmPOWER that have been used to develop the project will be introduced.

### 2.1. Software Defined Networking (SDN)

Nowadays, data communications networks are usually made up of a high number of forwarding devices such as routers, switches, firewalls, among others. The interconnection of them, by links, allows the transmission of data between different hosts or end users, which are also connected to that network. These network devices tend to be closed systems so that there is very limited control of their interfaces. In addition, they do not allow an innovation of the network infrastructure, since neither existing protocols can be modified nor can new ones be created to adapt the network to the needs that the users request.

A major drawback that exists in current networks and that does not allow their evolution is that control and data planes are coupled so that each device in the network makes its own forwarding decisions. This coupling between planes makes more complicated to develop new applications and functionalities in the elements of the network and any change that may be applied have to be configured directly on each element of the infrastructure.

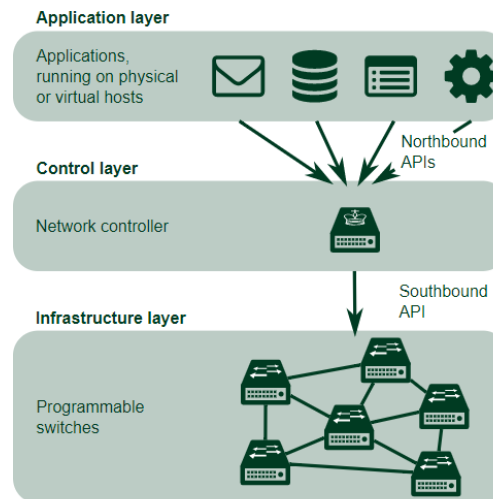
Software Defined Networking [3] has been developed with the aim of facilitating the innovation of the network and therefore of providing a better and easier control of the network data-path. Moreover, this new paradigm separates the control and data planes, that is, the physical infrastructure from the logic control, by providing the network with a common software control. This separation of planes provides direct virtualization and a major management of the network and easier implementation of new protocols and applications. In this way, the network is reduced to forwarding devices and a controller that will make the forwarding decisions of the different elements of the network.



**Figure 1. Architecture of a traditional network and a SDN network. Extracted from [3].**

In Figure 1 the evolution from a current network to a SDN network is shown, in which the forwarding hardware is separated from the control logic. In the SDN network the data-path is represented with solid lines and the control-path is represented with dashed lines.

The architecture of the SDN [4] approach is exposed in Figure 2.



**Figure 2. Architecture of SDN approach. Extracted from [5].**

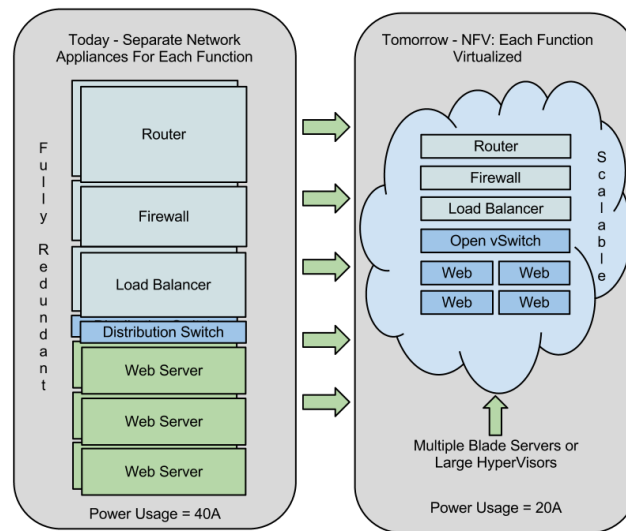
The SDN architecture is composed by the following components:

- **Applications:** They are programs that inform the controller about their network needs and behaviours through one or more Northbound interface.
- **Controller:** It is a logical control entity able to translating the requests from the applications to the underlying data plane, giving to the application plane an abstract vision of the network by means of statistics and events.
- **Programmable switches:** They are logical elements of the network that expose visibility and control over its forwarding and processing capacities. They communicate with the controller through a Southbound interface.

## 2.2. Network Virtualization Function (NFV)

Creating new services in current networks is a bit complicated since the existing network infrastructures have hardware and software that are very limited and are integrated in a specific way, which means that they are not very flexible. In order to integrate a new service there must be an inversion in new hardware and software which would lead to an increase in deployment and operation costs (CAPEX and OPEX). The evolution of the networks is fundamental to avoid these issues by developing new technologies that allow having more flexibility and management networks, less dependency on the manufacturers and use open standards, while the CAPEX and OPEX are reduced. Network Function Virtualization concept has emerged to alleviate the previous problems.





**Figure 3. Network Virtualization approach. Extracted from [6].**

Network Function Virtualization [7] is an IT virtualization technology that aims to migrate network node functions from a specific hardware to software instances which run on a general purpose virtualized networking. Moreover, it allows that the programmed software will be implanted in different platforms allowing multiple management points in the networks. From this idea arise a new one in which is intended to standardize and make compatible these management points for multi-vendor solutions. There are three principal features that characterize the virtualized networks:

- **Decoupling hardware and software.** The separation between hardware and software allows that each block evolves independently from each other.
- **Flexible deployment of network functions.** NFV can automatically implement the software of network function on a set of hardware resources which can execute different functions at different times in different datacenters.
- **Dynamic service provisioning.** Network operators can scale dynamically the NFV performance and grow as needed, having a precise granularity control based on the actual status of the network.

The implantation of NFV is done from the virtualization of network function elements, which are named Virtualized Network Functions (VNF). Each VNF is developing a network function and combining them the virtualized network segment will be created. There are three types of devices that can be virtualized: network function devices (routers, switches, Access Point Names (APN),...), IT devices related to the network (firewalls, network device management systems,...) and network-linked storage (file servers or databases).

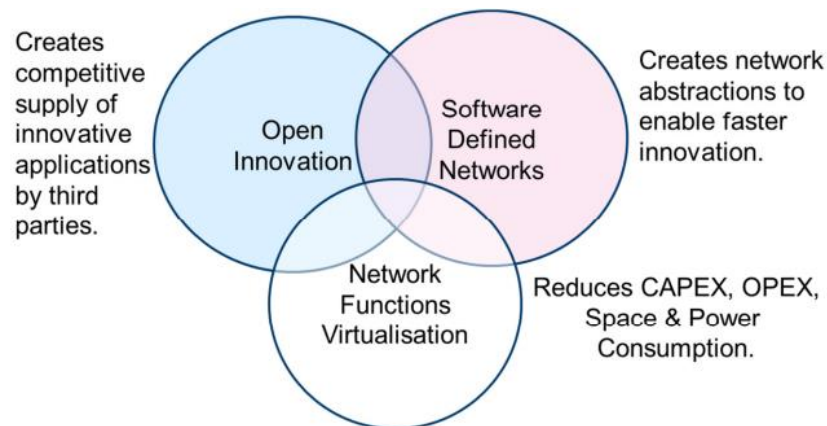
From the NFV implantation can be obtained many benefits [8], as are explained below:

- Simplification on the implantation of network elements.
- Increase of the speed in the deployment of the network elements.
- Higher network scalability.
- Independence of equipment manufacturers.
- Interoperability. Compatibility between several elements of the network.
- Wide variety of ecosystems and open standards.

- Security.
- Less Time-To-Market.
- Less inversion (CAPEX) and operation costs (OPEX).

### **Relationship between NFV and SDN approaches**

NFV is closely related to other emerging virtualization technologies such as the SDN [8]. Although both can be developed independently, without depending on each other, together they complement very well and can be obtained more powerful solutions because in both technologies the orchestration of the set of services is fundamental. Moreover, the scheme of management and monitoring tools of SDN for the different planes agrees with the needs of NFV, which implies that management solutions between SDN and NFV are ideals to manage multiprovider environments. On the other hand, both two technologies pursue the same objectives: innovation, creation, openness and competitiveness.



**Figure 4. Relationship diagram between SDN and NFV. Extracted from [8].**

In conclusion, NFV is compatible with SDN paradigm. The former can provide to the SDN the infrastructure upon which SDN software can run, reducing CAPEX, OPEX, space and power consumption while the SDN provides a faster innovation due to its network abstractions in the control and data planes.

### **2.3. Long Term Evolution (LTE)**

In the last two decades the demand for traffic in mobile communication networks has grown considerably due to the entry into the market of smartphones that have led to increased use of the Internet and mobile applications. Whereas before the users were interested on voice services and small text messages, now users tend to make more use of high-quality real-time audiovisual services. Given these circumstances, communication standards have had to be adapted to meet the technological and user needs. Currently, due to the existing level of traffic, standards such as GSM (2nd generation) or UMTS (3rd generation) are very limited so more and more operators are getting by the implementation of the LTE standard (4th generation).

In the following, the architecture of LTE standard is presented.



**Figure 6. Detailed E-UTRAN (left) and EPC (right) architectures. Extracted from [9].**

17

<b>E-UTRAN Uu</b>	It allows the information transfer by the radio channel between the eNBs and the UEs. The eNB implements all the functions and protocols to exchange information through the radio channel and controlling the operation of the E-UTRAN interface.
<b>S1</b>	This interface allows the communication between the UEs and the EPC. It is divided in two different interfaces: S1-MME (S1-C) and S1-U. This separation between interfaces allows the separation between the control and data planes and, consequently, the separation between their protocols in two different stacks. Whereas, the S1-MME is in charge of supporting the functions and procedures to manage the performance of this interface, the S1-U is in charge of controlling the exchange of data in this interface. This plane separation permits the communication between the eNB and two different entities MME (control plane) and P-GW (data plane).
<b>X2</b>	It is an optional interface that connects the different eNBs of E-UTRAN and allows the control and user information transfer between them.

**Table 1. Interfaces of the E-UTRAN.**

EPC [9] constitutes the Core network and it has been mainly designed to provide a service of IP connectivity thanks to its optimized architecture that exploits the new features offered by the E-UTRAN. There are four basic entities that constitute the EPC and provide the IP connectivity service to the users connected to the RAN and to external networks connected to the EPC: Home Subscriber Server (HSS), Mobility Management Entity (MME), PDN Gateway (P-GW) and Serving Gateway (S-GW).

<b>HSS</b>	It is the main database in which the information of the network users is stored. The data contained in the HSS includes the information related to the user subscription and also information about the performance of the network. This entity can be accessed from other elements of the network to manage the access to the connectivity service. If the access is done from the E-UTRAN the MME will interact with the database via the S6 interface.
<b>MME</b>	It represents the principal entity of the control plane in order to manage the access of the users to the network via the E-UTRAN. There are more than one MME for network and its selection is done basing on the terminals geographical location or loading balancing criteria. The main functions of this entity are the following: <ul style="list-style-type: none"> <li>- Authentication and authorization of the users.</li> <li>- Management of EPS carrier services and mobility of the users.</li> <li>- Termination of the NAS protocols.</li> <li>- Signaling for supporting the mobility.</li> </ul>

<b>P-GW</b>	This entity is a gateway to the external network named PDN, which provides connectivity between the LTE network and the PDN and can support functions such as the IP address allocation to users' terminals or control mechanisms for the service quality parameters from the data sessions established by the LTE network. All the IP packets exchanged between these networks are directed by this gateway or interface named SGi.
<b>S-GW</b>	It is the gateway of the user plane connecting the E-UTRAN with the EPC backbone. As in MME, when a user is connected to the LTE system a S-GW is assigned to him/her depending on the terminal geographical location and loading balancing criteria. This entity provides an attachment point when the terminal moves from one eNB to the other and also when it moves from one LTE RAN to others, such as UTRAN (UMTS) or GERAN (GSM). Also, it is in charge of users traffic routing and temporal storage of users IP packets from terminals that are in idle mode. This is connected with the MME by the S11 interface and with the P-GW with the S5 interface.

**Table 2. Components of the Core Network.**

The enhancement in the architecture and other technical aspects has become LTE an appropriate standard to handle the current networks. In the following some characteristics of LTE standard and their benefits are presented [10] [11]:

- All IP network architecture.
- It can support packet and data services.
- Great diversity of frequency bands, some of them above the 3G bands, which implies the increase of the bandwidth and speed of data transmission.
- Supports MIMO, so it can achieve high data rates.
- SC-FDMA protocol in the uplink, which indicates low powering during transmission.
- OFDMA protocol in the downlink, which means an increase of the total user capacity.
- Very low latency, which provides a better network performance.

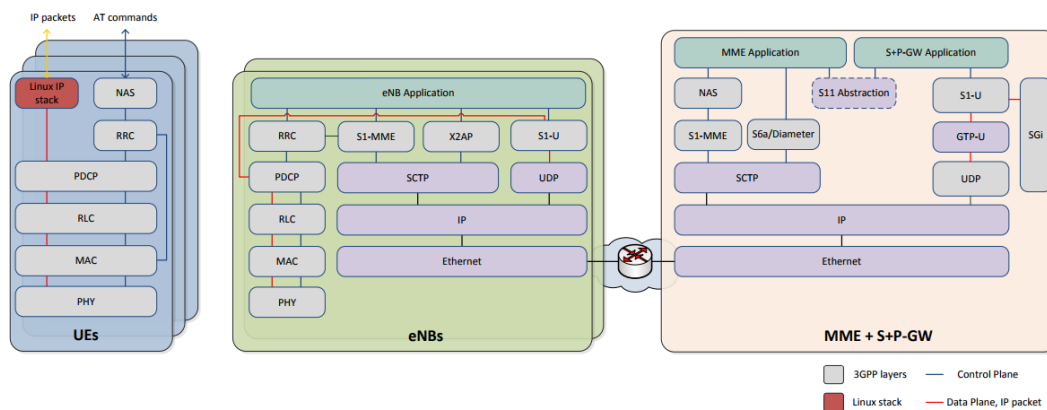
However, an excessive increase in data traffic that can lead to congestion and overflow of LTE networks has been predicted for 2020, since the current networks are not prepared to withstand such high capacities. This fact has led the operators of mobile networks to design more complex networks that have more features and extensions and which will be capable of dealing with the required traffic levels. All of this is needed to reach the 5th generation of mobile networks.

As explained in previous sections, virtualization technologies, such as SDN and NFV, are the most appropriate to achieve more flexible and manageable networks, which allow separating the control and user planes without increase the deployment or operation costs. Currently, studies are underway in which SDN and NFV are applied to the LTE

standard. An example of this is presented in [12] where a Cellular Software Defined Network has been designed using SDN and NFV.

## 2.4. OpenAirInterface platform

OpenAirInterface platform is an open-source software that implements the LTE system fully encompassing the 3GPP standard protocol both in E-UTRAN [15] and EPC [16] entities. It is based in a computer hosted software radio frontend architecture that allows to build and customize the LTE network entities on a computer and, to connect UEs to test different configurations in the network and monitor in real time the network and the mobile devices.



**Figure 7. OAI protocol stack scheme. Extracted from [17].**

In Figure 7, an outline is represented with the protocol stacks that implement each block of the software OAI platform. As shown, it consists of three blocks: the UE, the eNBs belonging to the E-UTRAN access network, and the MME + S + P-GW entities that make up the EPC. This scheme also allows us to appreciate the routes that make both the data plane and the control plane through the different layers of the 3GPP standard. Next, the main characteristics of the OAI software [18] are detailed.

### PHYSICAL LAYER

LTE release 8/9 with a subset of release 10 features

Frequency División Duplexing (FDD) and Time División Duplexing (TDD) in 5, 10 and 20 MHz in bandwidth

Transmission mode: 1,2,4, 5 and 6

Reporting of CQI and PMI

All downlink and uplink channels are supported.

Optimized baseband processing



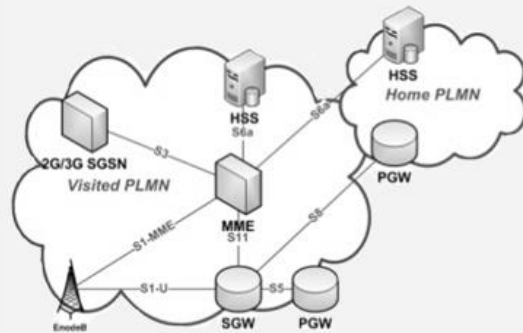
### E-UTRAN (RADIO ACCESS NETWORK)

- Implements MAC, RLC, PDCP and RRC
- Channel-aware proportional channel scheduling
- Reconfigurable protocol stack
- Support RRC measurements
- Standard S1AP and GTP-U interfaces with EPC
- IPv4 and IPv6 support



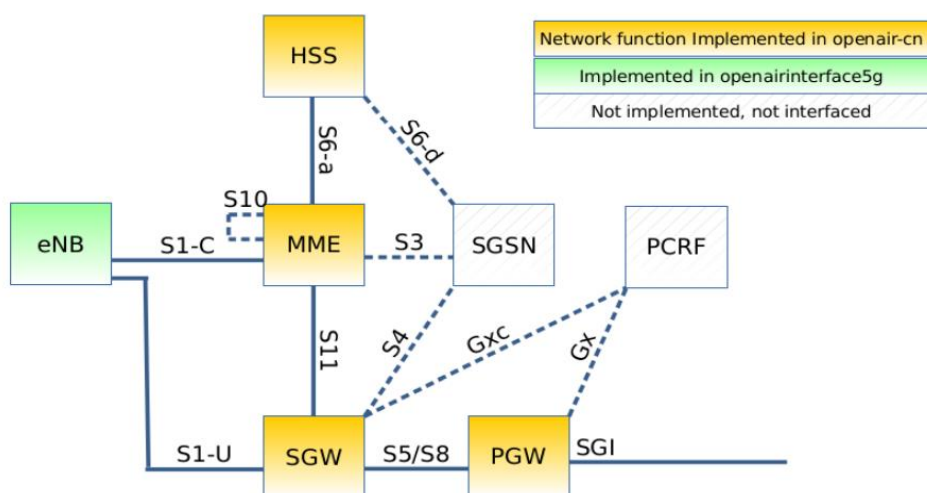
### EPC (CORE NETWORK)

- Implements MME, SGW, PGW and HSS
- NAS integrity
- UE handling procedures such as attach, authentication, service access and radio bearer establishment
- Transparent access to IP network
- Configurable APN, IP range and DNS
- IPv4 and IPv6 support



**Table 3. Characteristics of the physical layer, E-UTRAN and Core Network.**

Moreover, EUROCOM has exposed a complete scheme of the overall architecture of the software part of OAI platform, in which are defined the several entities that constitute the OAI LTE network and the interfaces that communicates the previous entities and indicates by which software is implemented each entity. In Figure 8 the scheme of the OAI architecture is presented.



**Figure 8. Overall OAI platform architecture. Extracted from [19].**

The OAI platform is constituted by two different software:

- **openairinterface5g** [15] which permits personalize the configuration parameters and run the eNB entity. As will be seen later, at the beginning of this project have been used the **Openairinterface5g** software because we just have been working with the LTE network but later the **empower-openairinterface** [25] that is a customized version of the former software have been deployed for the integration of the EmPOWER functions.
- **openair-cn** [16] which virtualize the four network functions of the core network: HSS, MME, S-GW and P-GW. In this case, the two last entities are seen as one, SPGW. In the same way as the eNB, these three entities can be customized according to the user's requirements.

In relation to the implemented entities are the interfaces that communicate them. Based on Figure 8, the following interfaces can be defined:

- **S1-C**: It communicates the eNB with the MME and is used to manage the control data.
- **S1-U**: It communicates the eNB with the SPGW and is used to manage the user data.
- **S6-a**: It communicates the MME with the HSS.
- **S11**: It communicates the MME with the SPGW.
- **S-GI**: It communicates the SPGW with the external network, such as Internet.

S5/S8 interface will not be needed because the SGW and PGW act as one unique entity.

In addition to software, radio frequency hardware will be necessary to create the interface between the commercial UEs and the eNB, allowing the real-time communication of the two parties. There are several frontend radio software platforms, such as EXPRESSMIMO2, USRP B2x0 or X300, RF Blade, LMS-SDR, among others. In this project, the USRP B210 (designed by Ettus Research/NI) is the one selected for the implementation of the eNB - UE interface. Here are some features of this device [20].

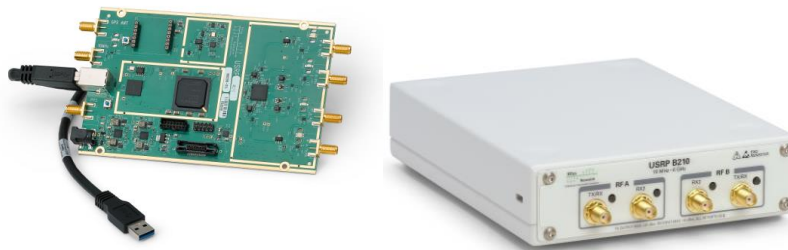


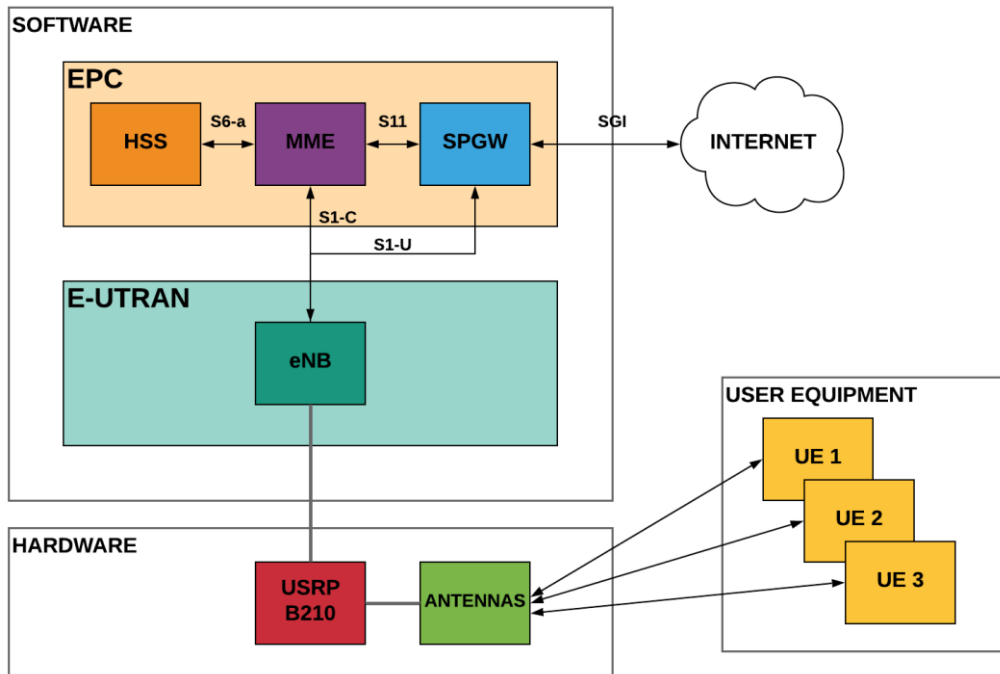
Figure 9. USRP B210, Ettus Research. Extracted from [20].

- Radio frequency coverage from 70 MHz to 6 GHz.
- USB 3.0 SuperSpeed interface and standard-B USB 3.0 connector.
- Multiple Input Multiple Output (MIMO), 2TX x 2RX) capability.
- Half and Full Duplex.
- Open and reconfigurable Xilinx Spartan 6 XC6SLX150 FPGA.
- Up to 30.72 MHz of real-time bandwidth in 2TX x 2RX.

According to the previous features 2 reception and 2 transmission antennas connected to the USRP device are required for the transmission and reception of information between the UEs and the LTE network.



Finally, a complete scheme of the OAI platform, integrating software and hardware, is shown below.



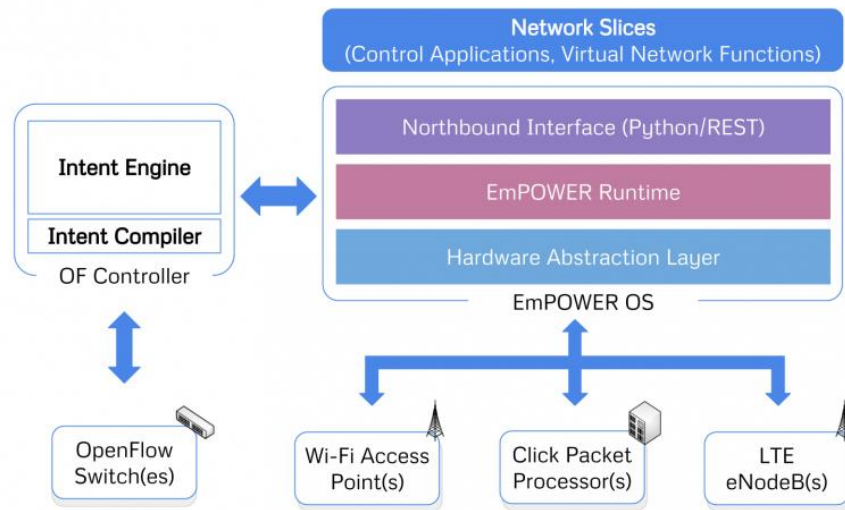
**Figure 10. Complete architecture of the OAI platform.**

## 2.5. 5G-EmPOWER platform

5G-EmPOWER (also referred as EmPOWER) is an open multi-access network operating system for Wi-Fi and LTE network, which is blurring the line between access and core network since it is introducing the concept of programmable data plane which simplifies the radio and packet processing resources of the network. This platform is characterized by the following features [14]:

- Flexible architecture and high-level programming APIs for fast prototyping of applications and services.
- Three types of virtualized network resources delivered: forwarding nodes, packet processing nodes and radio processing nodes.
- Virtualized radio access network that supports LTE and Wi-Fi technologies.
- Full visibility of the network status and dynamic deployment and orchestration of the network services.
- Centralized mobility management for LTE and Wi-Fi.
- Implementation of user resource allocation schemes within a network slice.
- Customized packet processing in a portion of the traffic.
- Supports multi-tenancy, that is supports the implementation of different slices of applications and services from different operators or tenants over the same network.
- Web-based control framework.

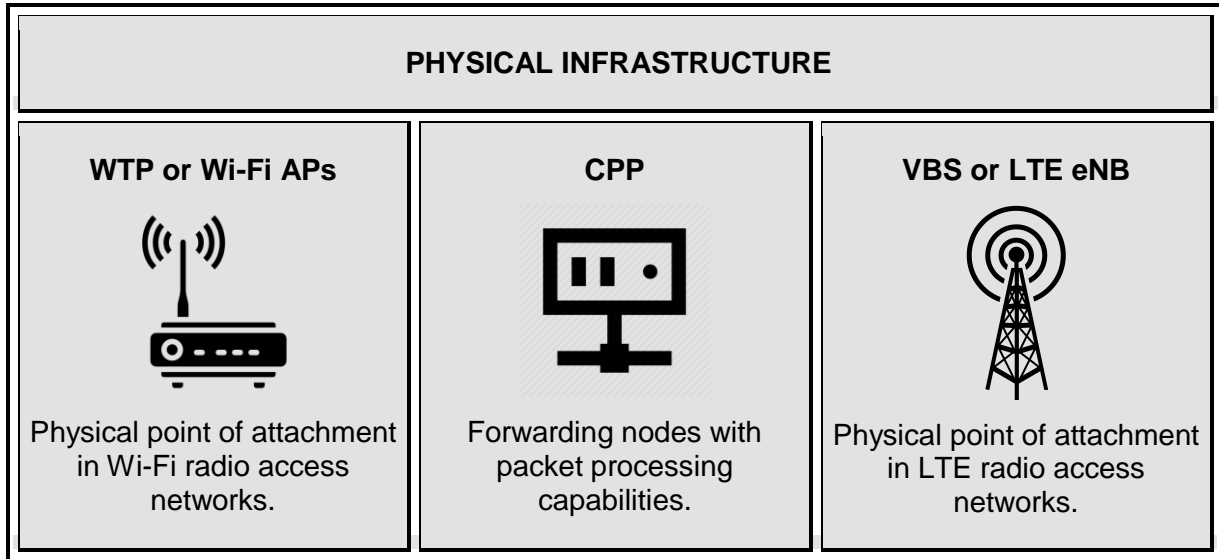
In the Figure 11 the architecture of the EmPOWER network is represented.



**Figure 11. EmPOWER network architecture. Extracted from [21].**

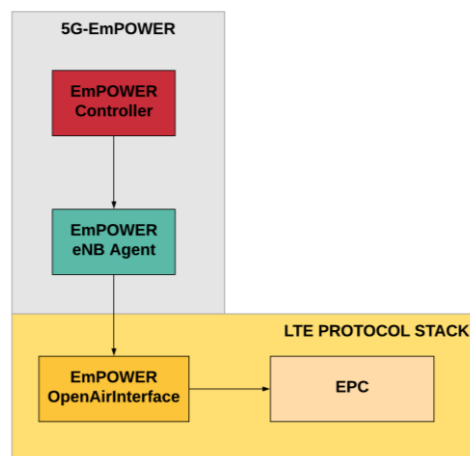
Below, the different components that make up the EmPOWER network architecture [22] are defined:

NETWORK APPLICATIONS		
They are in the top of the Controller running in their own slice of resources. They communicate with the controller via a Northbound interface.		
CONTROLLER		
The EmPOWER controller is an entity in the EmPOWER platform in charge of:		
<ul style="list-style-type: none"> <li>- Deploying the Light Virtual Access Point (LVAP)/Light Virtual Network Function (LVNF) on the network devices.</li> <li>- Supporting multi-tenancy on the top of the physical infrastructure. A tenant is defined as a virtual network with its own Wireless Termination Points (WTP)/ Click Packet Processors (CPPs) or Virtual Base Stations (VBSes).</li> <li>- Ensuring that a Network Application receives a view of the network from its own slice.</li> </ul>		
Some features that characterize the EmPOWER controller are presented in the following.		
SOFT STATE	MODULAR ARCHITECTURE	SLICING
List of network slices and clients authentication method are the only data that is kept stored in the controller whereas other are deleted when the network is disconnect from the controller.	All the tasks supported by the controller, excluding the logging subsystem, are deployed as a plug-in that is Python modules.	Multiple logical virtual slices or networks can be created in the top of the controller, each one of them defined by its own network name and set of WTPs (Wi-Fi) and VBS (LTE). Users can associate to a network or slice basing on its network identifier.



**Table 4. Architecture of the 5G-EmPOWER platform.**

Finally, a small diagram with the different components that make up the 5G-EmPOWER platform and how they interrelate with the LTE network is presented. Then, are described the different software that deploy each components of this diagram.



**Figure 12. 5G-EmPOWER block diagram.**

- **empower-runtime** [23] is implementing the Python-based EmPOWER controller.
- **empower-enb-agent** [24] is implementing the LTE Agent. The Agent is the interface between the LTE network and the Controller. It contains a wrapper that is used to translate the exchanged information to allow both entities understand among them.
- **empower-openairinterface** [25] is implementing the eNB entity of the LTE network. As aforementioned, this software is a customized version for the development of the EmPOWER functionalities.

Many other components are implemented by the 5G-EmPOWER platform but since we are going to work with an LTE network they are not explained in this thesis.

### 3. Methodology

As mentioned at the beginning of the thesis, the purpose of this work is to contribute to the implementation of RAN Slicing over LTE technology on a real-time platform based on the 5G-EmPOWER software. For the implementation of this slicing technique, previously, it is necessary to integrate the 5G-EmPOWER platform, which provides the controller and agent software, with the 3GPP protocol stack of OpenAirInterface (OAI), which emulates an LTE network. In Figure 13 the initial diagram of both platforms is represented.

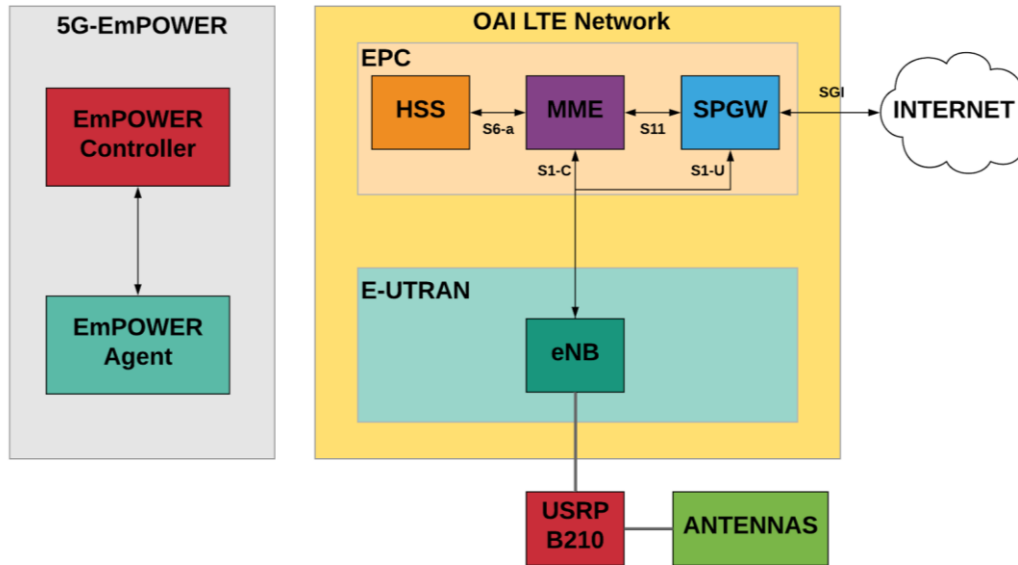


Figure 13. Overall architecture of the 5G-EmPOWER and OAI LTE network.

Next, the different steps that have been followed until reaching our target objective are detailed.

#### 1. Study of the OAI LTE network platform (1 and 2 PCs configuration) and mobile devices connection.

The project began with the study of the OAI platform. This platform can be configured in 2 PCs, implementing the evolvedNodeB (eNB) and the Evolved Packet Core (EPC) in two different PCs, or in 1 PC, implementing the eNB and the EPC in a single PC. In Figure 14 the two configurations are represented.

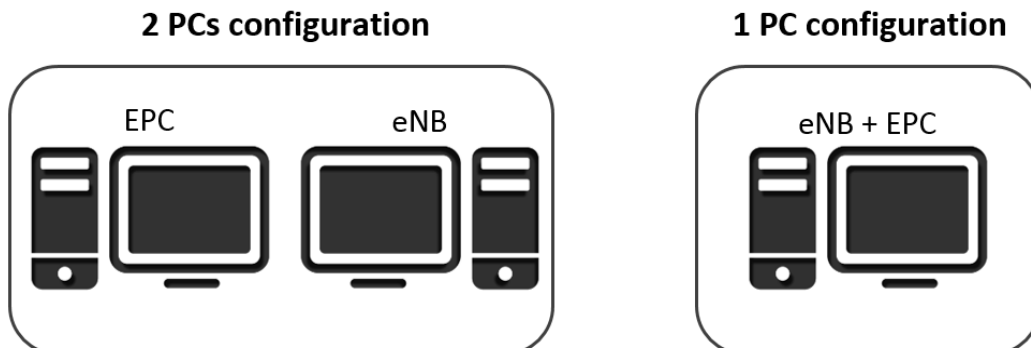


Figure 14. OAI LTE network configurations.

We started working with the configuration of 2PCs to familiarize ourselves with the platform. First, we look for the structure of the OAI network, that is, which entities and interfaces constitute it. Next, we were working on understanding the configuration files to mount the network (IP addresses of the interfaces and principal network parameters). Once the network was configured, we searched for the Linux commands to run the several entities and which was the order to run them to launch the network. Finally, we perform connection tests. Previously, we had to know how were configured several elements by former developers of this OAI implementation: the SIM cards to create a network (operator) client, the database to associate the clients to this network and the mobile terminals to connect to the network. The same procedure was performed in the 1PC configuration. From this study we made some tutorials for future developers, one explaining the procedure to launch the network for both configurations and perform connectivity tests and another detailing the process to register users in the database of the network.

From this point, we worked with the 1 PC configuration.

## 2. Study of 5G-EmPOWER platform. The controller.

Next, the 5G-EmPOWER platform was deeply analysed. In this case, we did some research to know about its structure, how it is configured, what are the commands to run the controller, what are the information messages that provides the controller terminal and how the controller web interface works.

Once we knew the different software that make up this platform, we set up the controller from its web interface. Thanks to the information provided in [22] we learned how to configure the controller, customizing the parameters according to our network. The first parameter that we need to configure was the Virtual Base Station (VBS) to associate it to the controller. Finally, we looked for the commands to launch the controller.

## 3. Integration of OAI and EmPOWER platforms.

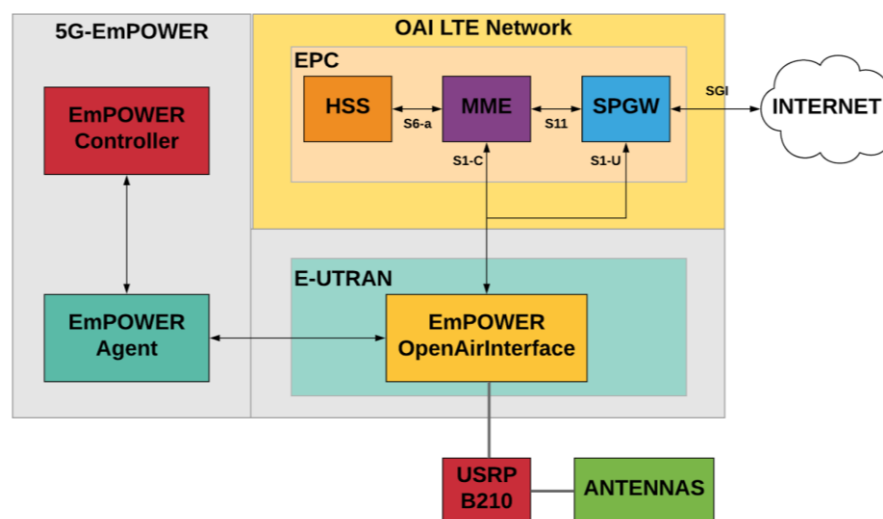


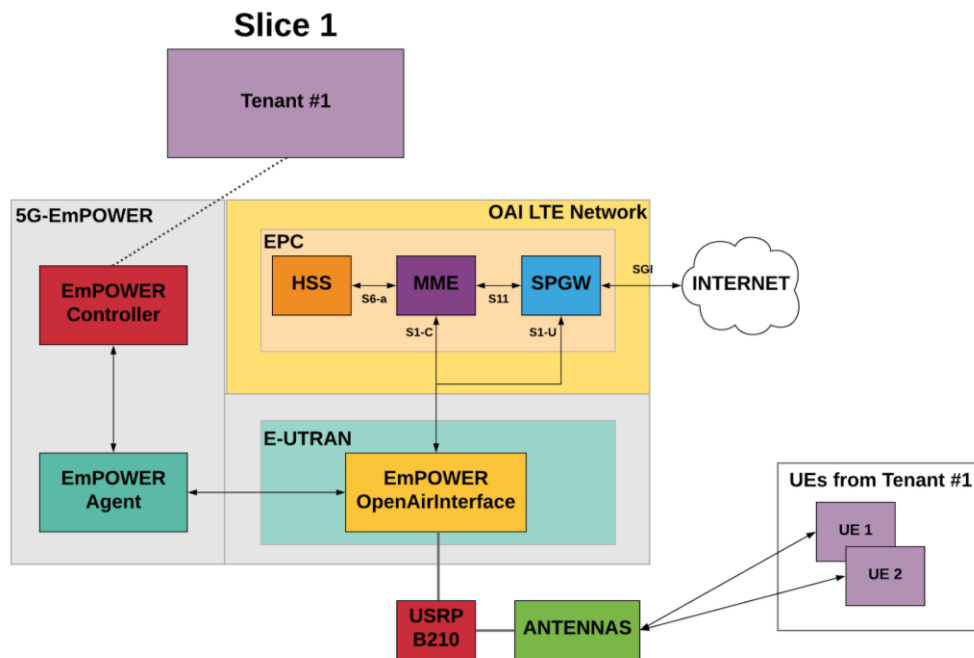
Figure 15. Integration of 5G-EmPOWER and OAI platforms.

For the integration of both platforms it was necessary to download a new version of the eNB (empower-openairinterface), from the 5G-EmPOWER, that had enabled the option to connect with the agent, and that allowed the connection of the eNB with the controller. First, we had to find a way to create the connection between the agent and the controller and the agent and the eNB. Subsequently, we set up the agent and performed the first connectivity tests between the controller and the OAI LTE network. To connect the different entities, previously, we had to find the order to run them. In Figure 15 the integration of both platforms is represented.

#### 4. RAN Slicing and mobile devices connection.

Once the network was integrated the last step was implementing the RAN Slicing for one virtual operator. Therefore, we needed to know how to create the slices or tenants associated with an operator and how to configure them to associate to it a VBS and mobile devices. After making all the configurations, we perform connectivity tests to verify the association of the VBS and users mobile devices with its tenant.

From the work done in points 3, 4 and 5 we wrote a tutorial detailing the different steps that must be followed for the configuration, integration and connection of the tenant, the controller and the LTE network. In Figure 16 the final network with the RAN Slicing implemented is shown.



**Figure 16. Final network with RAN Slicing implemented.**

At this point, we had already reached the goal proposed in the project. However, we decided to go further creating a new operator and a new eNB configuration that would allow to connect two different operators to the same RAN. The proposed objectives for this extension of the project were the creation of a new operator (programming new SIM cards, create database and configuration of mobile devices), the construction of a new EPC capable of connecting to the eNB and the controller, and finally, the integration of

the two EPCs with the eNB and the controller and therefore the association of the users of each operator to its corresponding tenant.

### 5. Creation of a new operator.

The first step to create the new operator was to define the parameters of the new network, taking into account that they could not match the parameters of the existing operators. Next, we had to find out how to configure the SIM cards to enter these parameters and create a client for that operator. Once the new client was defined, we create a new database according to the user's parameters.

### 6. Creation of a new EPC and mobile devices connection.

The next step was the configuration of the entities of the new EPC, adjusting to the network parameters defined above. We also had to reconfigure the eNB to create the new operator's network and define a new tenant for this network. Finally, we performed connectivity tests to verify that the entire network worked correctly.

### 7. Integration of the 2 EPCs with the eNB and Controller and mobile devices connection.

In the latter step we found that the eNB could be configured in some way to connect to 2 EPCs. After information research we found a type of configuration that allowed it. Therefore, we configure the eNB with the parameters of both EPCs. When using OAI in 1 PCs configuration we could not perform the connection correctly because the two EPCs could not connect at the same time. One solution to this was to use the 2 PCs configuration, in this case 3 PCs, which allowed us to separate the eNB and the two EPCs from each other. We learned how we had to configure the different PCs for this new configuration and perform connection tests. In Figure 17 the final network diagram with the implementation of two network operators is traced.

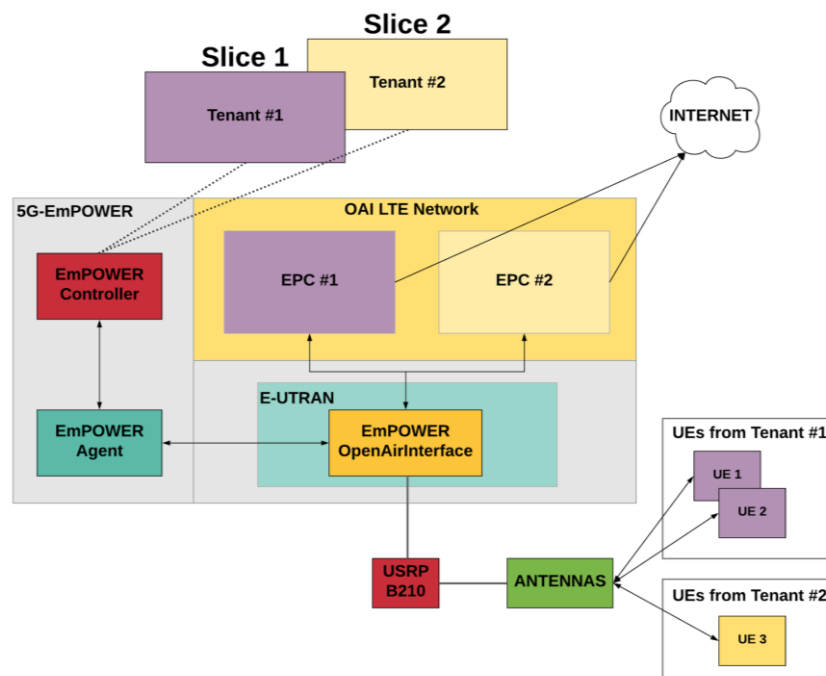


Figure 17. Final network with Multi Operator Core Networks (MOCN).



## 4. Development and achieved results

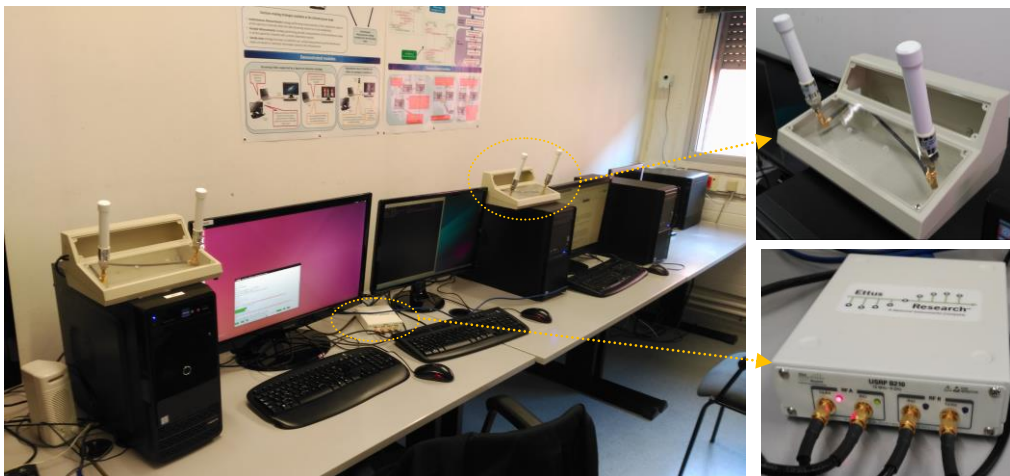
This section describes the various processes carried out during the development of the project to achieve the integration of the OpenAirInterface and 5G-EmPOWER platforms and the implementation of the RAN Slicing in a real-time testbed. In addition, the different results obtained from each process are exposed and analysed.

Before explaining the development of the project we will present the place where we have worked and the hardware that we have arranged to carry out the project.

### 4.1. Presentation of the workspace

This section shows the workspace in which the project has been developed, the different PCs that we have used to run the OAI network, the 5G-EmPOWER Controller and the mobile devices or User Equipments (UEs) used to test the network connectivity. Other hardware, such as the Universal Software Radio Peripheral (USRP) and antennas, that have also been necessary to carry out the project are presented.

The D4-112 laboratory at the UPC North Campus, assigned to the GRMC, has been the place where we have developed the project. Figure 18 shows the workspace.



**Figure 18. Project workspace.**

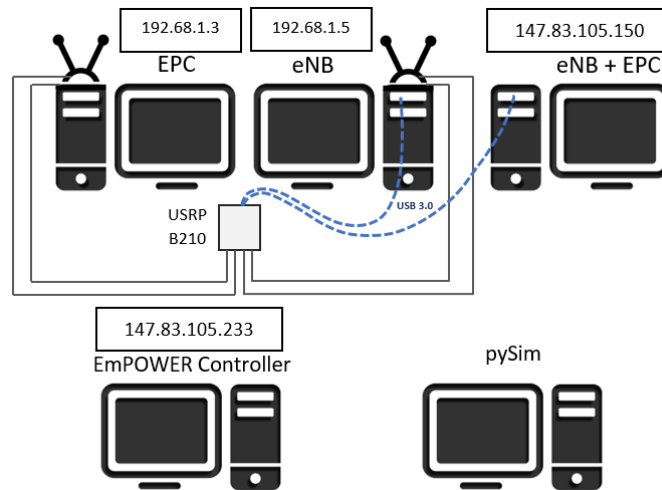
In this project we have had at our disposal different devices that have allowed us to implement the OpenAirInterface and 5G-EmPOWER platforms. All of them are presented below.

- 3 PCs with the distribution of Linux Ubuntu 14.4 installed for the implementation of the OpenAirInterface platform. As mentioned in the Methodology for the study of this platform, we have worked with 2 different PC configurations or distributions. The first one is the configuration of 2PCs, in which eNB and EPC are implemented in different PCs. The second one is the configuration of 1 PC, where all the entities are implemented in the same PC.
- 1 Virtual Machine (VM) with the Linux distribution Ubuntu 14.4 (Linux) installed for the implementation of the EmPOWER Controller.
- 1 PC for programming SIM cards, using the PySIM software.



- A USRP B210, which is a SDR system in charge of building the eNB to provide communication between the users and the network. This one is connected, by means of a cable USB 3.0, to the PC that contains the eNB software and, by means of UTP cables, to two pairs of antennas.
- 4 antennas working in band 7, that is, at a frequency of 2600 MHz and with MIMO configuration, with 2 reception antennas and 2 transmission antennas.

In Figure 19, a schema with the hardware explained above is shown. In this, in each PC its function is specified and for the PCs that are dedicated to implement network functions the host addresses are indicated. It is important to know these addresses, as we will see later, to configure the different interfaces of the network.



**Figure 19. Scheme of the OAI LTE network configurations.**

In addition to all the equipment necessary to run the LTE network and the controller, mobile devices or User Equipments (UEs) with customized SIM cards are required to verify the network connectivity.



**Figure 20 . User Equipments (UEs).**

The available terminals in the laboratory are Samsung Galaxy SIII, Sony Xperia Z5, Samsung Galaxy SV and 2 LTE Dongles Huawei E398, shown from left to right in Figure 20. Whereas, the SIM cards employed are the sysmoUSIM-SJS1 [26] from sysmocom,

which is presented in Figure 21. The main advantage of this SIM cards is that they are reprogrammable, so we can create and edit any parameter in the card. These parameters will be configured according to the parameters of the network to which the user wants to connect. Later on, the procedure to configure the SIM cards will be described.

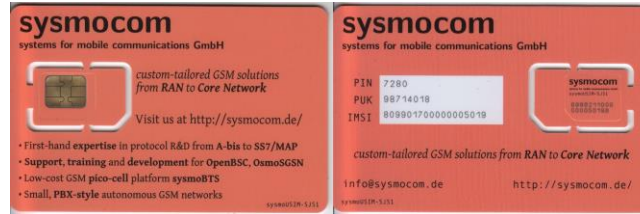


Figure 21. Sysmocom SimCard. Extracted from [26].

In the following sections, we will present the procedures followed to achieve the main objective of the project. As a first step, we will explain the study carried out for each of the platforms that constitute our testbed, with the configuration of the different entities and their launch. Later, we will detail the process of integration and implementation of the slicing of the radio access network for a single network operator.

## 4.2. OAI platform: Configuration and terminals connection

The OpenAirInterface platform is responsible for building the LTE network, using NFV technology. As we have mentioned several times in the thesis, the implementation of this platform can be done with 2 PCs, with which we would have the RAN separated from the CN, or in a compact version with the two entities running on the same PC.

To study this platform we have worked with the two PC distributions but at the integration phase we used the compact distribution, so we will present the configurations and results obtained for both cases.

### 4.2.1. Network architectures

The architectures of the OAI LTE network for the 2 and 1 PC configurations are shown in Figures 22 and 23. The diagrams include details of the IP addresses of each entity and the interfaces that connect the different elements of the network.

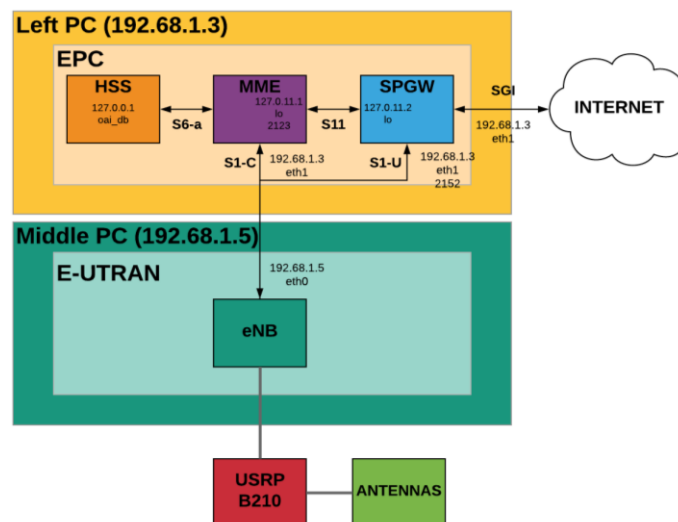


Figure 22. Configuration of the OAI LTE network for 2 PCs.

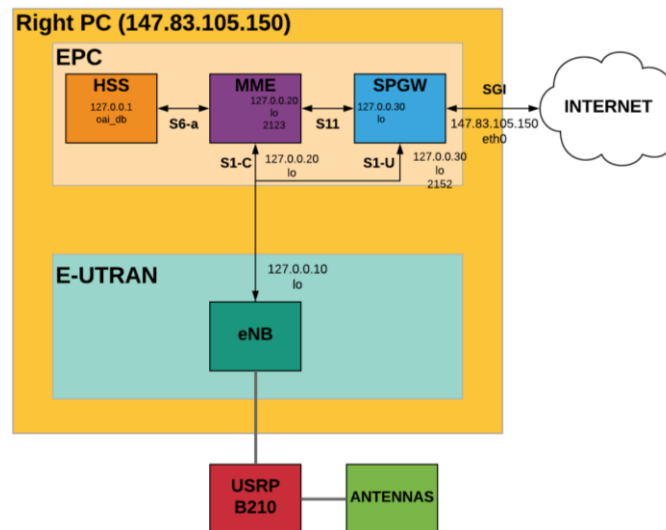


Figure 23. Configuration of the OAI LTE network for 1 PC.

In the following tables we can see in summary the assignment of IP addresses and interfaces for each configuration.

- **2 PCs configuration:** The interfaces that are communicating entities from the same host are defined as local addresses (lo) while the interfaces that connect entities from different hosts (MME and SPGW with eNB) or connect the EPC entity (SPGW) to the external network are defined as hosts addresses (eth0/eth1). In Table 5, the allocation of IP addresses for 2 PCs configuration is presented.

	eNB	MME	SPGW	External network
eNB		192.168.1.5 (eth0)	192.168.1.5 (eth0) 2152	
MME	192.168.1.3 (eth1)		127.0.11.1 (lo) 2123	
SPGW	192.168.1.3 (eth1) 2152	127.0.11.2 (lo)		192.168.1.3 (eth1)

Table 5. Allocation of IP addresses for 2 PCs configuration.

- **1 PC configuration:** All the interfaces are defined as local addresses (lo) because all the network entities are running in the same equipment, excluding the interface that connects the SPGW to the external network that is defined as a host address (eth0). In Table 6, the allocation of IP addresses for 1 PC configuration is presented.

	eNB	MME	SPGW	External network
eNB		127.0.0.10 (lo)	127.0.0.10 (lo) 2152	
MME	127.0.0.20 (lo)		127.0.10.20 (lo) 2123	
SPGW	127.0.0.30 (lo) 2152	127.0.0.30 (lo)		147.83.105.150 (eth0)

**Table 6. Allocation of IP addresses for 1 PC configuration.**

Although both architectures differ in the IP addresses assignment, the main network parameters are the same for both of them. Next, the main parameters that define the network are explained and the election of each of them is justified.

#### 4.2.2. General network parameters

The Public Land Mobile Network Identifier (PLMN Id) is a 5 or 6 digit parameter that identifies a specific network by defining the Mobile Country Code (MCC) that determines the country to which the network belongs and the Mobile Network Code (MNC) that specifies the operator of the network (MNC).

- **MCC** is a 3 digit value assigned by the ITU. In our network, the MCC is 214 corresponding to the code of Spain.
- **MNC** is a 2 or 3 digit assigned by National Authority. The choice of this parameter has been made based on the following table that indicates the MNC codes not currently available in Spain, accompanied by the operator they represent.

MNC	Network Operator	MNC	Network Operator
01	Vodafone	17	R Cable y Telec. Galicia SA
03	Orange	18	Cableuropa SAU (ONO)
04	Yoigo	19	Simyo/KPN
05	Movistar	20	fon You Wireless SL
06	Vodafone Enabler España SL	21	Jazz Telecom SAU
07	Movistar	22	Digi Spain Telecom SL
08	Euskaltel SA	23	Lycamobile SL
09	Orange	25	Lycamobile SL
11	Orange	26	Lleida
15	BT España SAU	27	Truphone
16	Telecable de Asturias SA	32	ION Mobile

**Table 7. Assignment of MNC codes by operator in Spain. Extracted from [27].**

Based on the MNC shown in Table 7, the MNC selected for our network is 91.

PLMN Id	
MCC	MNC
214	91

**Figure 24. PLMN Id definition.**

Therefore, knowing these two values we can extract the PLMN Identifier from our network which is 21491.

Other important identifiers, related to the MME entity, are Globally Unique MME Identity (GUMMEI) and Tracking Area Identity List (TAI List).

- **GUMMEI** is the unique MME identifier, which is formed by the PLMN Id, the MME Group Identifier (MMEGI, unique within a PLMN) and the MME Code (MMEC). These two last parameters uniquely identify the MME within a particular network and take the following values in our network: MMEGI is 4 and MMEC is 1.

GUMMEI		
PLMN Id	MMEGI	MMEC
21491	4	1

**Figure 25. GUMMEI definition.**

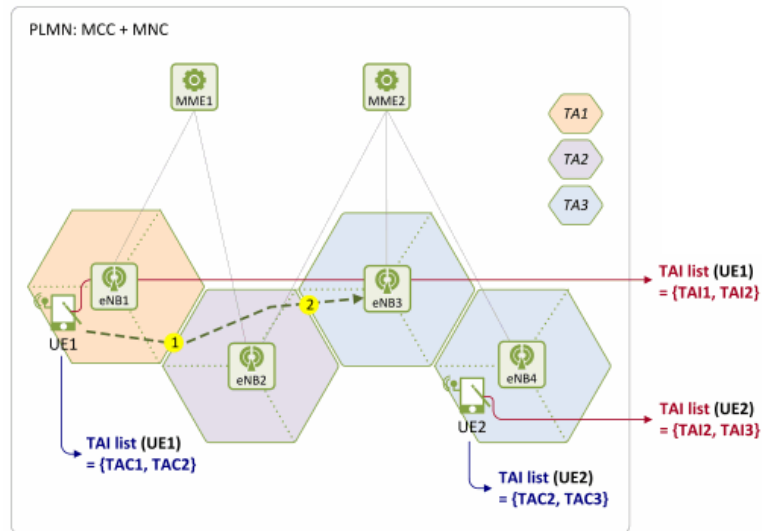
Therefore, the GUMMEI of our network is 2149141.

- **TAI List** is the set of Tracking Area Identities (TAIs) that constitute the network. This parameter is formed by the PLMN Id and the Tracking Area Code (TAC) that identifies a tracking area within a particular network. In our network, the TAI List is just composed by 1 TA, so the TAC is 1.

TAI List	
PLMN Id	TAC
21491	1

**Figure 26. TAI List definition.**

By the way, the Tracking Areas (TAs) are non-overlapping units that are used to track the location of the mobile devices that are in standby mode. Figure 27 shows a representation of TAs.

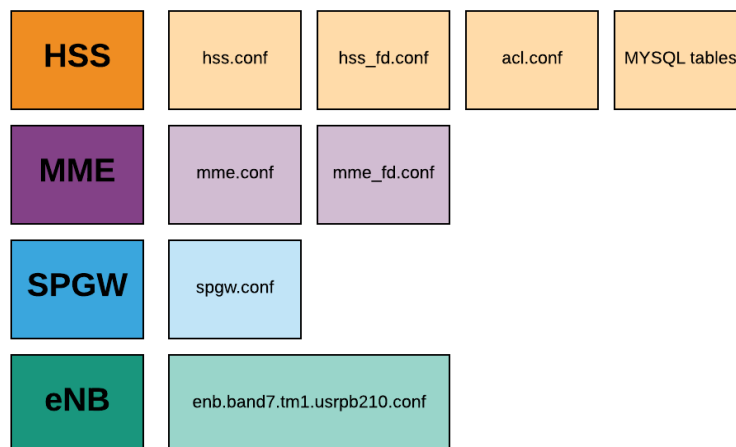


**Figure 27. Representation of TAs. Extracted from [28].**

All these parameters define the network and are essential together with the IP addresses and interfaces for their configuration. In the following subsection, the procedure to configure the OAI LTE network for the 2 and 1 PC configurations is detailed. Since in both cases the same procedure is followed, although they differ on some values, we will show in parallel how the two configurations were made.

#### 4.2.3. Network configuration for 2 and 1 PC

The OAI platform has different configuration files to customize the network according to the requirements of each user. Next, the files that we have modified to configure the different entities of the network are exposed.



**Figure 28. Configuration files OAI platform.**

The parameters that have to be modified for each entity, according to the network parameters and interfaces explained above, are detailed below.

## HSS configuration

The main function of the HSS is to store all the information of the network users, so for its configuration it will be necessary to create a database composed of MySQL tables that store all this information.

To configure the HSS there are 2 steps to follow:

1. Personalizing the configuration files according to our network parameters: hss.conf, hss\_fd.conf and acl.conf.
2. Configuring the MySQL tables (database).

The configuration of the HSS will be the same for the 2 configurations of OAI LTE network.

The first configuration file modified was **hss.conf**. In Figure 29 a screenshot with the varied parameters is shown.

```

21  HSS :
22  {
23  ## MySQL mandatory options
24  MYSQL_server = "127.0.0.1";      # HSS S6a bind address
25  MYSQL_user   = "root";          # Database server login
26  MYSQL_pass   = "mysqlpa33w0rd"; # Database server password
27  MYSQL_db     = "oai_db";        # Your database name
28
29  ## HSS options
30  #OPERATOR_key = "29C7D674257AF3B0C6C9F7879663F5FB"; # OP Ki from sysmocom USIM_2
31  #OPERATOR_key = "1006020f0a478bf6b699f15c062e42b3"; # OP key matching your database
32  OPERATOR_key = "11111111111111111111111111111111"; # OP key matching your database
33
34  RANDOM = "true";                # True random or only pseudo
    random (for subscriber vector generation)
35
36  ## Freediameter options
37  FD_conf = "/usr/local/etc/oai/freeDiameter/hss_fd.conf";
38  };

```

**Figure 29. Configuration of hss.conf for 1 and 2 PCs configurations.**

In Table 8, all these parameters are described:

Parameter	Description	Value
<b>MySQL_server</b>	IP address that indicates where the HSS database is stored.	127.0.0.1
<b>MySQL_user</b>	Username to access the database.	root
<b>MySQL_password</b>	Password to access the database.	mysqlpa33w0rd
<b>MySQL_db</b>	Name of the database.	oai_db
<b>OPERATOR_key</b>	Operator key matches the key entered in the SIM cards when they are configured.	11111111111111111111111111111111 1111111111

**Table 8. Configuration parameters hss.conf.**



This file configures the location and access data to the database and the OPc parameter, which is an operator key. All users of the same operator will have the same OPc in their configuration.

Then, the **hss\_fd.conf** and **acl.conf** configuration files were analysed. According to the EPC User Guide provided by OAI [16], there is no need to configure this files because they are created and configured themselves during the process of compilation basing on the host parameters (hostname, realm,...) where the HSS is implemented. In Figure 30 some important parameters for the **hss\_fd.conf** file are presented.

```

1  # ----- Local -----
2  # The first parameter in this section is Identity, which will be used to
3  # identify this peer in the Diameter network. The Diameter protocol mandates
4  # that the Identity used is a valid FQDN for the peer. This parameter can be
5  # omitted, in that case the framework will attempt to use system default value
6  # (as returned by hostname --fqdn).
7  Identity = "hss.openair4G.eur";
8
9  # In Diameter, all peers also belong to a Realm. If the realm is not specified,
10 # the framework uses the part of the Identity after the first dot.
11 Realm = "openair4G.eur";

65 # Specify the addresses on which to bind the listening server. This must be
66 # specified if the framework is unable to auto-detect these addresses, or if the
67 # auto-detected values are incorrect. Note that the list of addresses is sent
68 # in CER or CEA message, so one should pay attention to this parameter if some
69 # addresses should be kept hidden.
70 ListenOn = "127.0.0.1";

```

**Figure 30. Configuration parameters of hss\_fd.conf for 1 and 2 PCs configurations.**

In Table 9, all these parameters are described:

Parameter	Description	Value
<b>Identity</b>	Identifies the peer in the Diameter network that contains a valid Fully Qualified Domain Name (FQDN <sup>1</sup> )	hss.openair4G.eur
<b>Realm</b>	Domain name associated with the host.	openair4G.eur
<b>Listenon</b>	IP address on which to bind the listening server.	127.0.0.1

**Table 9. Configuration parameters of hss\_fd.conf.**

In the case of the **acl.conf** configuration file, there is just one parameter to configure as shown Figure 31.

```

18
19 ALLOW_OLD_TLS *.openair4G.eur
20

```

**Figure 31. Configuration of acl.conf for 1 and 2 PCs configurations.**

<sup>1</sup> FQDN is a name that includes the host name and the domain name associated with that host, named realm.



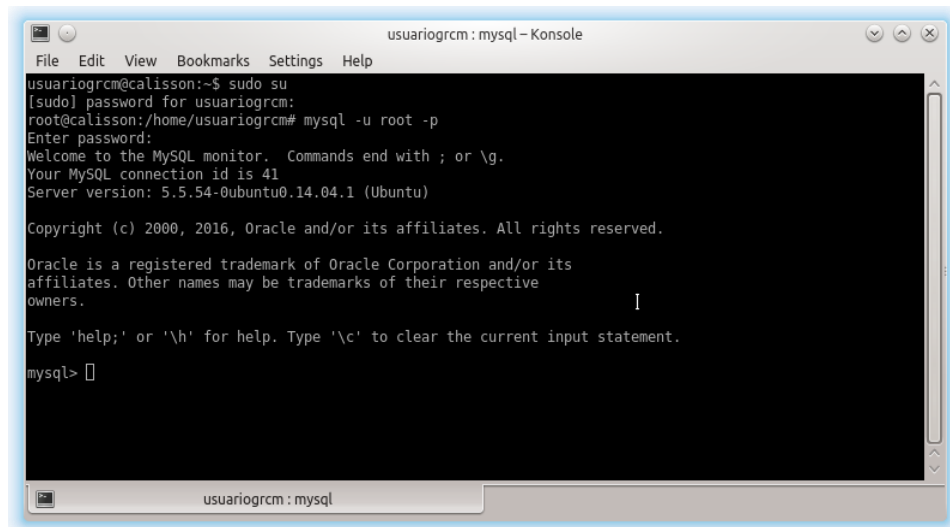
In Table 10, this parameter is described:

Parameter	Description	Value
<b>ALLOW_OLD_TLS</b>	It indicates that we accept unprotected CER/CEA exchange with Inband-Security-Id = TLS	*.openair4G.eur

**Table 10. Configuration parameter of acl.conf**

All the previous figures show specific parts of the configuration files in which have been modified or analysed some parameters to adjust to the network. The complete configurations files for all the network entities are exposed in Annex 1.

Then, the database was configured to register users of the network. This is built using MYSQL tables, which are configured by Linux console (Figure 32). In this section we will not specify the commands used to configure the database, for this look Annex 2, but we will indicate which tables have to be created and the parameters that must be entered in each of them.



**Figure 32. MySQL console interface.**

To facilitate the visualization of the content of the database, there is a user-friendly interface called phpmyadmin [29] that has allowed us to manage the MySQL administration over the web. To access this interface it is necessary to enter the following URL in the browser: <http://127.0.0.1/phpmyadmin/>.



**Figure 33. phpMyAdmin database web interface.**

Below the procedure to configure the database is explained.

We have a database created called `oai_db`, which contains several tables of the OAI, such as `apn`, `mmeidentity`, `pdn`, `pgw`, `terminal-info` and `users`. To register a new user just 3 of them will be updated:

- **users:** It contains the information about the subscriber.
- **pdn:** It contains the association between a subscriber (`user_imsi`) and an Access Point Name (APN) and its Quality of Service (QoS) parameters.
- **mmeidentity:** It contains the record corresponding to the MME.

To define the parameters of the three tables it is essential to know the users of the network. In this case, we had 4 SIM cards to test the connectivity of the network. These cards are the ones we have used for the construction of the database. In addition to the user data, other parameters have to be defined in the database. All of them have been defined according to the values indicated in [30].

	#1	#2	#3	#4
<b>MCC</b>	214			
<b>MNC</b>	91			
<b>TAI</b>	1			
<b>IMSI</b>	214910000009911	214910000009914	214910000009915	214910000009916
<b>Key</b>	29C7D674257AF3B0C6C 9F7879663F5FB	4403368D7F3ABEE7CB ED3155D134ABFA	3EC6CED8F8115403545 58 A9BF18C631B	3AFA3506B8F4775FBFA 80 8F56F59D801
<b>OPc</b>	11111111111111111111111111111111			
<b>ADM1</b>	86416194	71312245	47065816	00545387
<b>CC</b>	34			
<b>ICCID</b>	893491000000099112	893491000000099146	893491000000099153	893491000000099161

**Table 11. Configuration parameters of SIM cards.**

Table 11 shows the main configuration parameters of SIM cards programmed for our private LTE network.

The different parameters that have not been defined so far in the thesis will be described below to better understand each of them.

- **IMSI:** International Mobile Subscriber Identity. The IMSI has a maximum of 15 digits and is the combination of **MCC** | **MNC** | **MSIN**. The last parameter is the Mobile Subscriber Identification Number and it is a nine or ten digit code that identifies the Mobile Station.
- **Key:** This parameters and the OPc are the ones used for authenticating the users on a mobile network. The Key or K is a 128-bit master key located into the USIM and the HSS entity by the carrier. In our case we have employed the default values of the Ki, which is presented in the USIM's vendor (see Annex 2).
- **OP:** It is a 32-bit Operator Variant Algorithm Configuration Field. Each operator must have a different OP value (expressed in hexadecimal format). From this value and Key will be computed the OPc.
- **ADM1:** Administrator PIN for card personalization. This key is provided by the USIM's vendor and cannot be modified.
- **CC:** Country Code. For example, in Spain it is 34.
- **ICCID:** Integrated Circuit Card Identifier. The format of the ICCID is the following **MM CC II N C** :
  - **MM:** ISO 7812 Major Industry Identifier. For example, in our case this code is 89, which corresponds to the Telecommunications administrations and private operating agencies.
  - **CC:** Country Code. 34 in Spain.
  - **II:** Issuer Identifier.
  - **N:** Account ID (SIM number)
  - **C:** Checksum calculated from the other digits.

According to these parameters and those provided by OAI, we obtained the following tables<sup>2</sup>. To know the complete procedure to add a new user see Annex 3.

First, we created the user table as presented in Figure 34.

imsi	msisdn	imei	imei_sv	ms_ps_status	rau_tau_timer	ue_ambr_ul	ue_ambr_dl	access_restriction	mme_cap
IMSIs are the main reference key.	The basic MSISDN of the UE (Presence of MSISDN is optional).	International Mobile Equipment Identity	Mobile Equipment Identity Software Version Number	Indicates that ESM and EMM status are purged from MME		The Maximum Aggregated uplink MBR to be shared across all Non-GBR bearers according to the subscription of the user.	The Maximum Aggregated downlink MBR to be shared across all Non-GBR bearers according to the subscription of the user.	Indicates the access restriction subscription information, 3GPP TS 29272 #7.3.31	Indicates the capabilities of the MME with respect to core functionality e.g. regional access restrictions.
214910000009914	34600000001	NULL	NULL	NOT_PURGED	120	50000000	100000000		47 0000000000
214910000009911	34600000002	NULL	NULL	NOT_PURGED	120	50000000	100000000		47 0000000000
214910000009915	34600000003	NULL	NULL	NOT_PURGED	120	50000000	100000000		47 0000000000
214910000009916	34600000004	NULL	NULL	NOT_PURGED	120	50000000	100000000		47 0000000000

mmeidentity_id	mmeidentity_key	rfsp_index	urp_mme	sqn	rand	OPc
mmeidentity_id	UE security key	RFSP-Index: An index to specific RFSP configuration in the E-UTRAN. Possible values from 1 to 256	UE Reachability Request Parameter indicating that UE activity notification from MME has been requested by the HSS			Can be computed by HSS
1	4403368d7f3abec7bed3155d134abfa	1	0	0000000000000000000025824	a5a50a6e061ead0879e513681a9e8931	2497b7b0779c432007ab840cf92a4de
1	29c7d674257af3b0c6c9f7879663f5fb	1	0	0000000000000000000014272	3553c2326a93695c6b4e186babcdf6eb	9319ed3ee808bd467d42d3957c597cda
1	3ec6c6dbf811540354558a96f18c631b	1	0	000000000000000000005888	b30471328c4b037d6ca91005f103300b	de8928763accc20752137389ca3af72
1	3afa3506b8f4775f8fa808f56f99801	1	0	000000000000000000001088	9085cf18b4d0276cce58114ba24f7844	52f848b294c06909f519fca0a0a62724

Figure 34. Configuration of users table.

<sup>2</sup> users and pdn tables will be the same for the 1 PC configuration as for the 2 PCs configuration, while the mmeidentity table will vary due to the identity of the MME.

Next, the pdn table was filled in, as shown in Figure 35.

id	apn	pdn_type	pdn_ipv4	pdn_ipv6	aggregate_ambr_ul	aggregate_ambr_dl	pgw_id	users_imsi	qci	priority_level	pre_emp_cap	pre_emp_vul	LIPA-Permissions
1	oai.ipv4	IPv4	0.0.0.0	0:0:0:0:0:0	50000000	100000000	3	214910000009914	7	7	DISABLED	ENABLED	LIPA-only
2	oai.ipv4	IPv4	0.0.0.0	0:0:0:0:0:0	50000000	100000000	3	214910000009911	7	7	DISABLED	ENABLED	LIPA-only
3	oai.ipv4	IPv4	0.0.0.0	0:0:0:0:0:0	50000000	100000000	3	214910000009915	7	7	DISABLED	ENABLED	LIPA-only
4	oai.ipv4	IPv4	0.0.0.0	0:0:0:0:0:0	50000000	100000000	3	214910000009916	7	7	DISABLED	ENABLED	LIPA-only

Figure 35. Configuration of pdn table.

Finally, we built the mmeidentity table (Figures 36 and 37), which defines the identity of the MME of our network in order to be able to connect with it. In this case one MME is implemented, so a single value has been defined. It must be verified that the name assigned in the database is the same as the one entered in the mme\_fd.conf configuration file.

idmmeidentity	mmehost	mmerealm	UE-Reachability
1	nano.openair4G.eur	openair4G.eur	0

Figure 36. Configuration of mmeidentity table for 2 PCs configuration.

idmmeidentity	mmehost	mmerealm	UE-Reachability
1	calisson.openair4G.eur	openair4G.eur	0

Figure 37. Configuration of mmeidentity table for 1 PC configuration.

Once all the files of the HSS have been configured, the next entity that we have to configure is the MME.

### MME configuration

The MME entity was configured by customizing the mme.conf and the mme\_fd.conf, respectively.

In the **mme.conf** configuration file were configured the following parameters according to the network parameters defined previously.

```

73 # ----- MME served GUMMEIs
74 # MME code DEFAULT size = 8 bits
75 # MME GROUP ID size = 16 bits
76 # MME_CODE = [ 1, 30 , 31, 32, 33, 34, 35, 36, 56 , 29 , 8 ];
77 # MME_GID = [ 32768 , 4 , 5 , 30 , 8 , 9 , 50021 ];
78 GUMMEI_LIST = (
79     {MCC="214" ; MNC="12"; MME_GID="4" ; MME_CODE="1"; }
80 );
81
82 # ----- MME served TAIs
83 # TA (mcc.mnc:tracking area code) DEFAULT = 208.34:1
84 # max values = 999.999:65535
85 # maximum of 16 TAIs, comma separated
86 # !!! Actually use only one PLMN
87 TAI_LIST = (
88     {MCC="214" ; MNC="12"; TAC = "1"; }
89 );

```

Figure 38. Configuration of GUMMEI and TAI List for 1 and 2 PCs configurations.

```

151 NETWORK_INTERFACES :
152 {
153     # MME binded interface for S1-C or S1-MME communication (S1AP), can be
    ethernet interface, virtual ethernet interface, we don't advise wireless
    interfaces
154     MME_INTERFACE_NAME_FOR_S1_MME      = "lo";                # YOUR
    NETWORK CONFIG HERE
155     MME_IPV4_ADDRESS_FOR_S1_MME        = "127.0.0.20/8";      # YOUR
    NETWORK CONFIG HERE
156
157     # MME binded interface for S11 communication (GTPV2-C)
158     MME_INTERFACE_NAME_FOR_S11_MME     = "lo";                # YOUR NETWORK
    CONFIG HERE
159     MME_IPV4_ADDRESS_FOR_S11_MME       = "127.0.0.20/8";      # YOUR NETWORK
    CONFIG HERE
160     MME_PORT_FOR_S11_MME               = 2123;                # YOUR NETWORK
    CONFIG HERE
161 };

204 S-GW_LIST_SELECTION = (
205     {ID="tac-lb01.tac-hb00.tac.epc.mnc091.mcc214.3gppnetwork.org" ;
      SGW_IPV4_ADDRESS_FOR_S11="127.0.0.30/8";}
206 );

```

**Figure 39. Configuration of network interfaces for 1 PC configuration.**

```

151 NETWORK_INTERFACES :
152 {
153     # MME binded interface for S1-C or S1-MME communication (S1AP), can be
    ethernet interface, virtual ethernet interface, we don't advise wireless
    interfaces
154     MME_INTERFACE_NAME_FOR_S1_MME      = "eth1";              # YOUR
    NETWORK CONFIG HERE
155     MME_IPV4_ADDRESS_FOR_S1_MME        = "192.68.1.3/24";      # YOUR
    NETWORK CONFIG HERE
156
157     # MME binded interface for S11 communication (GTPV2-C)
158     MME_INTERFACE_NAME_FOR_S11_MME     = "lo";                # YOUR NETWORK
    CONFIG HERE
159     MME_IPV4_ADDRESS_FOR_S11_MME       = "127.0.11.1/8";      # YOUR NETWORK
    CONFIG HERE
160     MME_PORT_FOR_S11_MME               = 2123;                # YOUR NETWORK
    CONFIG HERE
161 };

204 S-GW_LIST_SELECTION = (
205     {ID="tac-lb01.tac-hb00.tac.epc.mnc091.mcc214.3gppnetwork.org" ;
      SGW_IPV4_ADDRESS_FOR_S11="127.0.11.2/8";}
206 );

```

**Figure 40. Configuration of network interfaces for 2 PCs configuration.**

The parameters defined in this file were the GUMMEI and the TAI List, explained above, and the IP addresses and interfaces that communicate the MME with the other entities in the OAI LTE network (SPGW and eNB).

Then, the **mme\_fd.conf** configuration file was modified the following parameter.

```

3  # Uncomment if the framework cannot resolv it.
4  Identity = "calisson.openair4G.eur";
5  Realm = "openair4G.eur";

```

**Figure 41. Configuration of mme\_fd.conf for 1 PC configuration.**

```

3  # Uncomment if the framework cannot resolv it.
4  Identity = "nano.openair4G.eur";
5  Realm = "openair4G.eur";

```

**Figure 42. Configuration of mme\_fd.conf for 2 PCs configuration.**

The identity tells us what the MME is called and is formed by the name of the host, where the MME is implemented, and the realm, which refers to the domain of that host. In the configuration of 2 PCs the identity of the MME is nano.openair4G.eur whereas with 1 PC the identity of the MME is calisson.openair.4G.eur. These parameters must match the

mmeidentity parameter of the database so that the database can communicate with the MME.

The last entity to be configured in the EPC is the SPGW.

## SPGW configuration

In the SPGW entity there was just one configuration file to customize, the **spgw.conf**. In Figure 43 the parameters that were adjusted are shown.

```

23 NETWORK_INTERFACES :
24 {
25     # S-GW binded interface for S11 communication (GTPV2-C), if none selected
26     # the ITTI message interface is used
27     SGW_INTERFACE_NAME_FOR_S11 = "lo"; #
28     STRING, interface name, YOUR NETWORK CONFIG HERE
29     SGW_IPV4_ADDRESS_FOR_S11 = "127.0.0.30/8" #
30     STRING, CIDR, YOUR NETWORK CONFIG HERE
31
32     # S-GW binded interface for S1-U communication (GTPV1-U) can be ethernet
33     # interface, virtual ethernet interface, we don't advise wireless interfaces
34     SGW_INTERFACE_NAME_FOR_S1U_S12_S4_UP = "lo"; #
35     STRING, interface name, YOUR NETWORK CONFIG HERE, USE "lo" if S-GW run on
36     # eNB host
37     SGW_IPV4_ADDRESS_FOR_S1U_S12_S4_UP = "127.0.0.30/8"; #
38     STRING, CIDR, YOUR NETWORK CONFIG HERE
39     SGW_IPV4_PORT_FOR_S1U_S12_S4_UP = 2152; #
40     INTEGER, port number, PREFER NOT CHANGE UNLESS YOU KNOW WHAT YOU ARE DOING
41
42     # S-GW binded interface for S5 or S8 communication, not implemented, so
43     # leave it to none
44     SGW_INTERFACE_NAME_FOR_S5_S8_UP = "none"; #
45     STRING, interface name, DO NOT CHANGE (NOT IMPLEMENTED YET)
46     SGW_IPV4_ADDRESS_FOR_S5_S8_UP = "0.0.0.0/24"; #
47     STRING, CIDR, DO NOT CHANGE (NOT IMPLEMENTED YET)
48 }
49
105 UE_MTU = 1400

```

Figure 43. Configuration of spgw.conf for 1 PC configuration.

```

23 NETWORK_INTERFACES :
24 {
25     # S-GW binded interface for S11 communication (GTPV2-C), if none selected
26     # the ITTI message interface is used
27     SGW_INTERFACE_NAME_FOR_S11 = "lo"; #
28     STRING, interface name, YOUR NETWORK CONFIG HERE
29     SGW_IPV4_ADDRESS_FOR_S11 = "127.0.11.2/8" #
30     STRING, CIDR, YOUR NETWORK CONFIG HERE
31
32     # S-GW binded interface for S1-U communication (GTPV1-U) can be ethernet
33     # interface, virtual ethernet interface, we don't advise wireless interfaces
34     SGW_INTERFACE_NAME_FOR_S1U_S12_S4_UP = "eth1"; #
35     STRING, interface name, YOUR NETWORK CONFIG HERE, USE "lo" if S-GW run on
36     # eNB host
37     SGW_IPV4_ADDRESS_FOR_S1U_S12_S4_UP = "192.68.1.3/24"; #
38     STRING, CIDR, YOUR NETWORK CONFIG HERE
39     SGW_IPV4_PORT_FOR_S1U_S12_S4_UP = 2152; #
40     INTEGER, port number, PREFER NOT CHANGE UNLESS YOU KNOW WHAT YOU ARE DOING
41
42     # S-GW binded interface for S5 or S8 communication, not implemented, so
43     # leave it to none
44     SGW_INTERFACE_NAME_FOR_S5_S8_UP = "none"; #
45     STRING, interface name, DO NOT CHANGE (NOT IMPLEMENTED YET)
46     SGW_IPV4_ADDRESS_FOR_S5_S8_UP = "0.0.0.0/24"; #
47     STRING, CIDR, DO NOT CHANGE (NOT IMPLEMENTED YET)
48 }
49
105 UE_MTU = 1400

```

Figure 44. Configuration of spgw.conf for 2 PCs configuration.

According to the screenshots of the code, we had to vary the IP addresses and interfaces based on the configuration of our PCs. Since the SGW and PGW components act as one, the s5 / s8 interface was defined as "none", that is, null. We also modify the IP address and interface that connects the LTE network with the external network. Finally, we changed the value of the MTU of UE to 1400 so that it does not cause any problem

with the transmission and reception of packets and the user can connect correctly to the network.

The next step, after having configured all CN entities, was the configuration of the eNB.

#### 4.2.4. eNB configuration

The eNB has a large number of configurations based on the frequency band in which it works and the SDR used. In our case, since we work in **band 7** and use the **USRPB210** we configured the file **enb.band7.tm1.usrbp210.conf**. In Figures 45, 46 and 47 the modified parameters are shown.

```
15      // Tracking area code, 0x0000 and 0xfffe are reserved values
16      tracking_area_code = "1";
17
18      mobile_country_code = "214";
19
20      mobile_network_code = "91";
```

**Figure 45. Configuration of network parameters for 1 and 2 PCs configuration.**

```
138      ////////// MME parameters:
139
140      mme_ip_address      = ( { ipv4      = "127.0.0.20";
141                               ipv6      = "192:168:30::17";
142                               active    = "yes";
143                               preference = "ipv4";
144                               }
145      );

147      NETWORK_INTERFACES :
148      {
149
150          ENB_INTERFACE_NAME_FOR_S1_MME      = "lo";
151          ENB_IPV4_ADDRESS_FOR_S1_MME        = "127.0.0.10/8";
152          ENB_INTERFACE_NAME_FOR_S1U         = "lo";
153          ENB_IPV4_ADDRESS_FOR_S1U           = "127.0.0.10/8";
154          ENB_PORT_FOR_S1U                    = 2152; # Spec 2152
155      };
```

**Figure 46. Configuration of MME IP address and network interfaces for 1 PC configuration.**

```
138      ////////// MME parameters:
139
140      mme_ip_address      = ( { ipv4      = "192.68.1.3";
141                               ipv6      = "192:168:30::17";
142                               active    = "yes";
143                               preference = "ipv4";
144                               }
145      );

147      NETWORK_INTERFACES :
148      {
149
150          ENB_INTERFACE_NAME_FOR_S1_MME      = "eth0";
151          ENB_IPV4_ADDRESS_FOR_S1_MME        = "192.68.1.5/24";
152          ENB_INTERFACE_NAME_FOR_S1U         = "eth0";
153          ENB_IPV4_ADDRESS_FOR_S1U           = "192.68.1.5/24";
154          ENB_PORT_FOR_S1U                    = 2152; # Spec 2152
155      };
```

**Figure 47. Configuration of MME IP address and network interfaces for 2 PCs configuration.**

In this file we first had to change the parameters of the MCC and MNC network, according to the defined parameters of our network. Next, we had to define the address in ipv4 and ipv6 format of the MME and the IP addresses of the interfaces that connect the eNB with the CN entities. It is important to verify that the parameters entered are the same as those entered in the other configuration files.

Once all the elements of the network were configured, we performed connectivity tests.



#### 4.2.5. Launch of the OAI platform

The start-up of the network was carried out with the launch of the different elements of the network. The start-up of the network was carried out with the launch of the different elements of the network. Through Linux consoles we accessed the directories of the CN (openair-cn) and the eNB (openairinterface5g) and launched the processes. In the first place, the HSS was run, then the MME and finally the SPGW. Next, the eNB was run. In Figures 48 and 49, an image of the different elements of the network in progress for the two configurations of PCs is presented.

```

root@nano: /opt/openair-cn-mme_scenario/SCRIPTS
usuarlogrcm@nano:~$ sudo su
[sudo] password for usuarlogrcm:
root@nano:/home/usuarlogrcm# cd /opt/openair-cn-mme_scenario/SCRIPTS
root@nano:/opt/openair-cn-mme_scenario/SCRIPTS# ./run_hss
OPENAIRCN_DIR = /opt/openair-cn-mme_scenario
===== EURECOM OPENAIR-HSS vBranch: Abrev. Hash: Date: =====
Please report any bug to: openaircn-user@lists.eurecom.fr

Parsing configuration file: /usr/local/etc/oai/hss.conf
Configuration
* Global:
- File .....: /usr/local/etc/oai/hss.conf
* MYSQL:
- Server .....: 127.0.0.1
- Database .....: oai_db
- User .....: root
- Password .....: *****
* FreeDiameter:
- Conf file .....: /usr/local/etc/oai/freediameter/hss_fd.conf
* Security:
- Operator key .....: *****
- Random .....: true
Initializing db layer
Initializing db layer: DONE

root@nano: /opt/openair-cn-mme_scenario/SCRIPTS
usuarlogrcm@nano:~$ sudo su
[sudo] password for usuarlogrcm:
root@nano:/home/usuarlogrcm# cd /opt/openair-cn-mme_scenario/SCRIPTS
root@nano:/opt/openair-cn-mme_scenario/SCRIPTS# ./run_mme
OPENAIRCN_DIR = /opt/openair-cn-mme_scenario
Initializing shared logging
Initializing shared logging Done
Initializing OAI Logging
Initializing OAI logging Done
mme_config_parse_opt_line mme_config.config_file /usr/local/etc/oai/mme.conf
000017 00000:236507 7f2d9550740 INFO CONFIG cenario/SRC/MME_APP/mme_config.c:0
017 ===== EURECOM MME vBranch: Abrev. Hash: Date: =====
000018 00000:236520 7f2d9550740 DEBUG CONFIG cenario/SRC/MME_APP/mme_config.c:0
019 Built with CMAKE_BUILD_TYPE .....: Debug
000019 00000:236525 7f2d9550740 DEBUG CONFIG cenario/SRC/MME_APP/mme_config.c:0
020 Built with DISABLE_ITTI_DETECT_SUB_TASK_ID ..: 1
000020 00000:236529 7f2d9550740 DEBUG CONFIG cenario/SRC/MME_APP/mme_config.c:0
021 Built with ENABLE_ITTI .....: 1
000021 00000:236535 7f2d9550740 DEBUG CONFIG cenario/SRC/MME_APP/mme_config.c:0
022 Built with ENABLE_ITTI_ANALYZER .....: 0
000022 00000:236540 7f2d9550740 DEBUG CONFIG cenario/SRC/MME_APP/mme_config.c:0
023 Built with ITTI_TASK_STACK_SIZE .....: 2097152
000023 00000:236544 7f2d9550740 DEBUG CONFIG cenario/SRC/MME_APP/mme_config.c:0
024 Built with ITTI_LITE .....: 0

root@nano: /opt/openair-cn-mme_scenario/SCRIPTS
176 Inserting new descriptor for task 7, sd 35
000111 00000:635694 7f2e25e30700 DEBUG UDP /SRC/UDP/udp_primitives_server.c:0
192 Received 1 events
rmmod: ERROR: Module gtp is not currently loaded
000112 00000:650732 7f2e2a763740 NOTICE GTPv1- rio/SRC/GTPv1-U/gtp_mod_kernel.c:0
006 Using the GTP kernel mode (genl ID is 26)
000113 00000:653687 7f2e2a763740 DEBUG GTPv1- rio/SRC/GTPv1-U/gtp_mod_kernel.c:0
109 Setting route to reach UE net 172.16.0.0 via gtp0
000114 00000:653734 7f2e2a763740 NOTICE GTPv1- rio/SRC/GTPv1-U/gtp_mod_kernel.c:0
116 GTP kernel configured
000115 00000:653763 7f2e2a763740 DEBUG GTPv1- enario/SRC/GTPv1-U/gtpvlu_task.c:0
143 Initializing GTPVLU interface: DONE
000116 00000:689877 7f2e2a763740 INFO SPGW-A rio/SRC/SGW/pgw_pcef_emulation.c:0
263 Loading PCC rule TEST_PING_PCC_RULE
000117 00000:689910 7f2e2a763740 INFO SPGW-A rio/SRC/SGW/pgw_pcef_emulation.c:0
263 Loading PCC rule DEFAULT_PCC_RULE
000118 00000:690152 7f2e2a763740 DEBUG SPGW-A -mme_scenario/SRC/SGW/sgw_task.c:0
226 Initializing SPGW-APP task interface: DONE
000119 00000:692810 7f2e26631700 DEBUG CMD cenario/SRC/UTILS/async_system.c:0
081 C system() call: iptables -t nat -I POSTROUTING -s 172.16.0.0/12 -o eth0
! --protocol sctp -j SNAT --to-source 147.83.105.253
000120 00000:699000 7f2e26631700 DEBUG CMD cenario/SRC/UTILS/async_system.c:0
081 C system() call: sysctl -w net.ipv4.tcp_ecn=1
net.ipv4.tcp_ecn = 1
  
```

Figure 48. Launching the OAI LTE network in 2 PCs configuration. HSS (top left), MME (top right), SPGW (bottom left) and eNB (bottom right).

```

root@nano: /opt/openair-cn-mme_scenario/SCRIPTS
usuarlogrcm@nano:~$ sudo su
[sudo] password for usuarlogrcm:
root@nano:/home/usuarlogrcm# cd /opt/openair-cn-mme_scenario/SCRIPTS
root@nano:/opt/openair-cn-mme_scenario/SCRIPTS# ./run_sgw
OPENAIRCN_DIR = /opt/openair-cn-mme_scenario
Initializing shared logging
Initializing shared logging Done
Initializing OAI Logging
Initializing OAI logging Done
sgw_config_parse_opt_line sgw_config.config_file /usr/local/etc/oai/sgw.conf
000017 00000:236507 7f2d9550740 INFO CONFIG cenario/SRC/SGW/sgw_config.c:0
017 ===== EURECOM SGW vBranch: Abrev. Hash: Date: =====
000018 00000:236520 7f2d9550740 DEBUG CONFIG cenario/SRC/SGW/sgw_config.c:0
019 Built with CMAKE_BUILD_TYPE .....: Debug
000019 00000:236525 7f2d9550740 DEBUG CONFIG cenario/SRC/SGW/sgw_config.c:0
020 Built with DISABLE_ITTI_DETECT_SUB_TASK_ID ..: 1
000020 00000:236529 7f2d9550740 DEBUG CONFIG cenario/SRC/SGW/sgw_config.c:0
021 Built with ENABLE_ITTI .....: 1
000021 00000:236535 7f2d9550740 DEBUG CONFIG cenario/SRC/SGW/sgw_config.c:0
022 Built with ENABLE_ITTI_ANALYZER .....: 0
000022 00000:236540 7f2d9550740 DEBUG CONFIG cenario/SRC/SGW/sgw_config.c:0
023 Built with ITTI_TASK_STACK_SIZE .....: 2097152
000023 00000:236544 7f2d9550740 DEBUG CONFIG cenario/SRC/SGW/sgw_config.c:0
024 Built with ITTI_LITE .....: 0

root@nano: /opt/openair-cn-mme_scenario/SCRIPTS
usuarlogrcm@nano:~$ sudo su
[sudo] password for usuarlogrcm:
root@nano:/home/usuarlogrcm# cd /opt/openair-cn-mme_scenario/SCRIPTS
root@nano:/opt/openair-cn-mme_scenario/SCRIPTS# ./run_eNB
OPENAIRCN_DIR = /opt/openair-cn-mme_scenario
Initializing shared logging
Initializing shared logging Done
Initializing OAI Logging
Initializing OAI logging Done
eNB_config_parse_opt_line eNB_config.config_file /usr/local/etc/oai/eNB.conf
000017 00000:236507 7f2d9550740 INFO CONFIG cenario/SRC/eNB/eNB_config.c:0
017 ===== EURECOM eNB vBranch: Abrev. Hash: Date: =====
000018 00000:236520 7f2d9550740 DEBUG CONFIG cenario/SRC/eNB/eNB_config.c:0
019 Built with CMAKE_BUILD_TYPE .....: Debug
000019 00000:236525 7f2d9550740 DEBUG CONFIG cenario/SRC/eNB/eNB_config.c:0
020 Built with DISABLE_ITTI_DETECT_SUB_TASK_ID ..: 1
000020 00000:236529 7f2d9550740 DEBUG CONFIG cenario/SRC/eNB/eNB_config.c:0
021 Built with ENABLE_ITTI .....: 1
000021 00000:236535 7f2d9550740 DEBUG CONFIG cenario/SRC/eNB/eNB_config.c:0
022 Built with ENABLE_ITTI_ANALYZER .....: 0
000022 00000:236540 7f2d9550740 DEBUG CONFIG cenario/SRC/eNB/eNB_config.c:0
023 Built with ITTI_TASK_STACK_SIZE .....: 2097152
000023 00000:236544 7f2d9550740 DEBUG CONFIG cenario/SRC/eNB/eNB_config.c:0
024 Built with ITTI_LITE .....: 0
  
```



**Figure 49. Launching the OAI LTE network in 1 PC configuration. HSS (top left), MME (top right), SPGW (bottom left) and eNB (bottom right).**

As we were launching the different entities we could see how the different interfaces between entities were being configured, how the HSS was able to connect to the database and how the eNB integrated perfectly with the CN without producing any errors. One more indicator of the good functioning of the network is the message shown in the console of the eNB (Figure 50) that informs us that this entity is ready for the connection of users.

```

Creating te_thread
waiting for sync (eNB_thread_single)
[PMW][I]thread created (id=4811[HW][I])[SCHED][eNB] eNB_thread_prach started on CPU 1 TID 4812, sched_policy = SCHED_FIFO , priority = 99, CPU Affinity= CPU_0_CPU_1_CPU_2_CPU_3
[HW][I][SCHED][eNB] eNB_thread_synch started on CPU 0 TID 4813, sched_policy = SCHED_FIFO , priority = 99, CPU Affinity= CPU_0_CPU_1_CPU_2_CPU_3
waiting for sync (eNB_thread_synch)
Setting eNB buffer to all-RX
Sending sync to all threads
TYPE <CTRL-C> TO TERMINATE
Entering ITTI signals handler
got sync (eNB_thread_single)
got sync (eNB_thread_synch)

```

**Figure 50. eNB is ready for users connection for 1 and 2 PCs configuration.**

The last step to verify that the network was reachable by the mobile devices was to test its connectivity.

#### 4.2.6. Connectivity tests

There are many applications capable of detecting existing networks in our environment. One of them is RFBENCHMARK that tells us which is the best operator based on our location, as well as being able to perform different efficiency tests.

To detect if the network was reachable from any mobile device, we downloaded this application and entered its initial window, where it indicates, at the top, to which network we were connected and, at the bottom, which was the best operator of the network zone. We also entered the Test Efficiency tab and detected the list of operators available at that time.

Two images are shown below with screenshots of the application in which it is perceived how the mobile device detected the implemented network, 21491. In this way, we can confirm that the network was well implemented.

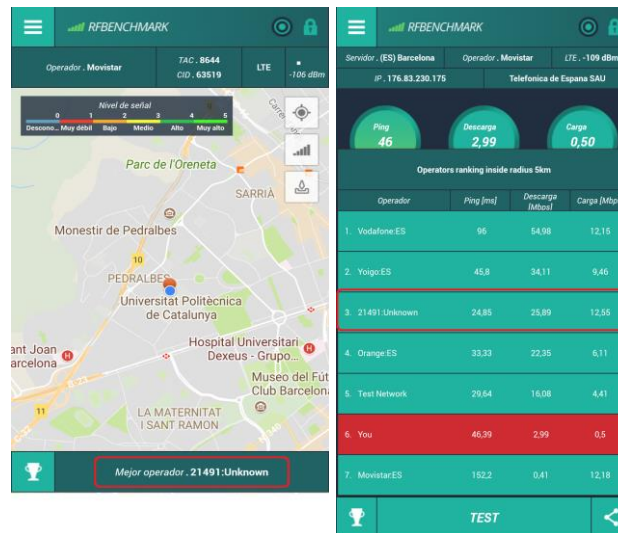


Figure 51. RFBENCHMARK: Availability of 21491 network.

Another application that helped us detect the network was Qualipoc, which provided us with a large amount of information about the network: quality parameters, power parameters, cell information, and communications technology, among others.



Figure 52. Qualipoc: Analysis of the 21491 network.

As seen in Figure 52, the 21491 network was reachable and had good signal power and quality.

The last step was to verify that users could connect to the 21491 network and could access the Internet. To make these tests we used the Samsung Galaxy III, the Sony Xperia Z5 and the Huawei Dongle with the different programmed SIM cards.

When the network was running we had to introduce the SIM card in the mobile devices and turn them on, in the case of mobile phones, or connect it to a PC, in the case of the Dongle. As long as they were connected to the network we had to configure the name of the APN. If we remember in the section of database configuration in the pdn table we defined the parameter, oai.ipv4. It has to match with the database value. In the mobiles,

The image is a composite of two side-by-side screenshots. The left screenshot shows a Samsung smartphone screen with the 'APNs' (Access Point Names) settings. The status bar at the top shows 'SAMSUNG', signal strength, and the time '6:32 PM'. The APN list shows 'grom.upc' as the selected profile. The right screenshot shows the 'Telenor Mobile Partner' software window. The 'Options' dialog box is open, and the 'Dial-Up' tab is selected. In the 'Dial-Up' tab, the 'Profile Name' is 'csl ipv4 (Default)', the 'APN' is 'grom', and the 'Authentication' is set to 'Static'. The 'Access number' is '9999', the 'User name' is empty, and the 'Password' is empty. The 'Advanced...' button is visible at the bottom of the 'Dial-Up' tab. The main window of the software shows a status bar at the bottom with '4G E91', a speed indicator '0.00 kbps', and a download icon.

Once we configured the devices we were able to see that the HSS detected the device that was connecting (Figure 54) and that the eNB (Figure 55) authenticated the user and allowed their connection, showing the following messages in both PC configurations and for the different SIM cards programmed.

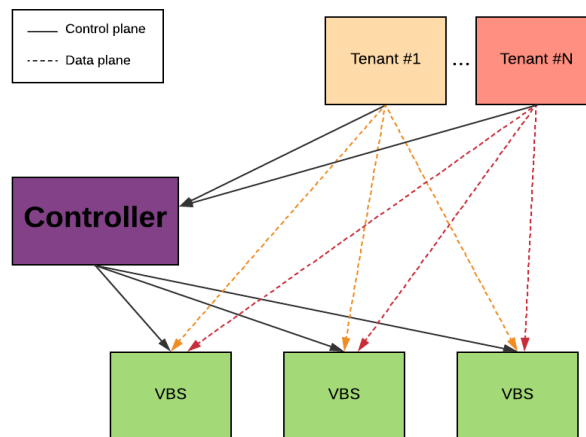
[illegible][illegible]

49

From the following section, whenever we talk about the OAI network we will refer to the configuration of 1 PC.

### 4.3. 5G-EmPOWER platform

In addition to the OAI platform that implements an LTE network; we need software capable of applying SDN technology that is separating the control plane and data plane. 5G-EmPOWER is a software that allows us to apply this technology from the implementation of a controller that will be responsible for managing the multiple slices or tenants over a common physical infrastructure and, consequently, their resources. To make this platform more understandable we are going to present a scheme with the complete system.



**Figure 56. Scheme of the 5G-EmPOWER complete system.**

The elements that make up the architecture of this platform are the following:

- **Tenants or MVNOs** are virtual operators that provide a service to the network users that are associated with that tenant. They have a guaranteed Service Level Agreement (SLA) in the network.
- **Controller** is in charge of managing tenants and their resources among the different VBS assigned to them, guaranteeing that SLA is accomplished.
- **VBSes** are access points of the LTE standard that allow the user to connect to the network.

An element to emphasize about this platform is the web interface, programmable at high level that allows the creation of tenants and the association of VBS or other wireless termination points.

In order to access the web interface, we had to install the empower-runtime software, which implements the controller, together with all the corresponding libraries, as indicated in [31]. Next, we verified that this software worked correctly. For this we checked if once the controller was running the initialization messages appeared in Figure 57.

```

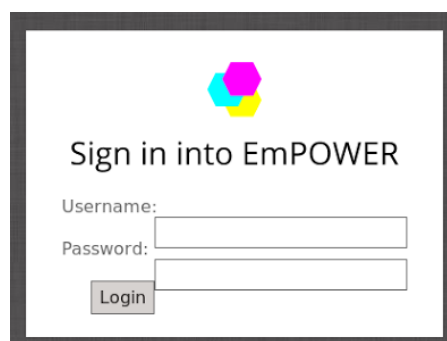
user@kali:~/opt/update/empower-runtime$
user@kali:~/opt/update/empower-runtime$ sudo ./opt/
user@kali:~/opt/update/empower-runtime$ sudo ./empower-runtime.py
[sudo] password for user@kali:
INFOcore:Starting EMPWR8 Runtime
INFOcore:Loading EMPWR8 Runtime defaults
INFOroot:Importing module: empower.restserver.restserver
INFOroot:Importing module: empower.lvnfp.lvnfpserver
INFOroot:Importing module: empower.lvapp.lvappserver
INFOroot:Importing module: empower.vbisp.vbispserver
INFOroot:Importing module: empower.energlosserver.energlosserver
INFOroot:Importing module: empower.lvapp_stats.lvapp_stats
INFOroot:Importing module: empower.uw_conf.uw_ccr_mwms_conf
INFOroot:Importing module: empower.uw_stats.uw_ccr_stats
INFOroot:Importing module: empower.vbs_stats.vbs_cell_stats
INFOroot:Importing module: empower.vbisp.uw_handler
INFOroot:Importing module: empower.events.wdpm3
INFOroot:Importing module: empower.events.wtsp
INFOroot:Importing module: empower.events.lvplleave
INFOroot:Importing module: empower.events.lvapjoin
INFOroot:Importing module: empower.events.uwjoin
INFOroot:Importing module: empower.bln_counter.bln_counter
INFOroot:Importing module: empower.wbp_bln_counter.wbp_bln_counter
INFOroot:Importing module: empower.tsp_bln_counter.tsp_bln_counter
INFOroot:Importing module: empower.cdn_links.cdn_links
INFOroot:Importing module: empower.maps.ucgn
INFOroot:Importing module: empower.maps.icge
INFOroot:Importing module: empower.maps.business
INFOroot:Importing module: empower.triggers.raai
INFOroot:Importing module: empower.triggers.summary
INFOroot:Importing module: empower.triggers.business
INFOroot:Importing module: empower.events.cdpdcm
INFOroot:Importing module: empower.events.ccppp
INFOroot:Importing module: empower.events.vbisp
INFOroot:Importing module: empower.events.lvnfjoin
INFOroot:Importing module: empower.events.lvnfleave
INFOroot:Importing module: empower.lvnf_ems.lvnf_get
INFOroot:Importing module: empower.lvnf_ems.lvnf_set
INFOroot:Importing module: empower.lvnf_stats.lvnf_stats
INFOcore:Registering 'empower.restserver.restserver'
INFOrestserver:REST Server available at 8888
INFOcore:Registering 'empower.lvnfp.lvnfpserver'
INFOcore:pfpsvrserver:LVNF Server available at 4432
INFOcore:Registering 'empower.lvapp.lvappserver'
INFOcore:pfpsvrserver:LVAP Server available at 4433
INFOcore:Registering 'empower.vbisp.vbispserver'
INFOcore:pfpsvrserver:VSP Server available at 2218
INFOcore:Registering 'empower.energlosserver.energlosserver'
INFOenerglosserver:Energino Server available at 5533
INFOcore:Registering 'empower.intentserver.intentserver'
INFOintentserver:Intent GET /intent/rules
ERROR:intentserver:Intent Interface not found
INFOintentserver:Intent GET /intent/poa
ERROR:intentserver:Intent Interface not found

```

**Figure 57. Initialization of the controller.**

#### 4.3.1. 5G-EmPOWER web interface

We access to the web interface by introducing in the browser the following URL <http://147.83.105.233:8888/>. Once we pressed enter, the following window appeared in which one user and password were required.



**Figure 58. 5G-EmPOWER web interface.**

There are three users to access the web interface of the controller. One of them with administrator permissions with username and password "root" and two with regular user permissions with usernames "foo" and "bar" and passwords "foo" and "bar", respectively. The functions of each type of user are explained below.

Accessing with administrator permissions we found the following screen formed by different tabs.



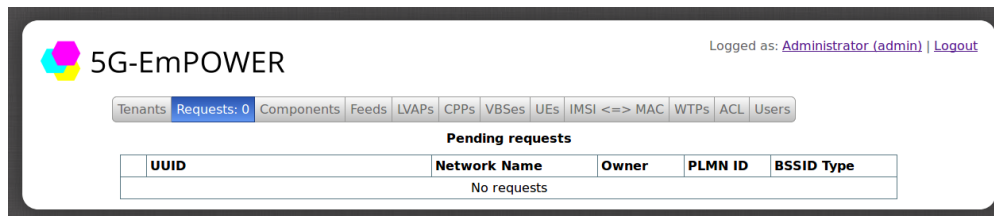


**Figure 59. Main window login as administrator. Tenants tab.**

Given that the LTE standard is in the focus of this work, we are going to talk about the tabs with which we have worked on the project (framed in red). The rest of the tabs are used to implement the slicing on the Wi-Fi standard.

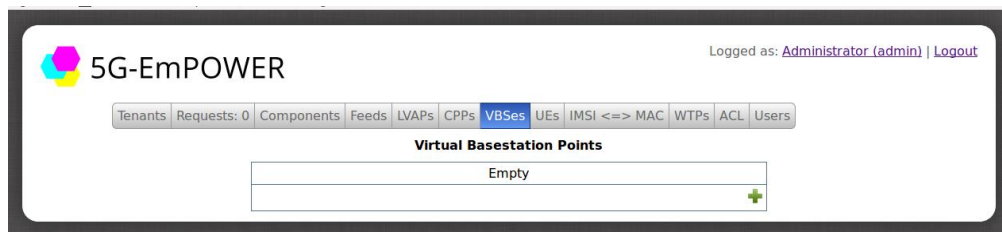
The tabs and their functionality are presented below.

- **Tenants:** This tab shows the active tenants, that is, those tenants that have been accepted by the administrator. In Figure 59 this tab is presented.
- **Requests:** In this tab the tenants that are requested by the users but have not been accepted by the administrator are exposed. In Figure 60 this tab is presented.



**Figure 60. Requests tab.**

- **VBSes:** In this tab the Virtual Base Stations that will associate to the controller and the tenants are created. To create a VBS the name and MAC address of our eNB should be introduced. In Figure 61 this tab is presented.



**Figure 61. VBSes tab.**

- **UEs:** This tab shows the connected users that the controller has detected. For each user, the VBS, the IMSI, the PLMN Id and a Radio Network Temporary Identifier (RNTI) are presented. The latter is used to identify information dedicated to a particular subscriber on the radio interface. In Figure 62 this tab is presented.

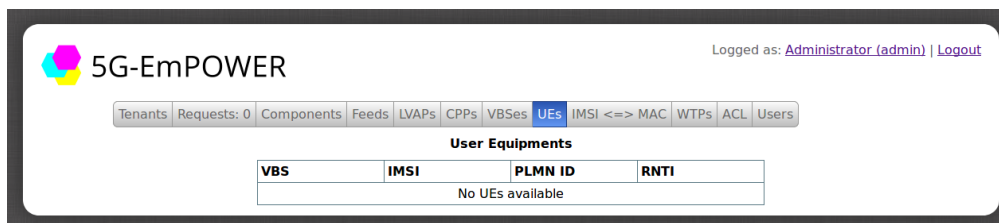


Figure 62. UEs tab.

- **IMSI < = > MAC:** In this tab, the IMSI of the SIM cards is mapped to the MAC of the mobile devices. In Figure 63 this tab is presented.

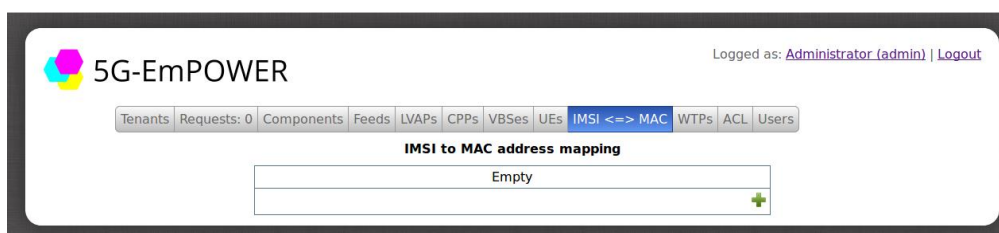


Figure 63. IMSI to MAC address mapping tab.

- **Users:** This tab defines the existing users: root, bar and foo. In Figure 64 this tab is presented.

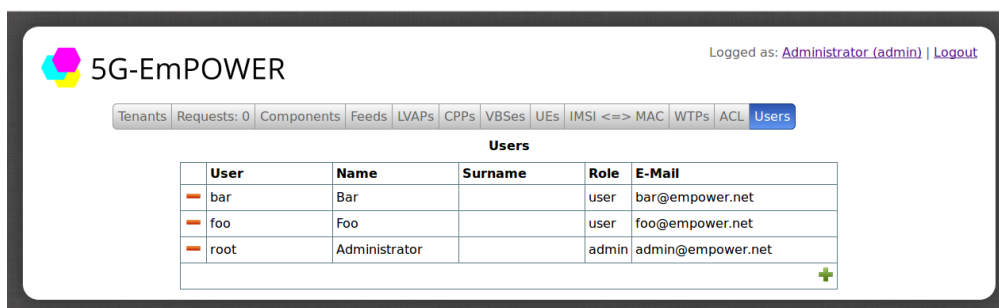


Figure 64. Users tab.

Accessing as a regular user, through the users "foo" or "bar" we found the following screen.

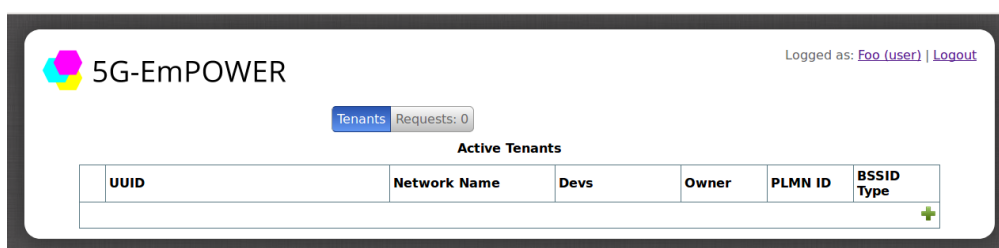


Figure 65. Main window login as regular user. Tenants tab.

With the permission of a regular user there are the following tabs.

- **Tenants:** In this tab the requested tenants are created and active tenants are seen, that is, those that have been accepted by the administrator. To create a

request, click on the + sign and fill in the data of the window that opens (Figure 66).

Figure 66. Tab to create of new tenants.

- **Requests:** This tab shows the tenants that have been created by users and are waiting to be accepted by the administrator.

UUID	Network Name	Owner	PLMN ID	BSSID Type
No requests				

Figure 67. Pending requests tab.

#### 4.4. Integration of OAI and 5G-EmPOWER platforms

At this point in the project we already had the two platforms ready to be integrated and we knew perfectly how to configure them to adapt to our requirements. The next step was to find how to make it join these platforms and if the software we had was appropriate for that. In Figure 68 are shown the 2 platforms previously to be integrated. The entities and interfaces that have already been configured are represented with solid lines while the entities and interfaces that must be configured or updated are represented with dashed lines.

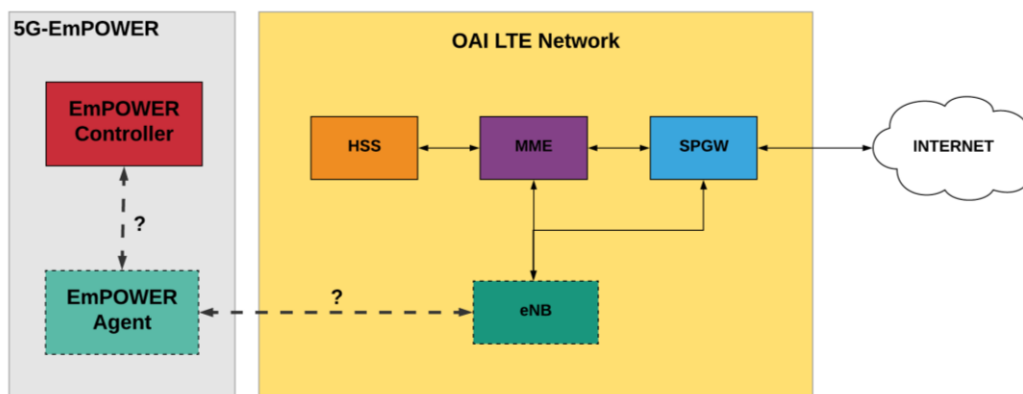


Figure 68. OAI and 5G-EmPOWER platforms before to be integrated.

After investigating the documentation of the 5G-EmPOWER platform [22], we found that we needed to include a new software called Agent. This software contains a block called



wrapper that translates what eNB and controller are saying so they can talk and understand each other.

The next question that we had to solve is whether the eNB OAI was able to communicate with the 5G-EmPOWER Agent with the actual software version of eNB. After documenting, we saw that it was not possible and that the eNB had to be customized to be able to connect to the Agent. Fortunately, FuN group, developers of 5G-EmPOWER, created a customized eNB from the code of OAI eNB named empower-openairinterface, by implementing a new folder (emage\_tech\_oai) that allows the interaction of the eNB with the Agent.

Once the new version of the eNB was downloaded and installed, we configured the network parameters and interfaces as we made in the previous version of 1 PC and enabled the option of the Agent and the controller in the **CMakefile.txt** file. Next, we compiled the code again to save the changes.

The last step to integrate both platforms was to configure the **agent.conf**, located in the /etc/empower directory.

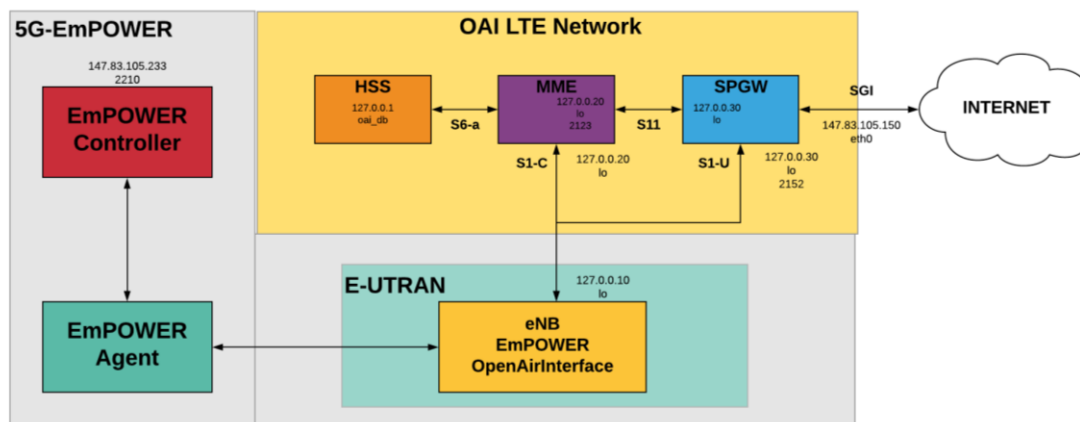
To configure this file was just needed to know the IP address where the controller was installed and the port where it was listening. The IP address is the host address of the controller (147.83.105.233) while the port (2210) is obtained from the information messages that come out when the controller is initialized. In Figure 69 this messages are shown.

```
INFO:root:Importing module: empower.events.vbsup
INFO:root:Importing module: empower.events.lvnfjoin
INFO:root:Importing module: empower.events.lvnfleave
INFO:root:Importing module: empower.lvnf_ems.lvnf_get
INFO:root:Importing module: empower.lvnf_ems.lvnf_set
INFO:root:Importing module: empower.lvnf_stats.lvnf_stats
INFO:core:Registering 'empower.restserver.restserver'
INFO:restserver:REST Server available at 8888
INFO:core:Registering 'empower.lvnfp.lvnfpserver'
INFO:core.pnfpserver:LVNF Server available at 4422
INFO:core:Registering 'empower.lvapp.lvappserver'
INFO:core.pnfpserver:LVAP Server available at 4433
INFO:core:Registering 'empower.vbsp.vbspserver'
INFO:core.pnfpserver:VBSP Server available at 2210
INFO:core:Registering 'empower.energinoserver.energinoserver'
INFO:energinoserver:Energino Server available at 5533
INFO:core:Registering 'empower.intentserver.intentserver'
```

**Figure 69. Initialization controller messages to identify VBSP port.**

Since we work with a 4th generation network, the VBSes are the access point of our network so the server available for these is on port 2210, which is the port that we configured.

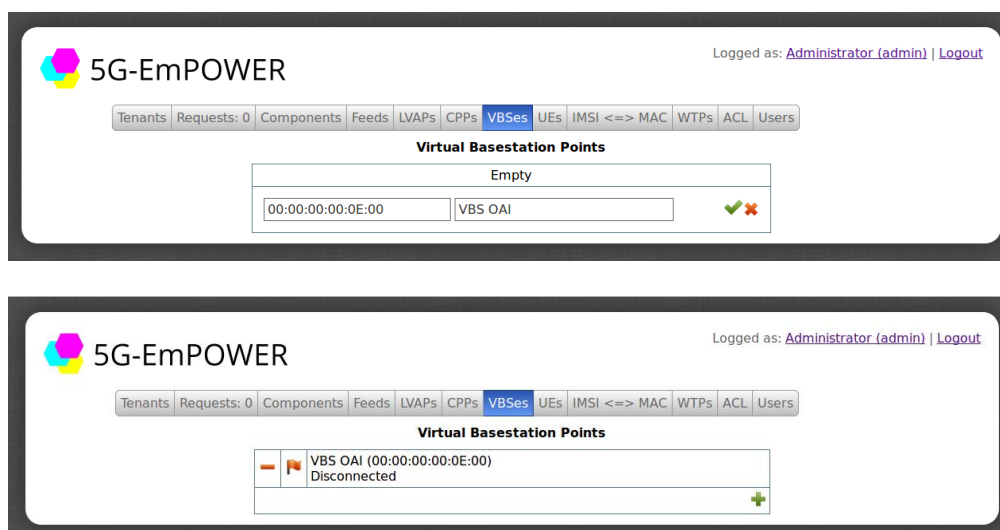
The definitive architecture of the network is presented in Figure 70. In this, network interfaces, host addresses and Agent settings have been specified.



**Figure 70. Integration of the OAI and 5G-EmPOWER platforms.**

The next step in the project was to verify that the integration had been done correctly. For this was necessary to configure the VBS in the 5G-EmPOWER web interface and make some connectivity tests.

To configure the VBS we had to go to the web interface of 5G-EmPOWER and enter with the administrator account. In the VBS tab we had to click on the **+** icon (see Figure 61). Then, we had to introduce a name for the VBS, in our case VBS OAI, and specify the MAC of the eNB with which we wanted to associate the VBS. This parameter can be found in the eNB configuration file (enb.band7.tm1.usrpb210.conf) and is the identifier of the eNB (eNB\_id = 0xe0). In the file of the eNB this identifier is expressed in hexadecimal format so we had to convert it into MAC address, obtaining the address shown in Figure 71.



**Figure 71. Creating VBS.**

Next to the parameters of the VBS created a red flag is shown. This indicator is used to inform about the state of the VBS. When the flag is red means that the LTE network is not running or that the controller cannot identify and connect with this VBS while the green flag represents that the LTE network is running and identified by the controller.

Next, we proceeded to perform the first connectivity tests between the OAI network and the 5G-EmPOWER platform. The procedure to establish a connection is as follows.

1. Launch the process of the controller through SSH or directly from the controller host. Remember that it has to be launched with administrator permissions to be able to access all the options of the web interface.
2. Launch EPC entities in the following order: HSS, MME and SPGW. In this way, all the interfaces between the EPC entities would be established.
3. Finally, launch the process of the eNB.

It is important to activate the controller before the network is running because otherwise the eNB is constantly trying to establish a connection with an element and several error messages appear.

To verify that the 2 platforms well were correctly integrated and that the VBS was well associated with the OAI eNB, we had to obtain three messages in the controller console, as indicated in the documentation of 5G-EmPOWER [31].

- Incoming connection message from the host where the eNB was implemented.
- Hello message from the eNB.
- Message in which the controller sends a UEs message to the VBS.

After launching all the processes we obtained the following result in the controller console.

```
rcm@empowerlte: /opt/update/empower-runtime
usuariogrcm@empowerlte: /opt/updat... x root@calisson: /opt/openair-cn-new/SC... x root@calisson: /opt/openair-cn-new/SC... x root@calisson: /opt/openair-cn-new/SC... x
INFO:core.pnfpserver:Incoming connection from ('147.83.105.150', 50860)
INFO:vbsp.vbspconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 0
INFO:vbsp.vbspconnection:Sending VBS (eNB) cells conf request to VBS 00:00:00:00:0E:00 (3584)
head {
  vers: 1
  b_id: 3584
  seq: 1
  t_id: 0
}
se {
  mENB_cells {
    req {
      enb_info_types: 1
    }
  }
}
```

**Figure 72. eNB is associated to the controller.**

The messages obtained indicated that the connection between the controller and the eNB had been established and therefore the integration of both platforms had been successful. On the other hand, in the web interface we can see in the VBS tab that the created VBS had a green flag, indicating that the controller was detecting that the VBS was running and therefore they were connected.

#### **4.5. Implementation of RAN Slicing and users connection**

The last procedure that we carried out was the creation of slices or tenants to implement the RAN slicing. By having a single operator we could only create slices with the PLMN Id of that network.

The next steps are followed to create a tenant [32]:

1. From a user account, foo or bar, click on the **+** button. The next step was determining the main parameters of the tenant:

- Assigning a name and optionally a description.
- Defining the Basic Service Set Identifier (BSSID) that is a unique identification name of all the packets of a wireless network to identify them as part of that network.
- Defining the PLMN Id of the network, which as we already know is the combination of the MCC and the MNC. In our case, this is 21491.

5G-EmPOWER

Logged as: [Foo \(user\)](#) | [Logout](#)

**Request new Tenant**

Network name:	EMPOWER1
Description:	SLICE1
BSSID Type:	<input checked="" type="radio"/> Unique <input type="radio"/> Shared
PLMN ID:	21491

[Request Tenant](#)

Figure 73. Creation of a new tenant.

- The next step was accepting the tenant from the administrator account in the Requests tab.

5G-EmPOWER

Logged as: [Administrator \(admin\)](#) | [Logout](#)

Tenants **Requests: 1** Components Feeds LVAPs CPPs VBSes UEs IMSI <=> MAC WTPs ACL Users

**Pending requests**

UUID	Network Name	Owner	PLMN ID	BSSID Type
<a href="#">7056f59f-cd50-4604-9744-231de5044bd2</a>	EMPOWER1	foo	21491	unique

Figure 74. Accepting the new tenant.

- Once the tenant was accepted, it becomes an active tenant.

5G-EmPOWER

Logged as: [Foo \(user\)](#) | [Logout](#)

**Tenants** Requests: 0

**Active Tenants**

UUID	Network Name	Devs	Owner	PLMN ID	BSSID Type
<a href="#">7056f59f-cd50-4604-9744-231de5044bd2</a>	EMPOWER1		foo	21491	unique

Figure 75. The new tenant is active.

- Finally, we clicked on the UUID and in the VBS tab we associated the VBS created before to this tenant.

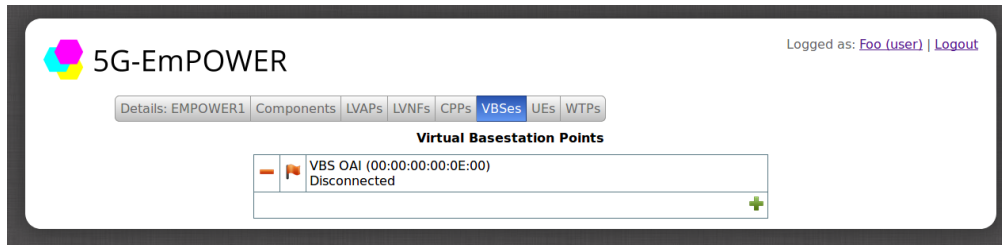
5G-EmPOWER

Logged as: [Foo \(user\)](#) | [Logout](#)

Details: EMPOWER1 Components LVAPs LVNFs CPPs **VBSes** UEs WTPs

**Virtual Basestation Points**

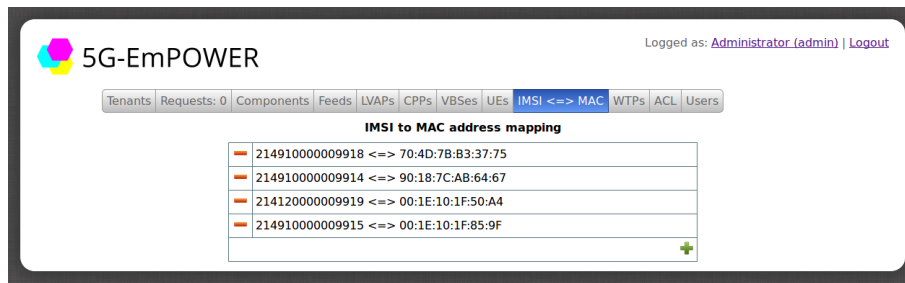
Select VBS:
00:00:00:00:00:00 (VBS OAI)



**Figure 76. Association of VBS with the new tenant.**

At this point we had already created a tenant for the 21491 network with its associated VBS. The last step we had to complete was the user connection.

Before doing the connectivity test in the administrator account, optionally, we could map the different IMSI of the SIM cards to the MAC address of a specific mobile device. In Figure 77 we can see some examples. This is employed to associate one SIM card with a specific mobile device. It is not necessary associate a SIM card with a mobile device to connect it to the network.



**Figure 77. IMSI to MAC address mapping.**

The procedure followed to check the connectivity of the users is the same as above. To see the complete procedure see Annex 4.

- Activating the controller.
- Launching the processes of the EPC entities.
- Launching the process of the eNB.
- Switching on the mobile device and configuring the APN.

The following results must be obtained to consider that the implementation of the RAN slicing has been done satisfactorily.

- In the console of the eNB we should observe that appear the following messages: eNB establishing connection with the controller, UEs authentication, eNB association with an existing MME, eNB ready to users' connection. Once the user is connected, in the eNB would appear repetitive messages showing the RNTI assigned to the user and some parameters of QoS of the connection. In Figure 78, we observe that the indicated results are indeed obtained.



```
[PHY][I]cal 0: freq 350000000.000000, offset 44.000000, dlf 96000000.000000
[PHY][I]cal 1: freq 260000000.000000, offset 49.800000, dlf 120000000.000000
[PHY][I]cal 2: freq 230000000.000000, offset 51.000000, dlf 240000000.000000
[PHY][I]cal 3: freq 180000000.000000, offset 51.000000, dlf 60000000.000000
[PHY][I]cal 4: freq 83000000.000000, offset 57.000000, dlf 172000000.000000
[PHY][I]RX gain 0 100.000000 (52.800000) => 55.200000 (max 76.000000)
[PHY][I]Actual master clock: 30.720000MHz...
[PHY][I]RF board max packet size 3836, size for 100µs jitter 1536
[PHY][I]x_max_num_samps 1536
[PHY][I]RX channel 0
[PHY][I] Actual RX sample rate: 15.360000MSPs...
[PHY][I] Actual RX frequency: 2.540000GHz...
[PHY][I] Actual RX gain: 55.000000...
[PHY][I] Actual RX bandwidth: 20.000000MHz...
[PHY][I] Actual RX antenna: RX2...
[PHY][I]TX channel 0
[PHY][I] Actual TX sample rate: 15.360000MSPs...
[PHY][I] Actual TX frequency: 2.660000GHz...
[PHY][I] Actual TX gain: 89.750000...
[PHY][I] Actual TX bandwidth: 20.000000MHz...
[PHY][I] Actual TX antenna: TX/RX...
[PHY][I]Device timestamp: 0.000305...
[RRH] has loaded USRP B200 device.
setup_eb_buffers: frame_parns = 0x7f8bc3f1788
[PHY][I]initializing enb 0 cc_id 0 (enodeB 3GPP synch to ext device),
[HW][I][SCHED][enb] enb_thread_single started on CPU 0 TID 4597, sched_policy = SCHED_FIFO , priority = 99, CPU Affinity= CPU_0_CPU_1_CPU_2_CPU_3
Creating a thread
waiting for sync (enb_thread_single)
[PHY][I]thread to created id=4600[HW][I][SCHED][enb] enb_thread_prach started on CPU 1 TID 4601, sched_policy = SCHED_FIFO , priority = 99, CPU Affinity= CPU_0_CPU_1_CPU_2_CPU_3
[HW][I]enb_thread_synch started on CPU 0 TID 4602, sched_policy = SCHED_FIFO , priority = 99, CPU Affinity= CPU_0_CPU_1_CPU_2_CPU_3
waiting for sync (enb_thread_synch)
Setting enb buffer to all-RX
Sending sync to all threads
TYPE <CTRL-C> TO TERMINATE
Entering RTTI signals handler
got sync (enb_thread_single)
got sync (enb_thread_synch)
```

**Figure 78. eNB ready to connect users.**

```
# File Edit View Search Terminal Tabs Help
```

```
[usr@iogrom0~$mpowerview /opt/updat... * root[atcalission:/opt/openair-cn-new/$CSD] ... * root[atcalission:/opt/openair-cn-new/$CSD] ... * root[atcalission:/opt/openair-cn-new/$CSD] ... * root[atcalission:/opt/openair-cn-new/$CSD]
```

```
[RRC][I][rrc_conn_req_received] [enB] MeasIDList Is NULL  
[SCTP][I][sctp_send_data] Successfully sent 53 bytes on stream 1 for assoc_id 3  
[RRC][I][RRConnectionReconfigurationComplete] RRConnectionReconfiguration Encoded 1892 bits (137 bytes)  
[RRC][I][rrc_enb_generate_defaultRRConnectionReconfiguration] [enB 0] Frame 0, Logical channel DL-DCCH, Generate RRCConnectionReconfiguration (bytes 137, UE id c5d) [RRC][E]  
[PDCCP][I][FRAME_00000][enB][MOD 00][RNTI C5BD]Received RRC_DCCH_DATA_REQ from TASK_RRC_ENB; Instance no., rb_ld 1, mulTP conf, mode 1  
[RRC][I][Frame 00000][enB][MOD 00][RNTI C5BD][SRB AM 01] RLC_AM_DATA Req size 142 Bytes, ENB SDU 7 current_sdu_index=6 next_sdu_index=7 conf 0 mut 2 VEA 6 vts 6  
[RRC][I][rrc_tx_pdu] [enB] [MOD 00] [RNTI C5BD] [SRB 1] tx pdu on SRB 1 with size 7 from ue c5bd  
[RRC][I][rrc_enb_task] [enB][MOD 00][RNTI C5BD] Received on DCCH 1 RRC_DCCH_DATA_IND  
[RRC][I][rrc_enb_decode_dch] ----->>>[MSG] RRC Connection Reconfiguration Complete  
[RRC][I][rrc_enb_decode_dch] [FRAME 00000][enB][MOD 00][RNTI C5BD] UE State = RRC_RECONFIGURED (default DRB, xid 2)  
[PDCCP][N][FRAME 00000][enB][MOD 00][RNTI C5BD][SRB 02] Action ADD LCID 2 (SRB ld 2) configured with SN size 5 bits and RLC AM  
[PDCCP][N][frame 00000][enB][MOD 00][RNTI C5BD][DRB 01] Action ADD LCID 3 (DRB ld 2) configured with SN size 12 bits and RLC UM  
[RLC][I][FRAME 00000][enB][MOD 00][RNTI C5BD][SRB 2] rrc_rlc_add_rlc_SRB  
[RLC][I][FRAME 00000][enB][MOD 00][RNTI C5BD][SRB AM 02][CONFIGURE] max_retx_threshold 32 poll_pdu 8 pol_byt_e 16960 t_poll_retransmit 15 t_reordering 35 t_status_t  
[RLC][I][FRAME 00000][enB][MOD 00][RNTI C5BD][DRB 1] rrc_rlc_add_rlc_DRB  
[RRC][I][rrc_enb_process_RRCConnectionReconfigurationComplete] [enB 0] Frame 0 CC 0 : SRB3 is now active  
[RRC][I][rrc_enb_process_RRCConnectionReconfigurationComplete] [enB 0] Frame 0 : Logical Channel UL-DCCH, Received RRCConnectionReconfigurationComplete from ue re  
igning DRB 1/LCID 3  
[RRC][I][rrc_enb_process_RRCConnectionReconfigurationComplete] [enB 0] Frame 0 : Logical Channel UL-DCCH, Received RRCConnectionReconfigurationComplete from UE 0,  
DRB 1/LCID 3  
[MAC][I][CONTEXT][enB 0/0] configuring MAC/PHY for UE 0 (c5sd)  
[PHY][I]phy_config_dedicated.enb: physicalConfigDedicated=0x7fBB8A002110  
[PHY][I]Transmission Mode (phy_config_dedicated_enb) 1  
# enable-debug:Scheduled a 1 job for 1 msec  
[SIAP][I][siap_enb_initial_ctxt_resp] Initial_ctxt_resp.p: e_rb_idx 5, enb_addr 127.0.0.10, SIZE 4  
[PHY][I][rrc_rx_pdu] Successfully received 43 bytes on stream 1 for assoc_id 3  
[PHY][I][enB 0] Sent physicalConfigDedicated=0x7FBB8A002110 for UE 0  
# enable-debug:Releasing a 1 job  
[RRC][N][rrc_data_ind] [enB 0] Frame 993: received a DCCH 2 message on SRB 2 with Size 16 from UE c5bd  
[RRC][I][rrc_enb_task] [FRAME 00000][enB][MOD 00][RNTI C5BD] Received on DCCH 2 RRC_DCCH_DATA_IND  
[RRC][I][enB 0] Frame 994: DCCH->ULSCH, CLcid 6 Requesting 277 bytes from RLC (RRC message)  
[RRC][I][rrc_enb_decode_dch] ----->>>[MSG] RRC UL Information Transfer.  
[SCTP][I][sctp_send_data] Successfully sent 64 bytes on stream 1 for assoc_id 3  
# enable-debug:New trigger enabled, id=2, type=3  
# enable-debug:Scheduled a 1 job for 1 msec  
# enable-debug:Releasing a 1 job  
[PHY][I]UE 0 : rnti c5bd  
[MAC][I]UE 0 : rnti c5bd ; In synch, PHR 40 dB CQI 15  
[RRC][I][rrc_rx_tx] UE rnti c5bd failure timer 0/20000  
[SCTP][I][sctp_enb_flush_socket] Found data for descriptor 42  
[SCTP][I][sctp_enb_read_from_socket] Received notification for sd 42, type 32777  
[PHY][I]UE 0 : rnti c5bd  
[MAC][I]UE 0 : rnti c5bd ; In synch, PHR 40 dB CQI 15  
[RRC][I][rrc_rx_tx] UE rnti c5bd failure timer 0/20000  
[PHY][I]UE 0 : rnti c5bd  
[MAC][I]UE 0 : rnti c5bd ; In synch, PHR 40 dB CQI 15  
[RRC][I][rrc_rx_tx] UE rnti c5bd failure timer 0/20000  
[PHY][I]rx_rf: rfdevice timing drift of -1 samples (ts_off 62605760)  
[PHY][I]UE 0 : rnti c5bd  
[MAC][I]UE 0 : rnti c5bd ; In synch, PHR 40 dB CQI 15  
[RRC][I][rrc_rx_tx] UE rnti c5bd failure timer 0/20000  
[PHY][I]UE 0 : rnti c5bd  
[MAC][I]UE 0 : rnti c5bd ; In synch, PHR 40 dB CQI 15  
[RRC][I][rrc_rx_tx] UE rnti c5bd failure timer 0/20000
```

**Figure 79. UE authentication and connection.**

- In the console of the controller, besides the three messages to start the connection with the eNB, Hello messages from eNB are also shown. Once the user is connected, in the console the data of the corresponding user is viewed, including its IMSI and the PLMN Id of the network to which it belongs. In the following figure we observe that the indicated results are indeed obtained.

```

root@empowerlite: /opt/update/empower-runtime
usr@rogrcm@empowerlite: /opt/updat... x root@calislon: /opt/openair-cn-new/SC... x root@calislon: /opt/openair-cn-new/SC... x root@calislon: /opt/openair-cn-new/SC... x root@calislon: /opt/openair-cn-new/SC... x root@calislon: /opt/openair-cn-new/SC... x
INFO:core.pnfserver:Incoming connection from ('147.83.105.150', 50860)
INFO:vbsp.vbspconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 0
INFO:vbsp.vbspconnection:Sending VBS (eNB) cells conf request to VBS 00:00:00:00:0E:00 (3584)
head {
  vers: 1
  b_id: 3584
  seq: 1
  t_id: 0
}
se {
  mENB_cells {
    req {
      enb_info_types: 1
    }
  }
}

```

**Figure 80. VBS detected by the controller.**

```
rcm@empowerite: /opt/update/empower-runtime
user@rcm:cm@empowerite: /opt/updat... * root@calisson: /opt/openair-cn-new/SC... * root@calisson: /opt/openair-cn-new/SC... * root@calisson: /opt/openair-cn-new/SC... * root@calisson: /opt/openair-cn-new/SC...
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 12
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 13
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 14
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 15
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 16
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 17
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 18
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 19
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 20
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 21
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 22
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 23
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 24
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 25
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 26
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 27
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 28
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 29
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 30
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 31
```

Figure 81. Controller waiting for UE connection.

```
rcm@empowerite: /opt/update/empower-runtime
user@rcm:cm@empowerite: /opt/updat... * root@calisson: /opt/openairdebug/em... * root@calisson: /opt/openair-cn-new/SC... * root@calisson: /opt/openair-cn-new/SC... * root@calisson: /opt/openair-cn-new/SC...
INFO:vbss.vbssconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 159
head {
  vers: 1
  b_id: 3584
  seq: 160
  t_id: 1
}
te {
  mUes_id {
    repl {
      status: CREQS_SUCCESS
      active_ue_id {
        rnti: 55550
        lmsl: "214910000009916"
        plmn_id: "21491"
      }
    }
  }
}
```

Figure 82. UE (214910000009916) detected by the controller.

```
INFO:vbss.vbssconnection:Adding (EtherAddress('00:00:00:0E:00'), 23360) to tenant 21491
INFO:core.pnfserver:UE JOIN (EtherAddress('00:00:00:0E:00'), 23360) (21491)
INFO:core.module:Sending UEs RRC meas. config req to 00:00:00:0E:00 (id=4)
```

Figure 83. Join messages of UE to the tenant.

- In the console of the HSS the connected user is identified. In the following image we can appreciate that the HSS has correctly identified the user that has connected.

```
Received new update location request
Query: SELECT 'access_restriction', 'mmeidentity_idmmeidentity', 'mslcn', 'ue_ambr_ul', 'ue_ambr_dl', 'rau_tau_timer' FROM 'users' WHERE 'users'. 'lmsl'='214910000009916'
Query: INSERT INTO 'mmeidentity' ('mmehost', 'mmearea', 'mmeidentity_idmmeidentity') SELECT 'calisson.openair4g.eur', 'openair4g.eur' FROM 'mmeidentity' WHERE NOT EXISTS (SELECT * FROM 'mmeidentity' WHERE 'mmehost'='calisson.openair4g.eur' AND 'mmearea'='openair4g.eur') LIMIT 1;UPDATE 'users', 'mmeidentity' SET 'users'. 'mmeidentity_idmmeidentity'='mmeidentity'. 'idmmeidentity', 'users'. 'ms_ps_status'='NOT_PURGED' WHERE 'users'. 'lmsl'='214910000009916' AND 'mmeidentity'. 'mmehost'='calisson.openair4g.eur' AND 'mmeidentity'. 'mmearea'='openair4g.eur'
0 rows affected
0 rows affected
Query: SELECT * FROM 'pdn' WHERE 'pdn'. 'users_lmsl'='214910000009916' LIMIT 10;
```

Figure 84. HSS identifies the UE connected: 214910000009916.

- Finally, in the web interface, within the tenant created in the UEs tab, the parameters of the UE connected and associated to that tenant are shown. We can see how the user has been detected and based on its PLMN Id has been associated with the tenant.

5G-EmPOWER
Logged as: Foo (user) | Logout

Details: EMPOWER1 Components LVAPs LVNFs CPPs VBSes **UEs** WTPs

User Equipments

VBS	IMSI	PLMN ID	RNTI
00:00:00:00:0E:00	214910000009916	21491	50573

Figure 85. UE (214910000009916) associated to the tenant.

In addition to the connection of a single operator to a tenant we carried out one more study.

[illegible]

```

Received new update location request
Query: SELECT `access_restriction`,`mmedntity`,`idmmedntity`,`msidsn`,`ue_anbr_ul`,`ue_anbr_dl`,`rau_tau_ttnr` FROM `users` WHERE `users`.`lnst`='214910000009915'
Query: SELECT `mnehost`,`nmrealn` FROM `mmedntity` WHERE `mmedntity`.`idmmedntity`='1'
Query: INSERT INTO `mmedntity` (`mnehost`,`nmrealn`) SELECT `callsson`,`openair4G_eur` FROM `mmedntity` WHERE NOT EXISTS (SELECT * FROM `mmedntity` WHERE `mnehost`='callsson`,`openair4G_eur` AND `nmrealn`='openair4G_eur') LIMIT 1;UPDATE `users`,`mmedntity` SET `users`.`mmedntity`,`idmmedntity`,`mmedntity`.`idmmedntity`,`users`.`ms_ps_status`='NOT_PURGED' WHERE `users`.`lnst`='214910000009915' AND `mmedntity`.`mnehost`='callsson`,`openair4G_eur` AND `mmedntity`.`nmrealn`='openair4G_eur'
0 rows affected
0 rows affected
Query: SELECT * FROM `pdn` WHERE `pdn`.`users_lnst`='214910000009915' LIMIT 10;

Received new update location request
Query: SELECT `access_restriction`,`mmedntity`,`idmmedntity`,`msidsn`,`ue_anbr_ul`,`ue_anbr_dl`,`rau_tau_ttnr` FROM `users` WHERE `users`.`lnst`='214910000009915'
Query: SELECT `mnehost`,`nmrealn` FROM `mmedntity` WHERE `mmedntity`.`idmmedntity`='1'
Query: INSERT INTO `mmedntity` (`mnehost`,`nmrealn`) SELECT `callsson`,`openair4G_eur`,`openair4G_eur` FROM `mmedntity` WHERE NOT EXISTS (SELECT * FROM `mmedntity` WHERE `mnehost`='callsson`,`openair4G_eur` AND `nmrealn`='openair4G_eur') LIMIT 1;UPDATE `users`,`mmedntity` SET `users`.`mmedntity`,`idmmedntity`,`mmedntity`.`idmmedntity`,`users`.`ms_ps_status`='NOT_PURGED' WHERE `users`.`lnst`='214910000009915' AND `mmedntity`.`mnehost`='callsson`,`openair4G_eur` AND `mmedntity`.`nmrealn`='openair4G_eur'
0 rows affected
0 rows affected
Query: SELECT * FROM `pdn` WHERE `pdn`.`users_lnst`='214910000009915' LIMIT 10;

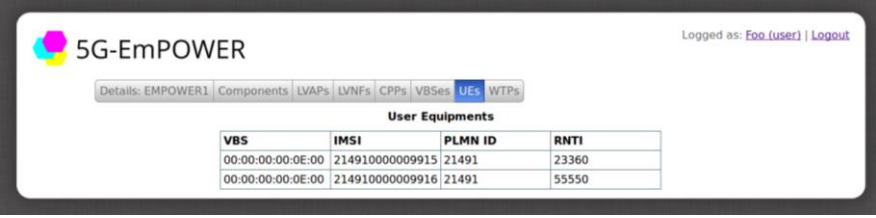
```

```

root@empowerlite:/opt/update/empower-runtime
root@empowerlite:/opt/update/empower-runtime# ./usrlog/crm@empowerlite:/opt/update/empower-runtime
root@calisson:/opt/openair-debug/em...
root@calisson:/opt/openair-cn-new/SC...
root@calisson:/opt/openair-cn-new/SC...
root@calisson:/opt/openair-cn-new/SC...
root@calisson:/opt/openair-cn-new/SC...
INFO:vbssp.vbsspconnection:Hello from 147.83.105.150 VBS 00:00:00:00:0E:00 seq 159
head {
  vers: 1
  b_id: 3584
  seq: 160
  t_id: 1
}
te {
  mUES_id {
    repl {
      status: CREQS_SUCCESS
      active_ue_id {
        rnti: 55550
        lmsi: "2149100000009916"
        plmn_id: "21491"
      }
      active_ue_id {
        rnti: 23360
        lmsi: "2149100000009915"
        plmn_id: "21491"
      }
    }
  }
}
INFO:vbssp.vbsspconnection:Adding (EtherAddress('00:00:00:00:0E:00'), 23360) to tenant 21491
INFO:core.mnppserver:UE JOIN (EtherAddress('00:00:00:00:0E:00'), 23360) (21491)
INFO:core.module:sending UES RRC MPRAS, con(*g, req, tx, 00:00:00:0E:00) (Id=4),

```





The screenshot shows the 5G-EmPOWER web interface. At the top, there is a navigation bar with tabs: Details: EMPOWER1, Components, LVAPs, LVNFs, CPPs, VBSes, **UEs**, and WTPs. The 'UEs' tab is selected. Below the navigation bar, there is a table titled 'User Equipments'. The table has four columns: VBS, IMSI, PLMN ID, and RNTI. There are two rows of data in the table.

VBS	IMSI	PLMN ID	RNTI
00:00:00:00:0E:00	214910000009915	21491	23360
00:00:00:00:0E:00	214910000009916	21491	55550

**Figure 88. Detection and association of 2 UEs, 214910000009915 and 214910000009916.**

In the same manner as in the previous case both terminals have been authenticated by the eNB without presenting any error, they have been identified by the HSS within its database, they have been detected by the controller and, therefore, associated with the tenant created. With these results we can conclude that more than one device can be connected to each tenant and that they do not affect each other. In addition, we tried to surfing the Internet and both devices were able to surf without problems and with a great quality of service.

With the completion of these studies we had already achieved the proposed goal for the project. First of all, we had been able to configure the different platforms so that our requirements were adjusted. Next, we were able to integrate both platforms to extend the platform based on SDN-NFV technologies with LTE technology. Finally, we had been able to create tenants and verify that users were able to associate with each other based on their PLMN Id.

Despite meeting all the proposed objectives we had time to make more progress in the study of the RAN Slicing so we decided to expand our goal and try to implement multiple operator in our network, that is, we wanted to build a second operator, or what is the same a new EPC, that together with the first EPC were able to connect at the same time to the same eNB.

#### **4.6. Implementation of Multiple Operator Core Network (MOCN)**

The objective of this second part of the project was to create a new EPC with a new configuration and with its respective users, which was able to connect, first, to the existing eNB, and then together with the EPC of the 21491 network could connect to the eNB simultaneously.

To develop this part of the project, previously, we had to verify if the implementation of the multiple operators was possible with one of the existing configurations files provided by the eNB. Fortunately, we found a configuration (enb.band7.tm1.usrpb210.epc.conf) in which it was allowed to implement the MOCN option and configure up to six possible PLMN Ids with which to connect. When we verify that the developing the MOCN was possible, we set out to create the new network operator.

##### **4.6.1. Creation of a new operator**

The creation of a new operator entails the creation of new elements:

- Network parameters
- Operator users
- EPC
- Database

The following sections describe how the process to create all these elements was performed.

### General network parameters

As first step, the definition of the network parameters was done. As explained in the definition of parameters for the network 21491, to define the PLMN Id of this network we only needed to define the values of the MCC and the MNC. In the case of the MCC, the same value would be maintained as in the other EPC, 214 corresponding to the country code. While the MNC had to be chosen taking into account the MNCs not available in the country (see MNC table). In this case we decided that the MNC of our network would be 12. Therefore, the PLMN Id of the new network was 21412.

PLMN Id	
MCC	MNC
214	12

**Figure 89. PLMN Id definition.**

Changing the operator involves modifying the MNC parameter and, therefore, changing the parameters of the MME, GUMMEI and TAI List. Next, the values of both are presented.

GUMMEI		
PLMN Id	MMEGI	MMEC
21412	4	1

**Figure 90. GUMMEI definition.**

TAI List	
PLMN Id	TAC
21412	1

**Figure 91. TAI List definition.**

The created PLMN Id had to be entered in the `mcc_mnc_list` file, located in the `openair-cn-new` software, for the appropriate operation of the new network.

### Creation of a new user

The creation of a new user entails programming a new SIM card with the main operator parameters that are what we defined previously with the users of the 21491 network. In this case, the parameters that we defined for the new operator are the following:

	#1
MCC	214
MNC	12
TAI	1
IMSI	214120000009919
Key	EE3701E7BDF200C42207A 073891F8202
OPc	1006020f0a478bf6b699f15c062e42b3
ADM1	91969642
CC	34
ICCID	8934121000000099195

Table 12. Configuration parameter of new SIM card.

The PySIM software was used for the creation of the new user, since it allowed us to configure the new SIM card by introducing the previous defined parameters, using Linux commands.

#### 4.6.2. Network architecture

The creation of a new EPC led to the redistribution of PCs. From this moment we would no longer work with the configuration of 1 PC since the processes of 2 EPCs could not be launched at the same time and neither could the eNB be implemented in the same host as any of the EPCs. In this case we need 3 PCs to carry out the implementation of the MOCN. In Figure 92, the architecture is presented detailing the IP addresses of the hosts and the interfaces that connect the different entities of the network.

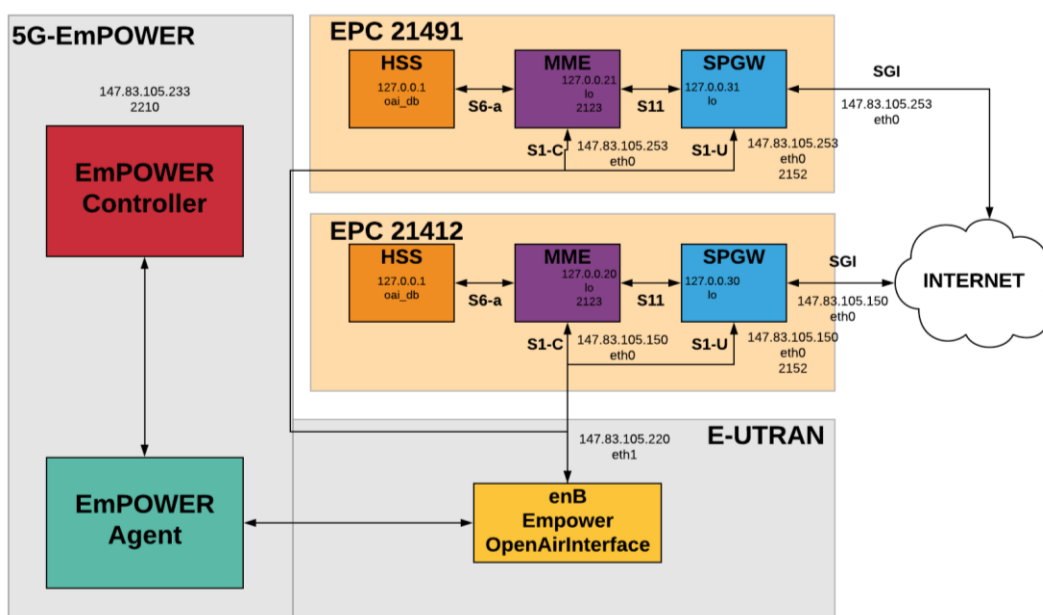


Figure 92. Overall architecture with MOCN.

In Table 13 the IP addresses and interfaces allocation for the different entities are presented.

EPC 21491	eNB	MME	SPGW	External network
eNB		147.83.105.220 (eth1)	147.83.105.220 (eth1)	
MME	147.83.105.253 (eth0)		127.0.10.21 (lo) 2123	
SPGW	147.83.105.253 (eth0)	127.0.0.31 (lo)		147.83.105.253 (eth0)
EPC 21412	eNB	MME	SPGW	External network
eNB		147.83.105.220 (eth1)	147.83.105.220 (eth1)	
MME	147.83.105.150 (eth0)		127.0.10.20 (lo) 2123	
SPGW	147.83.105.150 (eth0)	127.0.0.30 (lo)		147.83.105.150 (eth0)

**Table 13. Allocation of IP addresses for 21491 and 21412 networks.**

With the new distribution of PCs, some changes in the configuration files of the eNB and EPC1 (network 21491), which is now located in the host 147.83.105.220, were needed in relation to IP addresses and interfaces. These changes are shown below (complete configuration files are in Annex 1).

In the MME, configuration file **mme.conf**, the following IP addresses and interfaces had to be modified.

```

151 NETWORK_INTERFACES :
152 {
153     # MME binded interface for S1-C or S1-MME communication (S1AP), can be
154     # ethernet interface, virtual ethernet interface, we don't advise wireless
155     # interfaces
156     MME_INTERFACE_NAME_FOR_S1_MME = "eth0"; # YOUR
157     NETWORK CONFIG HERE
158     MME_IPV4_ADDRESS_FOR_S1_MME = "147.83.105.253/32"; #
159     YOUR NETWORK CONFIG HERE
160
161     # MME binded interface for S11 communication (GTPV2-C)
162     MME_INTERFACE_NAME_FOR_S11_MME = "lo"; # YOUR NETWORK
163     CONFIG HERE
164     MME_IPV4_ADDRESS_FOR_S11_MME = "127.0.0.21/8"; # YOUR NETWORK
165     CONFIG HERE
166     MME_PORT_FOR_S11_MME = 2123; # YOUR NETWORK
167     CONFIG HERE
168 };

```

```

204 S-GW_LIST_SELECTION = (
205     {ID="tac-lb01.tac-hb00.tac.epc.mnc091.mcc214.3gppnetwork.org" ;
206     SGW_IPV4_ADDRESS_FOR_S11="127.0.0.31/8";}
207 );

```

**Figure 93. Configuration of mme.conf for 21491 network.**

In the file **mme\_fd.conf** the identity of the MME had to be varied. In this case the new host was called nano, so the following change was implemented (Figure 94). At the same

time, the identity of the MME in the mmeidentity table of the database had to be varied so that the HSS could detect the MME.

```
1 # ----- Local -----
2
3 # Uncomment if the framework cannot resolv it.
4 Identity = "nano.openair4G.eur";
5 Realm = "openair4G.eur";
```

**Figure 94. Configuration of mme\_fd.conf for 21491 network.**

Finally, in the SPGW entity, IP addresses and interfaces had to be changed in **spgw.conf** as shown in Figure 95.

```
23 NETWORK_INTERFACES :
24 {
25     # S-GW binded interface for S11 communication (GTPV2-C), if none selected
26     # the ITTI message interface is used
27     SGW_INTERFACE_NAME_FOR_S11 = "lo"; #
28     STRING, interface name, YOUR NETWORK CONFIG HERE
29     SGW_IPV4_ADDRESS_FOR_S11 = "127.0.0.31/8" #
30     STRING, CIDR, YOUR NETWORK CONFIG HERE
31
32     # S-GW binded interface for S1-U communication (GTPV1-U) can be ethernet
33     # interface, virtual ethernet interface, we don't advise wireless interfaces
34     SGW_INTERFACE_NAME_FOR_S1U_S12_S4_UP = "eth0"; #
35     STRING, interface name, YOUR NETWORK CONFIG HERE, USE "lo" if S-GW run on
36     # eNB host
37     SGW_IPV4_ADDRESS_FOR_S1U_S12_S4_UP = "147.83.105.253/32"; #
38     STRING, CIDR, YOUR NETWORK CONFIG HERE
39     SGW_IPV4_PORT_FOR_S1U_S12_S4_UP = 2152; #
40     INTEGER, port number, PREFER NOT CHANGE UNLESS YOU KNOW WHAT YOU ARE DOING
41
42     # S-GW binded interface for S5 or S8 communication, not implemented, so
43     # leave it to none
44     SGW_INTERFACE_NAME_FOR_S5_S8_UP = "none"; #
45     STRING, interface name, DO NOT CHANGE (NOT IMPLEMENTED YET)
46     SGW_IPV4_ADDRESS_FOR_S5_S8_UP = "0.0.0.0/24"; #
47     STRING, CIDR, DO NOT CHANGE (NOT IMPLEMENTED YET)
48 };
```

**Figure 95. Configuration of spgw.conf for 21491 network.**

In the eNB the following changes were done for the configuration file that allows the eNB to connect with EPC and with 2 EPC respectively.

For the configuration file that allows to connect the eNB with a single EPC (**enb.band7.tm1.usrpb210.conf**) the MNC, the IP address of the MME and the IP addresses and interfaces that connect the eNB with the MME and the SPGW were changed.

```
15 // Tracking area code, 0x0000 and 0xfffe are reserved values
16 tracking_area_code = "1";
17
18 mobile_country_code = "214";
19 mobile_network_code = "12";

140 mme_ip_address = ( { ipv4 = "147.83.105.150";
141                       ipv6 = "192:168:30::17";
142                       active = "yes";
143                       preference = "ipv4";
144                       }

147 NETWORK_INTERFACES :
148 {
149
150     ENB_INTERFACE_NAME_FOR_S1_MME = "eth1";
151     ENB_IPV4_ADDRESS_FOR_S1_MME = "147.83.105.220/32"; #
152     127.0.0.10/8 for same pc config
153     ENB_INTERFACE_NAME_FOR_S1U = "eth1";
154     ENB_IPV4_ADDRESS_FOR_S1U = "147.83.105.220/32";
155     ENB_PORT_FOR_S1U = 2152; # Spec 2152
156 };
```

**Figure 96. Changes in eNB configuration file to connect with new EPC.**

For the configuration file that allows to connect the eNB with the 2 EPCs (**enb.band7.tm1.usrpb210.epc.conf**) the MNC, the IP addresses of the EPC1 and EPC2 MMEs and the IP addresses and interfaces that connect the eNB with the MMEs and the SPGWs were modified.

```

15      // Tracking area code, 0x0000 and 0xfffe are reserved values
16      tracking_area_code = "1";
17
18      mobile_country_code = "214";
19      mobile_network_code = "12";
20
21      ///// Max of 6 PLMN IDS can be broadcasted
22      multiple_OCN = "True";
23      plmn_ids = ( {
24          mobile_country_code = "214";
25          mobile_network_code = "12";
26      },
27      {
28          mobile_country_code = "214";
29          mobile_network_code = "91";
30      }
31      );

151      mme_ip_address = ( {ipv4 = "147.83.105.150";
152                          ipv6="192:168:30::17";
153                          active="yes";
154                          preference="ipv4";
155                      },
156                      {ipv4 = "147.83.105.253";
157                          ipv6="192:168:30::18";
158                          active="yes";
159                          preference="ipv4";
160                      }
161      );
162
163      NETWORK_INTERFACES :
164      {
165          ENB_INTERFACE_NAME_FOR_S1_MME = "eth1";
166          ENB_IPV4_ADDRESS_FOR_S1_MME = "147.83.105.220/32";
167
168          ENB_INTERFACE_NAME_FOR_S1U = "eth1";
169          ENB_IPV4_ADDRESS_FOR_S1U = "147.83.105.220/32";
170          ENB_PORT_FOR_S1U = 2152; # Spec 2152
171      };

```

**Figure 97. Changes in eNB configuration file to connect with 2 EPCs.**

#### 4.6.3. Configuration of the new EPC

Once the SIM of the new operator was programmed and the configurations of the eNB and the EPC1 varied, the next step consisted of the personalization of the new EPC according to the new network parameters and the IP addresses previously shown. We created a copy of the openair-cn software of EPC1 to create EPC2 and customized it. Following the same procedure that has been discussed in previous sections we configured the following files for the different entities.

#### HSS

The creation of a new operator involves the implementation of a new database. We called it oai2\_db and created the following tables:

- users
- pdn
- mmeidentity

In the following figures, extracted from the phpmyadmin interface, you can see the tables defined above. To know the complete procedure to add a new user see Annex 3.

imsi	msisdn	imei	imei_sv	ms_ps_status	rau_tau_timer	ue_ambr_ul	ue_ambr_dl	access_restriction	mme_cap
imsi is the main reference key.	The basic MSISDN of the UE (Presence of MSISDN is optional).	International Mobile Equipment Identity	International Mobile Equipment Identity Software Version Number	Indicates that ESM and EMM status are purged from MME		The Maximum Aggregated uplink MBRs to be shared across all Non-GBR bearers according to the subscription of the user.	The Maximum Aggregated downlink MBRs to be shared across all Non-GBR bearers according to the subscription of the user.	Indicates the access restriction subscription information. 3GPP TS-29272 #7.3.31	Indicates the capabilities of the MME with respect to core functionality e.g. regional access restrictions.
214120000009919	34600000005	NULL	NULL	NOT_PURGED	120	50000000	100000000		47 0000000000

mmeidentity_id	mmeidentity_key	RFSP-Index	urpp_mme_sqn	rand	OPc
	UE security key	An index to specific RFM configuration in the E-UTRAN. Possible values from 1 to 256	UE Reachability Request Parameter indicating that UE activity notification from MME has been requested by the HSS		Can be computed by HSS
1	ee3701e7bdf200c42207a073891f8202	1	0	0000000000000000006816	2baa4e55e388e96bdee2031e121c257d
					a469f2da4cadeaf3d51d387b1c33dbd6

Figure 98. Configuration of users table.

id	apn	pdn_type	pdn_ipv4	pdn_ipv6	aggregate_ambr_ul	aggregate_ambr_dl	pgw_id	users_imsi	qci	priority_level	pre_emp_cap	pre_emp_vul	LIPA-Permissions
5	oai2.ipv4	IPv4	0.0.0.0	0:0:0:0:0:0	50000000	100000000	3	214120000009919	7	7	DISABLED	ENABLED	LIPA-only

Figure 99. Configuration of pdn table.

idmmeidentity	mmehost	mmerealm	UE-Reachability
1	calisson.openair4G.eur	openair4G.eur	Indicates whether the MME supports UE Reachability Notification
			0

Figure 100. Configuration of mmeidentity table.

According to the previous tables, the configuration of this new user entails the creation of a new access point oai2.ipv4 so that no confusion is created with the APN of EPC1. Another aspect to take into account is the name of the MME identity that is calisson. It is necessary that the identities of each MME be unique, which confirms the need to separate the EPCs in different host. The configuration of the user and pdn tables has been made from the parameters of the new user and the values of the parameters defined by the OAI [30].

In the **hss.conf** file, the OPc parameter was modified according to the one defined in the operator, as shown in the following figure.

```

21 HSS :
22 {
23   ## MySQL mandatory options
24   MYSQL_server = "127.0.0.1"; # HSS S6a bind address
25   MYSQL_user = "root"; # Database server login
26   MYSQL_pass = "mysqlpa33w0rd"; # Database server password
27   MYSQL_db = "oai2_db"; # Your database name
28
29   ## HSS options
30   #OPERATOR_key = "29C7D674257AF3B0C6C9F7879663F5FB"; # OP Ki from sysmocom USIM_2
31   OPERATOR_key = "1006020f0a478bf6b699f15c062e42b3"; # OP key matching your database
32   #OPERATOR_key = "11111111111111111111111111111111"; # OP key matching your database
33
34   RANDOM = "true"; # True random or only pseudo
35   random (for subscriber vector generation)
36
37   ## Freediameter options
38   FD_conf = "/usr/local/etc/oai/freeDiameter/hss2_fd.conf";
39 };
```

Figure 101. Configuration of hss.conf for the 21412 network.

## MME

In the file **mme.conf** we made the following changes.



```

73 # ----- MME served GUMMEIs
74 # MME code DEFAULT size = 8 bits
75 # MME GROUP ID size = 16 bits
76 # MME_CODE = [ 1, 30, 31, 32, 33, 34, 35, 36, 56, 29, 8 ]; #grcm
77 # MME_GID = [ 32768, 4, 5, 30, 8, 9, 50021 ]; #grcm
78 GUMMEI_LIST = (
79     {MCC="214" ; MNC="12"; MME_GID="4" ; MME_CODE="1"; }
80 );
81
82 # ----- MME served TAIs
83 # TA (mcc.mnc:tracking area code) DEFAULT = 208.34:1
84 # max values = 999.999:65535
85 # maximum of 16 TAIs, comma separated
86 # !!! Actually use only one PLMN
87 TAI_LIST = (
88     {MCC="214" ; MNC="12"; TAC = "1"; }
89 );
90
91
92 NETWORK_INTERFACES :
93 {
94     # MME binded interface for S1-C or S1-MME communication (S1AP), can be
95     # ethernet interface, virtual ethernet interface, we don't advise wireless
96     # interfaces
97     MME_INTERFACE_NAME_FOR_S1_MME = "eth0"; # YOUR
98     NETWORK_CONFIG_HERE
99     MME_IPV4_ADDRESS_FOR_S1_MME = "147.83.105.150"; # YOUR
100     NETWORK_CONFIG_HERE
101
102     # MME binded interface for S11 communication (GTPV2-C)
103     MME_INTERFACE_NAME_FOR_S11_MME = "lo"; # YOUR NETWORK
104     CONFIG_HERE
105     MME_IPV4_ADDRESS_FOR_S11_MME = "127.0.0.20/8"; # YOUR NETWORK
106     CONFIG_HERE
107     MME_PORT_FOR_S11_MME = 2123; # YOUR NETWORK
108     CONFIG_HERE
109 }
110
111 S-GW_LIST_SELECTION = (
112     {ID="tac-lb01.tac-hb00.tac.epc.mnc012.mcc214.3gppnetwork.org" ;
113     SGW_IPV4_ADDRESS_FOR_S11="127.0.0.30/8";}
114 );

```

**Figure 102. Configuration of mme.conf for the 21412 network.**

According to the network parameters, we had to change the values of the MNC in the GUMMEI and in the TAI List. We also varied the IP addresses and interfaces that connected the MME with the eNB and the SPGW. Finally, we made a modification in the IP address and interface that connects the SPGW with the MME.

On the other hand, in the **mme\_fd.conf** file was verified that the identity of the MME was calisson.

```

1 # ----- Local -----
2
3 # Uncomment if the framework cannot resolve it.
4 Identity = "calisson.openair4G.eur";
5 Realm = "openair4G.eur";

```

**Figure 103. Configuration of mme\_fd.conf for the 21412 network.**

## SPGW

Finally, in the **spgw.conf** file were varied the IP addresses and interfaces which connect the SPGW to the other network entities, as shown Figure 104.



```

23 NETWORK_INTERFACES :
24 {
25     # S-GW binded interface for S11 communication (GTPV2-C), if none selected
26     # the ITTI message interface is used
27     SGW_INTERFACE_NAME_FOR_S11 = "lo"; #
28     STRING, interface name, YOUR NETWORK CONFIG HERE
29     SGW IPV4 ADDRESS FOR S11 = "127.0.0.30/8" #
30     STRING, CIDR, YOUR NETWORK CONFIG HERE
31
32     # S-GW binded interface for S1-U communication (GTPV1-U) can be ethernet
33     # interface, virtual ethernet interface, we don't advise wireless interfaces
34     SGW_INTERFACE_NAME_FOR_S1U_S12_S4_UP = "eth0"; #
35     STRING, interface name, YOUR NETWORK CONFIG HERE, USE "lo" if S-GW run on
36     # eNB host
37     SGW IPV4 ADDRESS FOR S1U_S12_S4_UP = "147.83.105.150/32"; #
38     STRING, CIDR, YOUR NETWORK CONFIG HERE
39     SGW IPV4 PORT FOR S1U_S12_S4_UP = 2152; #
40     INTEGER, port number, PREFER NOT CHANGE UNLESS YOU KNOW WHAT YOU ARE DOING
41
42     # S-GW binded interface for S5 or S8 communication, not implemented, so
43     # leave it to none
44     SGW_INTERFACE_NAME_FOR_S5_S8_UP = "none"; #
45     STRING, interface name, DO NOT CHANGE (NOT IMPLEMENTED YET)
46     SGW IPV4 ADDRESS FOR S5_S8_UP = "0.0.0.0/24"; #
47     STRING, CIDR, DO NOT CHANGE (NOT IMPLEMENTED YET)
48 }
49
50 PGW_IPV4_ADDRESS_FOR_SGI = "147.83.105.150/32"; #

```

Figure 104. Configuration of spgw.conf for 21412 network.

#### 4.6.4. Creation of a new tenant

Before carrying out the connectivity tests to verify if the network had been properly configured, we had to create a new tenant for the 21412 network that was able to give resources to the users of its network. The procedure for the creation of a new tenant was the same that the explained for the tenant in network 21491, but taking into account the parameters of this network. We created the new tenant and associated it with the existing VBS. In Figure 105, the created tenant and the association of the VBS to it are shown.

The screenshot shows the 5G-EmPOWER interface. The top part displays 'Active Tenants' with a table listing three tenants. The bottom part shows 'Virtual Basestation Points' with a table listing one VBS associated with the tenant.

UUID	Network Name	Devs	Owner	PLMN ID	BSSID Type
60105977-16b5-441a-a456-0f1ee18e1e76	EMPOWER3	00:00:00:00:0E:00	foo	21412	unique
0169c00f-16c6-4af1-81ab-4992d0a71527	EMPOWER2	00:00:00:00:0E:00	foo	21491	unique
7056f59f-cd50-4604-9744-231de5044bd2	EMPOWER1	00:00:00:00:0E:00	foo	21491	unique

Virtual Basestation Points	
VBS OAI (00:00:00:00:0E:00)	at 147.83.105.220,57918, last seen: 16

Figure 105. Creation of a new tenant and VBS association.

#### 4.6.5. Connectivity tests of the new operator

The next step after having configured the entire network was checking if the new operator worked correctly. For this, as we have explained previously for other configurations of the network, we launched the process of the controller so that it could be initialized. Next, we run the processes of the EPC entities (HSS, MME and SPGW). Finally, we ran the eNB. The results obtained are shown in Figures 106 - 112.

The new operator had been configured correctly. In the terminal of the controller we were able to see how connection established connection with the eNB as well as observe in the web interface how the VBS was shown in operation (green flag). On the other hand, in the eNB we observed that it had been initialized correctly, had made the connection with the EPC and was prepared for the connection of users.

```
INFO:core.pnfpserver:incoming connection from ('147.83.105.150', 50800)
INFO:vbsp.vbspconnection:Hello from 147.83.105.150 VBS 00:00:00:00:00:00 seq 0
INFO:vbsp.vbspconnection:Sending VBS (eNB) cells conf request to VBS 00:00:00:00:00:00 (3584)
head {
  vers: 1
  b_id: 3584
  seq: 1
  t_id: 0
}
se {
  mENB_cells {
    req {
      enb_info_types: 1
    }
  }
}
```

Figure 106. Controller establish connection with eNB.

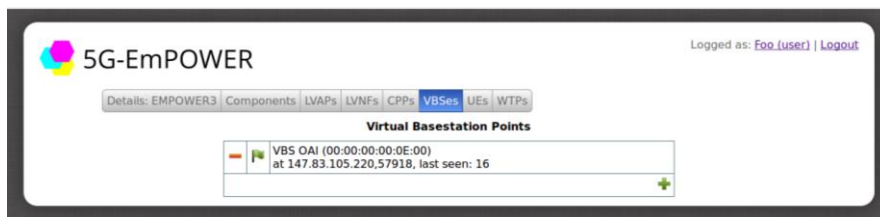


Figure 107. VBS is active.

```
[PHY]RX Channel 0
[PHY] Actual RX sample rate: 15.360000Mps...
[PHY] Actual RX frequency: 2.540000GHz...
[PHY] Actual RX gain: 90.750000...
[PHY] Actual RX bandwidth: 20.000000M...
[PHY] Actual RX antenna: RX2...
[PHY]TX Channel 0
[PHY] Actual TX sample rate: 15.360000Mps...
[PHY] Actual TX frequency: 2.540000GHz...
[PHY] Actual TX gain: 90.750000...
[PHY] Actual TX bandwidth: 20.000000M...
[PHY] Actual TX antenna: TX/RX...
[PHY] Device timestamp: 0.000000...
[RRH] has loaded USRP B200 device
Setup eNB buffers: frame_parms = 0x7f3d107d7000
[PHY] Initializing eNB 0 cc_id 0 (eNB0B-3GPP_synth to ext_device)
[HW] [SCHED] eNB thread single started on CPU 0 TID 3987, sched_policy = SCHED_FIFO, priority = 99, CPU Affinity= CPU_0 CPU_1 CPU_2 CPU_3
Creating thread
Waiting for sync (eNB_thread_single)
[PHY] Thread created id=3988[HW] [SCHED] eNB_thread_prach started on CPU 1 TID 3991, sched_policy = SCHED_FIFO, priority = 99, CPU Affinity= CPU_0 CPU_1 CPU_2 CPU_3
[HW] [SCHED] eNB_thread_synth started on CPU 0 TID 3992, sched_policy = SCHED_FIFO, priority = 99, CPU Affinity= CPU_0 CPU_1 CPU_2 CPU_3
Waiting for sync (eNB_thread_synth)
Setting eNB buffer to all-RX
Sending sync to all threads
TYPE <CTRL-C> TO TERMINATE
Entering ITTI signals handler
Not sync (eNB_thread_single)
Not sync (eNB_thread_synth)
```

Figure 108. eNB is ready for the UE connection.

The following step was to verify that we had correctly configured the new SIM card and was able to connect to the network. To test we use Dongle as mobile device with the new programmed SIM card (214120000009919). We entered the Telenor application and once the devices were initialized we saw the following behavior in the controller, the webinterface, the eNB and the HSS.

```
Received new update location request
Query: SELECT access_restriction, mmeidentity_idmmeidentity, nstsdn, ue_enbr_ul, ue_enbr_dl, rau_tau_timer FROM 'users' WHERE 'users'. 'nsi'='214120000009919'
Query: SELECT mmehost, mmearea FROM 'mmeidentity' WHERE 'mmeidentity'. 'idmmeidentity'='1'
Query: INSERT INTO 'mmeidentity' ('mmehost', 'mmearea') SELECT 'calisson.openair4g.eur', 'openair4g.eur' FROM 'mmeidentity' WHERE NOT EXISTS (SELECT * FROM 'mmeidentity' WHERE 'mmehost'='calisson.openair4g.eur' AND 'mmearea'='openair4g.eur') LIMIT 1;UPDATE 'users', 'mmeidentity' SET 'users'. 'mmeidentity_idmmeidentity'='mmeidentity'. 'idmmeidentity', 'users'. 'ns_ps_status'='NOT_PURGED' WHERE 'users'. 'nsi'='214120000009919' AND 'mmeidentity'. 'mmehost'='calisson.openair4g.eur' AND 'mmeidentity'. 'mmearea'='openair4g.eur'
0 rows affected
0 rows affected
Query: SELECT * FROM 'pdn' WHERE 'pdn'. 'users nsi'='214120000009919' LIMIT 10;
```

Figure 109. HSS has identified the UE connected.



In the terminal of the HSS we see how the entity was able to connect with the new database and was able to detect the user that was connecting to the network. In the eNB is observed the detection of a new UEs, the UEs authentication messages and association processes with the MME. Also, the connection messages in which an RNTI and QoS are assigned to the user are presented. Finally, in the terminal of the controller and in the web interface we can see how the user is recognised and associated with the new tenant in accordance with its PLMN Id. In conclusion, the user was able to connect to the network satisfactorily.

The last step we had to develop to implement the MOCN was the simultaneous connection of the 2 EPCs to the same eNB. For this, as we have indicated at the beginning of this section, it was necessary to have an eNB capable of connecting to both EPCs. Since we already had all the entities configured to connect with each other we just had to launch the controller, then the processes of the two EPCs and finally the eNB with the new configuration. Next, we present the results obtained.

The eNB was able to create peers with the 2 EPCs and to communicate with them. However, in the association of both EPCs process with the MMEs, one of the networks (21412) was able to associate to the eNB but the other seems to be discarded and the context that comes from this is rejected, as shown Figures 113 and 114.

```

File Edit View Bookmarks Settings Help
[SCPT][1][[sctp_get_sockinfo] state ..... 2
[SCPT][2][[sctp_get_sockinfo] cwnd ..... 4380
[SCPT][1][[sctp_get_sockinfo] srtt ..... 0
[SCPT][1][[sctp_get_sockinfo] rto ..... 3000
[SCPT][1][[sctp_get_sockinfo] mtu ..... 1500
[SCPT][1][[sctp_get_sockinfo] .....
[SCPT][1][[sctp_ehb_read from socket] Com up notified for sd 42, assigned assoc_id 23
[SIAP][1][[slap_ehb_generate sl setup request] 3584 -> 00e000
[SCPT][1][[sctp_send data] Successfully sent 59 bytes on stream 0 for assoc_id 23
[SCPT][1][[sctp_ehb_flush sockets] Found data for descriptor 42
[SCPT][1][[sctp_ehb_read from socket] Received notification for sd 42, type 32777
[SCPT][1][[sctp_ehb_flush sockets] Found data for descriptor 43
[SCPT][1][[sctp_ehb_read from socket] Received notification for sd 43, type 32769
[SCPT][1][[sctp_ehb_read from socket] Client association changed: 0
[SCPT][1][[sctp_get_peeraddresses] .....
[SCPT][1][[sctp_get_peeraddresses] Peer addresses:
[SCPT][1][[sctp_get_peeraddresses] - [sct]83.105.150]
[SCPT][1][[sctp_get_peeraddresses] .....
[SCPT][1][[sctp_get_sockinfo] .....
[SCPT][1][[sctp_get_sockinfo] Sctp status:
[SCPT][1][[sctp_get_sockinfo] assoc id ..... 24
[SCPT][1][[sctp_get_sockinfo] state ..... 4
[SCPT][1][[sctp_get_sockinfo] antrms ..... 2
[SCPT][1][[sctp_get_sockinfo] outstrms ..... 2
[SCPT][1][[sctp_get_sockinfo] fragmentation : 1452
[SCPT][1][[sctp_get_sockinfo] pending data ..... 0
[SCPT][1][[sctp_get_sockinfo] unack data ..... 0
[SCPT][1][[sctp_get_sockinfo] rwnd ..... 106496
[SCPT][1][[sctp_get_sockinfo] peer info .....
[SCPT][1][[sctp_get_sockinfo] state ..... 2
[SCPT][1][[sctp_get_sockinfo] cwnd ..... 4380
[SCPT][1][[sctp_get_sockinfo] srft ..... 0
[SCPT][1][[sctp_get_sockinfo] rto ..... 3000
[SCPT][1][[sctp_get_sockinfo] mtu ..... 1500
[SCPT][1][[sctp_get_sockinfo] .....
[SCPT][1][[sctp_ehb_read from socket] Com up notified for sd 43, assigned assoc_id 24
[SIAP][1][[slap_ehb_generate sl setup request] 3584 -> 00e000
[SCPT][1][[sctp_send data] Successfully sent 59 bytes on stream 0 for assoc_id 24
[SCPT][1][[sctp_ehb_flush sockets] Found data for descriptor 42
[SCPT][1][[sctp_ehb_read from socket] [sct]83.105.150] Msg of length 15 received from port 36412, on stream 0, PPID 18
[SCPT][1][[sctp_ehb_flush sockets] Found data for descriptor 43
[SCPT][1][[sctp_ehb_read from socket] Received notification for sd 43, type 32777
[SCPT][1][[sctp_ehb_read from socket] [sct]83.105.150] Decoding message Slap SisetupFailureIEs (/opt/openairdebug/empower-openairinterface/cmake_targets/lte_build_oai/build/ChokeFiles/R10.5/slap_decoder.cc:5328)
[SIAP][1][[slap_ehb_handle sl setup failure] [SCPT 23] Received S1 setup response from non existing RWE context
[SCPT][1][[sctp_ehb_flush sockets] Found data for descriptor 43
[SCPT][1][[sctp_ehb_read from socket] [sct]83.105.150] Msg of length 27 received from port 36412, on stream 0, PPID 18
[SIAP][1][[slap_decode slap sisetupresponses] Decoding message Slap SisetupResponseIEs (/opt/openairdebug/empower-openairinterface/cmake_targets/lte_build_oai/build/ChokeFiles/R10.5/slap_decoder.cc:3535)
[SIAP][1][[slap_ehb_handle sl setup response] ServedMMEIEs.list.count 1
[SIAP][1][[slap_ehb_handle sl setup response] ServedPLMNs.list.count 1
[SIAP][1][[slap_ehb_handle sl setup response] .....
[PHY][1][[initializing ehb CC id 0 : (elnodeB_3GPP_synch to eut_device)

```

**Figure 113. MME from 21491 is considered none existing.**

[illegible]

**Figure 114. Error in MME of 21491 network: No common PLMN Id with eNB.**

From the study of the different configuration files we found that in the eNB, the EPC that was associated with the eNB was the first one that was defined in the file while the other was discarded. We modified the file of the eNB to see what the behavior of the eNB was if we varied those parameters and changed the order of the values of the MNC. The result was the same than the previous one, the eNB was connected to both EPCs but was only able to associate with the MME of the first defined network, in this case 21491.

After making several changes to the configurations, studying the software codes of the eNB (empower-openairinterface) and doing research to find answers from other developers, we were not able to find what change had to be made in order to implement the MOCN.

Despite this we had been able to create a new operator, generate a new client for this operator, configure the different entities of the network in different hosts and finally connect the two EPCs separately with the eNB.

## 5. Conclusions and future development

In this master thesis, RAN Slicing has been studied and implemented on a platform based on SDN and NFV technologies. This platform has been constituted by two software: OpenAirInterface (OAI), which emulates the LTE protocol stack of 3GPP, and 5G-EmPOWER, which provides the system with a Controller, responsible for managing the different tenants, and an Agent that acts as an interface between the controller and the LTE network. Before integrating the software, both have been studied, configured and implemented separately.

The implementation of RAN Slicing allows, as we have seen, to create a more flexible and manageable architecture from a high-level programmable web interface, which does not require barely hardware changes, which is a very attractive approach since the costs are reduced and allows a better management of resources due to the performance of controller.

The first part of the project has been devoted to the study of the LTE network implementation using OAI, in which it has managed to configure and implement a 4th generation network according to the requirements established in the project. Thanks to applications such as RFBENCHMARK or Qualipoc it has been possible to verify the availability and reach of the network. On the other hand, with the PySim and MyQSL softwares, it has been possible to create different clients and enter them into the network database to be recognized as users of that operator. Finally, as a result of the different connection studies carried out, an optimal network operation has been contemplated, which is capable of providing connectivity to both one and multiple connected clients at the same time, providing them in any case with an optimum quality of service.

The most important part of the project has come with the study of the 5G-EmPOWER platform and the integration of the two parts to implement a real-time testbed to test the RAN Slicing concept. From the configuration of the Agent and the new customized version of the eNB (empower-openairinterface) it has been possible to integrate the two software getting the state of the network (stop or running) from the web interface. Subsequently, the web interface has allowed to create and to configure one or several tenants in a simple and intuitive way. The result of the studies carried out with the implementation of the RAN slicing is that the controller is able to associate one or multiple users of the network to its corresponding tenant, providing them connectivity and a good quality Internet access.

Finally, in the last part of the project we have tried to implement the Multiple Operator Core Network. In this case, it has been possible to build a new operator by adjusting the configurations to the new network parameters and create a new user for this operator. As a result of the connectivity studies of this new operator it has been possible for the user to connect to the network, being able to access the Internet with a high quality of service. On the contrary, we have not got to implement the multiple operator since the eNB has been not able to associate with the two EPCs is only associated with one, probably due to configuration problems.

In conclusion, we have managed to build an LTE network integrated with a controller, which have allowed us to easily create tenants that, through the controller management, have been associated with one or several users to which to provide resources. These tenants share the same RAN, thus implementing the RAN slicing concept. Consequently, users have been able to connect to the network and access and surf the Internet with an optimum quality of service.



Despite having achieved the objective proposed for this thesis, this project offers many possibilities to continue with the study of RAN Slicing over LTE networks. Below are some proposals.

- Finish the implementation of RAN Slicing for Multiple Operator Core Network to verify that multiple users of different operators are able to associate with their corresponding tenant.
- For the MOCN case, conducting connectivity studies with multiple users varying the quality and priority parameters in database to check the behavior of the network and the quality of the service.
- Introducing a weight management algorithm in the controller to maximize the utilization of VBSes resources.
- Creating a new implementation of the architecture of the LTE network with a new software for the eNB, called SRS LTE, integrating with the controller and implementing the RAN Slicing with one and multiple operators.

## **Bibliography**

- [1] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2015–2020. USA; 2018:1-6. Available at:  
[https://www.cisco.com/c/dam/m/en\\_in/innovation/enterprise/assets/mobile-white-paper-c11-520862.pdf](https://www.cisco.com/c/dam/m/en_in/innovation/enterprise/assets/mobile-white-paper-c11-520862.pdf).
- [2] Sallent, O., Perez-Romero, J., Ferrus, R. and Agusti, R. (2017). On Radio Access Network Slicing from a Radio Resource Management Perspective. *IEEE Wireless Communications*, 24(5), pp.166-174.
- [3] Nunes B, Mendonca M, Nguyen X, Obraczka K, Turletti T. A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks. *IEEE Communications Surveys & Tutorials*. 2014;16(3):1617-1634.  
doi:10.1109/surv.2014.012214.00180.
- [4] SDN Architecture Overview. Palo Alto, CA: Open Networking Foundation; 2018:3-5. Available at: <https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/SDN-architecture-overview-1.0.pdf>. Accessed January 23, 2018.
- [5] Monnet, Q. (2018). An introduction to SDN. [online] Qmonnet.github.io. Available at: <https://qmonnet.github.io/whirl-offload/2016/07/08/introduction-to-sdn/>
- [6] DeStefano, A. (2018). Up Next on the Virtualization List: Networking | YourDailyTech. [online] YourDailyTech. Available at:  
<https://yourdailytech.com/virtualization/virtualization-up-next-on-the-virtualization-list-networking/>
- [7] Han B, Gopalakrishnan V, Ji L, Lee S. Network function virtualization: Challenges and opportunities for innovations. *IEEE Communications Magazine*. 2015;53(2):90-97. doi:10.1109/mcom.2015.7045396.
- [8] Network Functions Virtualisation. Darmstadt-Germany; 2012:3-6. Available at: [https://portal.etsi.org/NFV/NFV\\_White\\_Paper.pdf](https://portal.etsi.org/NFV/NFV_White_Paper.pdf). Accessed January 24, 2018.
- [9] Agustí Comes R, Bernardo F, Casadevall F, Ferrús R, Pérez-Romero J, Sallent O. *LTE: NUEVAS TENDENCIAS EN COMUNICACIONES MÓVILES*. [Madrid]: Fundación Vodafone España; 2010.
- [10] Advantages of LTE | Disadvantages of LTE | Long Term Evolution. Rfwireless-worldcom. 2018. Available at: <http://www.rfwireless-world.com/Terminology/Advantages-and-Disadvantages-of-LTE.html>.
- [11] Advantages of 4G-LTE. 4G LTE. 2018. Available at:  
<https://about4glte.wordpress.com/advantages/>.
- [12] Bradai A, Singh K, Ahmed T, Rasheed T. Cellular software defined networking: a framework. *IEEE Communications Magazine*. 2015;53(6):36-43.  
doi:10.1109/mcom.2015.7120043.
- [13] OpenAirInterface – 5G software alliance for democratising wireless innovation. Openairinterfaceorg. 2018. Available at: <http://www.openairinterface.org/>.



- [14] Riggio R. 5G-EmPOWER: Multi-access Edge Computing Operating System. Empowercreate-netorg. 2018. Available at: <http://empower.create-net.org/>. 2018.
- [15] GitLab. (2018). oai / openairinterface5G. [online] Available at: <https://gitlab.eurecom.fr/oai/openairinterface5g>
- [16] OPENAIRINTERFACE/openair-cn. GitHub. 2018. Available at: <https://github.com/OPENAIRINTERFACE/openair-cn>.
- [17] Openairinterface (OAI) For Experimentation In 5G.; 2018:7. Available at: <http://www.iith.ac.in/newslab/sites/default/files/Documents/ANTStutorial.pdf>.
- [18] Nikaein, N, Knopp R, Kaltenberger F et al. OpenAirInterface 4G: An Open LTE Network in a PC. 2014.
- [19] González López G, Hernández Carlón J. OAI'S LTE SYSTEM CONFIGURATION For Open Air Interface Software. Barcelona; 2017.
- [20] USRP B200/B210. Santa Clara, CA; 2018:1-2. Available at: [https://www.ettus.com/content/files/b200-b210\\_spec\\_sheet.pdf](https://www.ettus.com/content/files/b200-b210_spec_sheet.pdf).
- [21] 5G-EmPOWER. Xipieu. 2018. Available at: <https://www.xipi.eu/Infrastructures/5G-EmPOWER>.
- [22] 5g-empower/5g-empower.github.io. GitHub. 2018. Available at: <https://github.com/5g-empower/5g-empower.github.io/wiki/Overview>.
- [23] GitHub. (2018). 5g-empower/empower-runtime. [online] Available at: <https://github.com/5g-empower/empower-runtime>
- [24] GitHub. (2018). 5g-empower/empower-enb-agent. [online] Available at: <https://github.com/5g-empower/empower-enb-agent>
- [25] herlesupreeth/empower-openairinterface. GitHub. 2018. Available at: <https://github.com/herlesupreeth/empower-openairinterface>.
- [26] sysmocom site - sysmoUSIM-SJS1 SIM + USIM Card (10-pack). [online] Shop.sysmocom.de. Available at: <http://shop.sysmocom.de/products/sysmousim-sjs1>.
- [27] Mcc-mnc.com. (2018). Most up to date list of MCC and MNC codes: mobile country codes – mobile network codes. [online] Available at: <http://www.mcc-mnc.com/>.
- [28] LTE: Tracking Area (TA) and Tracking Area Update (TAU). [online] Available at: <https://www.netmanias.com/en/post/blog/5930/lte-tau/lte-tracking-area-ta-and-tracking-area-update-tau>.
- [29] phpMyAdmin. [online] phpMyAdmin. Available at: <https://www.phpmyadmin.net/>.
- [30] GitLab.Howtoconnectcotsuewithoaienb · Wiki · oai / openairinterface5G. [online] Available at: <https://gitlab.eurecom.fr/oai/openairinterface5g/wikis/HowToConnectCOTSUEwithOAleNB>.
- [31] 5g-empower/5g-empower.github.io. [online] Available at: <https://github.com/5g-empower/5g-empower.github.io/wiki/Setting-up-the-EmPOWER-Controller>.
- [32] 5g-empower/5g-empower.github.io. [online] Available at: <https://github.com/5g-empower/5g-empower.github.io/wiki/Setting-up-a-Network-Slice>.

## Annex 1. Configuration of the OAI platform

In this Annex all the configuration files used in the project are presented. Since the most part of the parameters are the same we are going to present just once the files and we will highlight the parameters modified indicating for which network or PC configuration has been used.

### HSS configurations

The configuration file **hss.conf** is exposed below:

```
HSS :
{
## MySQL mandatory options
MYSQL_server = "127.0.0.1";      # HSS S6a bind address
MYSQL_user    = "root";          # Database server login
MYSQL_pass    = "mysqlpa33w0rd"; # Database server password

MYSQL_db      = "oai_db";        # for users of 21491 network
MYSQL_db      = "oai2_db";      # for users of 21412 network

## HSS options
OPERATOR_key  = "1006020f0a478bf6b699f15c062e42b3"; # for users of 21412
                                                    network

OPERATOR_key  = "11111111111111111111111111111111"; # for users of 21491
                                                    network

RANDOM = "true";                  # True random or only
pseudo random (for subscriber vector generation)

## Freediameter options
FD_conf = "/usr/local/etc/oai/freeDiameter/hss2_fd.conf";
};
```

The configuration file **hss\_fd.conf** is exposed below:

```
# ----- Local -----
# The first parameter in this section is Identity, which will be used to
# identify this peer in the Diameter network. The Diameter protocol
# mandates that the Identity used is a valid FQDN for the peer. This
# parameter can be omitted, in that case the framework will attempt to
# use system default value (as returned by hostname --fqdn).
Identity = "hss.openair4G.eur";

# In Diameter, all peers also belong to a Realm. If the realm is not
# specified, the framework uses the part of the Identity after the first
# dot.
Realm = "openair4G.eur";
```

```
# This parameter is mandatory, even if it is possible to disable TLS for
# peers connections. A valid certificate for this Diameter Identity is
# expected.
TLS_Cred = "/usr/local/etc/oai/freeDiameter/hss.cert.pem",
"/usr/local/etc/oai/freeDiameter/hss.key.pem";
TLS_CA = "/usr/local/etc/oai/freeDiameter/hss.cacert.pem";

# Disable use of TCP protocol (only listen and connect in SCTP)
# Default : TCP enabled
No_SCTP;

# This option is ignored if freeDiameter is compiled with DISABLE_SCTP
# option.
# Prefer TCP instead of SCTP for establishing new connections.
# This setting may be overwritten per peer in peer configuration blocs.
# Default : SCTP is attempted first.
Prefer_TCP;

# Disable use of IPv6 addresses (only IP)
# Default : IPv6 enabled
No_IPv6;

# Overwrite the number of SCTP streams. This value should be kept low,
# especially if you are using TLS over SCTP, because it consumes a lot
# of resources in that case. See tickets 19 and 27 for some additional
# details on this.
# Limit the number of SCTP streams
SCTP_streams = 3;

# By default, freeDiameter acts as a Diameter Relay Agent by forwarding
# all messages it cannot handle locally. This parameter disables this
# behavior.
NoRelay;

# Use RFC3588 method for TLS protection, where TLS is negotiated after
# CER/CEA exchange is completed on the unsecure connection. The
# alternative is RFC6733 mechanism, where TLS protects also the CER/CEA
# exchange on a dedicated secure port.
# This parameter only affects outgoing connections.
# The setting can be also defined per-peer (see Peers configuration
# section).
# Default: use RFC6733 method with separate port for TLS.

#TLS_old_method;

# Number of parallel threads that will handle incoming application
# messages.
```

# This parameter may be deprecated later in favor of a dynamic number of threads depending on the load.

AppServThreads = 4;

# Specify the addresses on which to bind the listening server. This must  
# be specified if the framework is unable to auto-detect these addresses,  
# or if the auto-detected values are incorrect. Note that the list of  
# addresses is sent in CER or CEA message, so one should pay attention  
# to this parameter if some addresses should be kept hidden.

#ListenOn = "127.0.0.1";

Port = 3868;

SecPort = 5868;

# ----- Extensions -----

# Uncomment (and create rtd.conf) to specify routing table for this peer.

#LoadExtension = "rt\_default.fdx" : "rtd.conf";

# Uncomment (and create acl.conf) to allow incoming connections from  
other peers.

LoadExtension = "acl\_wl.fdx" :

"/usr/local/etc/oai/freeDiameter/acl.conf";

# Uncomment to display periodic state information

#LoadExtension = "dbg\_monitor.fdx";

# Uncomment to enable an interactive Python interpreter session.

# (see doc/dbg\_interactive.py.sample for more information)

#LoadExtension = "dbg\_interactive.fdx";

# Load the RFC4005 dictionary objects

#LoadExtension = "dict\_nasreq.fdx";

LoadExtension = "dict\_nas\_mipv6.fdx";

LoadExtension = "dict\_s6a.fdx";

# Load RFC4072 dictionary objects

#LoadExtension = "dict\_eap.fdx";

# Load the Diameter EAP server extension (requires diameap.conf)

#LoadExtension = "app\_diameap.fdx" : "diameap.conf";

# Load the Accounting Server extension (requires app\_acct.conf)

#LoadExtension = "app\_acct.fdx" : "app\_acct.conf";

# ----- Peers -----

```
# The framework will actively attempt to establish and maintain a
connection
# with the peers listed here.
# For only accepting incoming connections, see the acl_wl.fx extension.

#ConnectPeer = "ubuntu.localdomain" { ConnectTo = "127.0.0.1";
No_TLS; };
```

The configuration file **acl.conf** is exposed below:

```
# This extension is meant to allow connection from remote peers, without
# actively maintaining this connection ourselves (as it would be the
# case by declaring the peer in a ConnectPeer directive).
# The format of this file is very simple. It contains a list of peer
# names separated by spaces or newlines.
# The peer name must be a fqdn. We allow also a special "*" character as
# the first label of the fqdn, to allow all fqdn with the same domain
name.
# Example: *.example.net will allow host1.example.net and
# host2.example.net

# At the beginning of a line, the following flags are allowed (case
# sensitive) -- either or both can appear:
# ALLOW_OLD_TLS : we accept unprotected CER/CEA exchange with Inband-
Security-Id = TLS
# ALLOW_IPSEC : we accept implicitly protected connection with with
peer (Inband-Security-Id = IPSec)
# It is specified for example as:
# ALLOW_IPSEC vpn.example.net vpn2.example.net *.vpn.example.net

ALLOW_OLD_TLS *.openair4G.eur
```

## **MME configurations**

The configuration file **mme.conf** is exposed below:

```
MME :
{
    REALM = "openair4G.eur";                # YOUR REALM HERE
    # Define the limits of the system in terms of served eNB and served
    UE.
    # When the limits will be reached, overload procedure will take
    place.
    MAXEN = 2;                             # power of 2
    MAXUE = 16;                             # power of 2
    RELATIVE_CAPACITY = 10;

    EMERGENCY_ATTACH_SUPPORTED = "no";
```

```

UNAUTHENTICATED_IMSI_SUPPORTED                = "no";

# EPS network feature support
EPS_NETWORK_FEATURE_SUPPORT_IMS_VOICE_OVER_PS_SESSION_IN_S1      =
"no";    # DO NOT CHANGE
EPS_NETWORK_FEATURE_SUPPORT_EMERGENCY_BEARER_SERVICES_IN_S1_MODE =
"no";    # DO NOT CHANGE
EPS_NETWORK_FEATURE_SUPPORT_LOCATION_SERVICES_VIA_EPC             =
"no";    # DO NOT CHANGE
EPS_NETWORK_FEATURE_SUPPORT_EXTENDED_SERVICE_REQUEST              =
"no";    # DO NOT CHANGE

# Display statistics about whole system (expressed in seconds)
MME_STATISTIC_TIMER = 10;

IP_CAPABILITY = "IPV4V6";
# UNUSED, TODO

INTERTASK_INTERFACE :
{
    # max queue size per task
    ITTI_QUEUE_SIZE = 2000000;
};

S6A :
{
    S6A_CONF                =
"/usr/local/etc/oai/freeDiameter/mme_fd.conf"; # YOUR MME freeDiameter
config file path
    HSS_HOSTNAME = "hss";          # THE HSS HOSTNAME
};

# ----- SCTP definitions
SCTP :
{
    # Number of streams to use in input/output
    SCTP_INSTREAMS = 8;
    SCTP_OUTSTREAMS = 8;
};

# ----- S1AP definitions
S1AP :
{
    # outcome drop timer value (seconds)
    S1AP_OUTCOME_TIMER = 10;
};

# ----- MME served GUMMEIs

```

```
# MME code DEFAULT size = 8 bits
# MME GROUP ID size = 16 bits
# MME_CODE = [ 1, 30 , 31, 32, 33, 34, 35, 36, 56 , 29 , 8 ]; #grcm
# MME_GID = [ 32768 , 4 , 5 , 30 , 8 , 9 , 50021 ]; #grcm
GUMMEI_LIST = (
    {MCC="214" ; MNC="91"; MME_GID="4" ; MME_CODE="1"; } # for 21491
    {MCC="214" ; MNC="12"; MME_GID="4" ; MME_CODE="1"; } # for 21412
);
# ----- MME served TAIs
# TA (mcc.mnc:tracking area code) DEFAULT = 208.34:1
# max values = 999.999:65535
# maximum of 16 TAIs, comma separated
# !!! Actually use only one PLMN
TAI_LIST = (
    {MCC="214" ; MNC="91"; TAC = "1"; } # for 21491
    {MCC="214" ; MNC="12"; TAC = "1"; } # for 21412
);
NAS :
{
# 3GPP TS 33.401 section 7.2.4.3 Procedures for NAS algorithm selection
# decreasing preference goes from left to right
# ORDERED_SUPPORTED_INTEGRITY_ALGORITHM_LIST = [ "EIA2" , "EIA1" ,
# "EIA0" ];
ORDERED_SUPPORTED_CIPHERING_ALGORITHM_LIST = [ "EEA0" , "EEA1" ,
"EEA2" ];
# EMM TIMERS
# T3402 start:
# At attach failure and the attempt counter is equal to 5.
# At tracking area updating failure and the attempt counter is
# equal to 5.
# T3402 stop:
# ATTACH REQUEST sent, TRACKING AREA REQUEST sent.
# On expiry:
# Initiation of the attach procedure, if still required or TAU
# procedure
# attached for emergency bearer services.
T3402 = 1 # in minutes (default is 12 minutes)

# T3412 start:
# In EMM-REGISTERED, when EMM-CONNECTED mode is left.
# T3412 stop:
# When entering state EMM-DEREGISTERED or when entering EMM-
# CONNECTED mode.
# On expiry:
# Initiation of the periodic TAU procedure if the UE is not
# attached for emergency bearer services. Implicit detach from
# network if the UE is Attached for emergency bearer services.
T3412 = 54 # in minutes (default is 54 minutes, network
```



```
# dependent)
# T3422 start: DETACH REQUEST sent
# T3422 stop: DETACH ACCEPT received
# ON THE 1st, 2nd, 3rd, 4th EXPIRY: Retransmission of DETACH
# REQUEST
T3422 = 6 # in seconds (default is 6s)

# T3450 start:
# ATTACH ACCEPT sent, TRACKING AREA UPDATE ACCEPT sent with GUTI,
# TRACKING AREA UPDATE ACCEPT sent with TMSI,
# GUTI REALLOCATION COMMAND sent
# T3450 stop:
# ATTACH COMPLETE received, TRACKING AREA UPDATE COMPLETE
# received, GUTI REALLOCATION COMPLETE received
# ON THE 1st, 2nd, 3rd, 4th EXPIRY: Retransmission of the same
# message type
T3450 = 6 # in seconds (default is 6s)

# T3460 start: AUTHENTICATION REQUEST sent, SECURITY MODE
# COMMAND sent
# T3460 stop:
# AUTHENTICATION RESPONSE received, AUTHENTICATION FAILURE
# received,
# SECURITY MODE COMPLETE received, SECURITY MODE REJECT received
# ON THE 1st, 2nd, 3rd, 4th EXPIRY: Retransmission of the same
# message type
T3460 = 6 # in seconds (default is 6s)

# T3470 start: IDENTITY REQUEST sent
# T3470 stop: IDENTITY RESPONSE received
# ON THE 1st, 2nd, 3rd, 4th EXPIRY: Retransmission of IDENTITY
# REQUEST
T3470 = 6 # in seconds (default is 6s)

# ESM TIMERS
T3485 = 8 # UNUSED in seconds (default is 8s)
T3486 = 8 # UNUSED in seconds (default is 8s)
T3489 = 4 # UNUSED in seconds (default is 4s)
T3495 = 8 # UNUSED in seconds (default is 8s)
};

NETWORK_INTERFACES :
{
# MME binded interface for S1-C or S1-MME communication (S1AP), can be
# ethernet interface, virtual ethernet interface, we don't advise
# wireless interfaces
```

```
# 2 PCs configurations just OAI
MME_INTERFACE_NAME_FOR_S1_MME      = "eth1";
MME_IPV4_ADDRESS_FOR_S1_MME         = "192.68.1.3/24";
# 1 PC configuration 21491 network
MME_INTERFACE_NAME_FOR_S1_MME      = "lo";
MME_IPV4_ADDRESS_FOR_S1_MME         = "127.0.0.20/8";
# 2 PC configuration 21491 network
MME_INTERFACE_NAME_FOR_S1_MME      = "eth0";
MME_IPV4_ADDRESS_FOR_S1_MME         = "147.83.105.150/32";
# 2 PC configuration 21412 network
MME_INTERFACE_NAME_FOR_S1_MME      = "eth0";
MME_IPV4_ADDRESS_FOR_S1_MME         = "147.83.105.253/32";

# MME binded interface for S11 communication (GTPV2-C)
# 2 PCs configurations just OAI
MME_INTERFACE_NAME_FOR_S11_MME     = "lo";
MME_IPV4_ADDRESS_FOR_S11_MME       = "127.0.11.1/8";
# 1 and 2 PCs configuration 21491 network
MME_INTERFACE_NAME_FOR_S11_MME     = "lo";
MME_IPV4_ADDRESS_FOR_S11_MME       = "127.0.0.20/8";
# 2 PC configuration 21412 network
MME_INTERFACE_NAME_FOR_S11_MME     = "lo";
MME_IPV4_ADDRESS_FOR_S11_MME       = "127.0.0.21/8";

MME_PORT_FOR_S11_MME                = 2123;
};

LOGGING :
{
# OUTPUT choice in { "CONSOLE", "SYSLOG", `path to file`, ``IPv4@`:`TCP
# port num`"}
# `path to file` must start with '.' or '/'
# if TCP stream choice, then you can easily dump the traffic on the
# remote or local host: nc -l `TCP port num` > received.txt
    OUTPUT          = "CONSOLE";
    #OUTPUT          = "SYSLOG";
    #OUTPUT          = "/tmp/mme.log";
    #OUTPUT          = "127.0.0.1:5656";

# THREAD_SAFE choice in { "yes", "no" } means use of thread safe
# intermediate buffer then a single thread pick each message log one
# by one to flush it to the chosen output
    THREAD_SAFE     = "yes";

# COLOR choice in { "yes", "no" } means use of ANSI styling codes or no
    COLOR           = "yes";
```

```
# Log level choice in { "EMERGENCY", "ALERT", "CRITICAL", "ERROR",
"WARNING", "NOTICE", "INFO", "DEBUG", "TRACE"}
    SCTP_LOG_LEVEL      = "TRACE";
    S11_LOG_LEVEL       = "TRACE";
    GTPV2C_LOG_LEVEL    = "TRACE";
    UDP_LOG_LEVEL       = "TRACE";
    S1AP_LOG_LEVEL      = "TRACE";
    NAS_LOG_LEVEL       = "TRACE";
    MME_APP_LOG_LEVEL   = "TRACE";
    S6A_LOG_LEVEL       = "TRACE";
    UTIL_LOG_LEVEL      = "TRACE";
    MSC_LOG_LEVEL       = "ERROR";
    ITTI_LOG_LEVEL      = "ERROR";
    MME_SCENARIO_PLAYER_LOG_LEVEL = "TRACE";

    # ASN1 VERBOSITY: none, info, annoying
    # for S1AP protocol
    ASN1_VERBOSITY      = "none";
};
TESTING :
{
    # file should be copied here from source tree by following command:
    # run_mme --install-mme-files ...
    SCENARIO_FILE = "/usr/local/share/oai/test/MME/no_regression.xml";
};

S-GW_LIST_SELECTION = (
# 2 PCs configurations just OAI
{ID="tac-lb01.tac-hb00.tac.epc.mnc091.mcc214.3gppnetwork.org";
SGW_IPV4_ADDRESS_FOR_S11="127.0.11.2/8";}
# 1 and 2 PCs config. 21491 network
{ID="tac-lb01.tac-hb00.tac.epc.mnc091.mcc214.3gppnetwork.org";
SGW_IPV4_ADDRESS_FOR_S11="127.0.0.30/8";}
# 2 PC config. 21412 network
{ID="tac-lb01.tac-hb00.tac.epc.mnc12.mcc214.3gppnetwork.org";
SGW_IPV4_ADDRESS_FOR_S11="127.0.0.31/8";}

);
};
```

The configuration file **mme\_fd.conf** is exposed below:

```
# ----- Local -----

# Uncomment if the framework cannot resolv it.
Identity = "nano.openair4G.eur";      # used in 2 PCs configuration and
                                      # 2 EPCs (21491 network).
Identity = "calisson.openair4G.eur";  # used in 1 PC configuration and
                                      # 2 EPCs (21412 network).
```

```
Realm = "openair4G.eur";

# TLS configuration (see previous section)
TLS_Cred = "/usr/local/etc/oai/freeDiameter/mme.cert.pem",
           "/usr/local/etc/oai/freeDiameter/mme.key.pem";
TLS_CA    = "/usr/local/etc/oai/freeDiameter/mme.cacert.pem";

# Disable use of TCP protocol (only listen and connect in SCTP)
# Default : TCP enabled
No_SCTP;

# This option is ignored if freeDiameter is compiled with DISABLE_SCTP
# option.
# Prefer TCP instead of SCTP for establishing new connections.
# This setting may be overwritten per peer in peer configuration blocs.
# Default : SCTP is attempted first.
Prefer_TCP;

No_IPv6;

# Overwrite the number of SCTP streams. This value should be kept low,
# especially if you are using TLS over SCTP, because it consumes a lot
# of resources in that case. See tickets 19 and 27 for some additional
# details on this.
# Limit the number of SCTP streams
SCTP_streams = 3;

# By default, freeDiameter acts as a Diameter Relay Agent by forwarding
# all messages it cannot handle locally. This parameter disables this
# behavior.
NoRelay;

# Use RFC3588 method for TLS protection, where TLS is negotiated after
# CER/CEA exchange is completed on the unsecure connection. The
# alternative is RFC6733 mechanism, where TLS protects also the CER/CEA
# exchange on a dedicated secure port.
# This parameter only affects outgoing connections.
# The setting can be also defined per-peer (see Peers configuration
# section).
# Default: use RFC6733 method with separate port for TLS.

#TLS_old_method;

AppServThreads = 4;

# Specify the addresses on which to bind the listening server. This must
# be specified if the framework is unable to auto-detect these addresses,
# or if the auto-detected values are incorrect. Note that the list of
```

```
# addresses is sent in CER or CEA message, so one should pay attention
# to this parameter if some addresses should be kept hidden.
#ListenOn = ;

Port = 3870;
SecPort = 5870;

# ----- Extensions -----

# Uncomment (and create rtd.conf) to specify routing table for this peer.
#LoadExtension = "rt_default.fdx" : "rtd.conf";

# Uncomment (and create acl.conf) to allow incoming connections from
other peers.
#LoadExtension = "acl_wl.fdx" : "acl.conf";

# Uncomment to display periodic state information
#LoadExtension = "dbg_monitor.fdx";

# Uncomment to enable an interactive Python interpreter session.
# (see doc/dbg_interactive.py.sample for more information)
#LoadExtension = "dbg_interactive.fdx";

# Load the RFC4005 dictionary objects
#LoadExtension = "dict_nasreq.fdx";

LoadExtension = "dict_nas_mipv6.fdx";
LoadExtension = "dict_s6a.fdx";

# Load RFC4072 dictionary objects
#LoadExtension = "dict_eap.fdx";

# Load the Diameter EAP server extension (requires diameap.conf)
#LoadExtension = "app_diameap.fdx" : "diameap.conf";

# Load the Accounting Server extension (requires app_acct.conf)
#LoadExtension = "app_acct.fdx" : "app_acct.conf";

# ----- Peers -----

# The framework will actively attempt to establish and maintain a
connection with the peers listed here.
# For only accepting incoming connections, see the acl_wl.fx extension.

# ConnectPeer
# Declare a remote peer to which this peer must maintain a connection.
```

```
# In addition, this allows specifying non-default parameters for this
# peer only (for example disable SCTP with this peer, or use RFC3588-
# flavour TLS).
# Note that by default, if a peer is not listed as a ConnectPeer entry,
# an incoming connection from this peer will be rejected. If you want to
# accept ncoming connections from other peers, see the acl_wl.fdx?
# extension which allows exactly this.
```

```
ConnectPeer= "hss.openair4G.eur" { ConnectTo = "127.0.0.1"; No_SCTP ;
No_IPv6; Prefer_TCP; No_TLS; port = 3868; realm = "openair4G.eur";};
```

### SPGW configurations

The configuration file **spgw.conf** is exposed below:

```
S-GW :
{
    NETWORK_INTERFACES :
    {
# S-GW binded interface for S11 communication (GTPV2-C), if none
selected the ITTI message interface is used
# 2 PCs configurations just OAI
SGW_INTERFACE_NAME_FOR_S1 = "lo";
SGW_IPV4_ADDRESS_FOR_S11 = "127.0.11.2/8"
# 1 and 2 PCs config. 21491 network
SGW_INTERFACE_NAME_FOR_S1 = "lo";
SGW_IPV4_ADDRESS_FOR_S11 = "127.0.0.30/8"
# 2 PC config. 21412 network
SGW_INTERFACE_NAME_FOR_S1 = "lo";
SGW_IPV4_ADDRESS_FOR_S11 = "127.0.0.31/8"

# S-GW binded interface for S1-U communication (GTPV1-U) can be ethernet
# interface, virtual ethernet interface, we don't advise wireless
# interfaces
# 2 PCs configurations just OAI
SGW_INTERFACE_NAME_FOR_S1U_S12_S4_UP = "eth1";
SGW_IPV4_ADDRESS_FOR_S1U_S12_S4_UP = "192.68.1.3/24";
# 1 and config. 21491 network
SGW_INTERFACE_NAME_FOR_S1U_S12_S4_UP = "lo";
SGW_IPV4_ADDRESS_FOR_S1U_S12_S4_UP = "127.0.0.20/8";
# 2 PCs config. 21491 network
SGW_INTERFACE_NAME_FOR_S1U_S12_S4_UP = "eth0";
SGW_IPV4_ADDRESS_FOR_S1U_S12_S4_UP = "147.83.105.253/32";
# 2 PC config. 21412 network
SGW_INTERFACE_NAME_FOR_S1U_S12_S4_UP = "eth0";
SGW_IPV4_ADDRESS_FOR_S1U_S12_S4_UP = "147.83.105.150/32";
```

```

SGW_IPV4_PORT_FOR_S1U_S12_S4_UP = 2152;

# S-GW binded interface for S5 or S8 communication, not implemented, so
# leave it to none
    SGW_INTERFACE_NAME_FOR_S5_S8_U = "none"; # (NOT IMPLEMENTED YET)
    SGW_IPV4_ADDRESS_FOR_S5_S8_UP = "0.0.0.0/24"; # (NOT IMPLEMENTED
YET)
};
INTERTASK_INTERFACE :
{
    # max queue size per task
    ITTI_QUEUE_SIZE = 2000000; # INTEGER
};
LOGGING :
{
    # OUTPUT choice in { "CONSOLE", "SYSLOG", `path to file`,
    # "`IPv4@`:`TCP port num`"}
    # `path to file` must start with '.' or '/'
    # if TCP stream choice, then you can easily dump the traffic on
    # the remote or local host: nc -l `TCP port num` > received.txt
    OUTPUT = "CONSOLE";          # see 3 lines above
    #OUTPUT = "SYSLOG";          # see 4 lines above
    #OUTPUT = "/tmp/spgw.log";   # see 5 lines above
    #OUTPUT = "127.0.0.1:5656"; # see 6 lines above

    # THREAD_SAFE choice in { "yes", "no" } means use of thread safe
    # intermediate buffer then a single thread pick each message log
    # one by one to flush it to the chosen output
    THREAD_SAFE          = "no";

    # COLOR choice in { "yes", "no" } means use of ANSI styling
    # codes or no
    COLOR                 = "yes";

    # Log level choice in { "EMERGENCY", "ALERT", "CRITICAL",
    # "ERROR", "WARNING", "NOTICE", "INFO", "DEBUG", "TRACE"}
    ASYNC_SYSTEM          = "TRACE";
    UDP_LOG_LEVEL         = "TRACE";
    GTPV1U_LOG_LEVEL      = "TRACE";
    GTPV2C_LOG_LEVEL      = "TRACE";
    SPGW_APP_LOG_LEVEL    = "TRACE";
    S11_LOG_LEVEL         = "TRACE";
    UTIL_LOG_LEVEL        = "TRACE";
    ITTI_LOG_LEVEL        = "WARNING";
};
};
P-GW =

```



```
{
NETWORK_INTERFACES :
{
    # P-GW binded interface for S5 or S8 communication, not
    # implemented, so leave it to none
    PGW_INTERFACE_NAME_FOR_S5_S8 = "none"; # (NOT IMPLEMENTED YET)
    # P-GW binded interface for SGI (egress/ingress internet
    # traffic)
    # 2 PCs configurations just OAI
    PGW_INTERFACE_NAME_FOR_SGI = "eth1";
    PGW_IPV4_ADDRESS_FOR_SGI = "192.68.1.3/24";
    # 1 and config. 21491 network
    PGW_INTERFACE_NAME_FOR_SGI = "eth0";
    PGW_IPV4_ADDRESS_FOR_SGI = "147.83.105.150/32";
    # 2 PCs config. 21491 network
    PGW_INTERFACE_NAME_FOR_SGI = "eth0";
    PGW_IPV4_ADDRESS_FOR_SGI = "147.83.105.253/32";
    # 2 PC config. 21412 network
    PGW_INTERFACE_NAME_FOR_SGI = "eth0";
    PGW_IPV4_ADDRESS_FOR_SGI = "147.83.105.150/32";

    PGW_MASQUERADE_SGI = "yes";
    UE_TCP_MSS_CLAMPING = "no"; # STRING, {"yes", "no"}.
};

# Pool of UE assigned IP addresses
# Do not make IP pools overlap
# first IPv4 address X.Y.Z.1 is reserved for GTP network device on
# SPGW
# Normally no more than 16 pools allowed, but since recent GTP
# kernel module use, only one pool allowed (TODO).
IP_ADDRESS_POOL :
{
    IPV4_LIST = (
        "172.16.0.0/12"
    # "192.188.2.0/24" # YOUR NETWORK CONFIG HERE
    );
};

# DNS address communicated to UEs
DEFAULT_DNS_IPV4_ADDRESS = "8.8.8.8";
DEFAULT_DNS_SEC_IPV4_ADDRESS = "8.8.4.4";

# Non standard feature, normally should be set to "no", but you may need
# to set to yes for UE that do not explicitly request a PDN address
# through NAS signalling
FORCE_PUSH_PROTOCOL_CONFIGURATION_OPTIONS = "yes";
UE_MTU = 1400
```

```

GTPV1U_REALIZATION = "GTP_KERNEL_MODULE";
# STRING {"NO_GTP_KERNEL_AVAILABLE", "GTP_KERNEL_MODULE", "GTP_KERNEL"}.
# In a container you may not be able to unload/load kernel modules.

PCEF :
{
    PCEF_ENABLED = "yes"; # "no"; # STRING, {"yes", "no"}, if yes
then all parameters bellow will/should be taken into account
    TRAFFIC_SHAPPING_ENABLED = "yes";
# STRING, {"yes", "no"}, TODO, should finally work for egress but only
# on ingress bearers and not on ingress SDF flows
    TCP_ECN_ENABLED = "yes";
# STRING, {"yes", "no"}, TCP explicit congestion notification
    AUTOMATIC_PUSH_DEDICATED_BEARER_PCC_RULE= 0;
# INTEGER [ 0..n], SDF identifier (Please check with enum sdf_id_t in
# pgw_pcef_emulation.h,

# 0 = No push of dedicated bearer

# 17 = SDF_ID_GBR_VOLTE_16K, // see corresponding
TFT and QOS params in pgw_pcef_emulation.h
# 18 = SDF_ID_GBR_VOLTE_24K, // see corresponding
TFT and QOS params in pgw_pcef_emulation.h
# 19 = SDF_ID_GBR_VOLTE_40K, // see corresponding
TFT and QOS params in pgw_pcef_emulation.h
# 20 = SDF_ID_GBR_VOLTE_64K, // see corresponding
TFT and QOS params in pgw_pcef_emulation.h

# 21 = SDF_ID_GBR_VILTE_192K, // see corresponding
TFT and QOS params in pgw_pcef_emulation.h
# 22 = SDF_ID_GBR_VILTE_384K, // see corresponding
TFT and QOS params in pgw_pcef_emulation.h
# 23 = SDF_ID_GBR_VILTE_768K, // see corresponding
TFT and QOS params in pgw_pcef_emulation.h
# 24 = SDF_ID_GBR_VILTE_2M, // see corresponding
TFT and QOS params in pgw_pcef_emulation.h
# 25 = SDF_ID_GBR_VILTE_4M, // see corresponding
TFT and QOS params in pgw_pcef_emulation.h
# 26 = SDF_ID_GBR_NON_CONVERSATIONAL_VIDEO_256K, // see corresponding
TFT and QOS params in pgw_pcef_emulation.h
# 27 = SDF_ID_GBR_NON_CONVERSATIONAL_VIDEO_512K, // see corresponding
TFT and QOS params in pgw_pcef_emulation.h
# 28 = SDF_ID_GBR_NON_CONVERSATIONAL_VIDEO_1M, // see corresponding
TFT and QOS params in pgw_pcef_emulation.h
# 29 = SDF_ID_NGBR_IMS_SIGNALLING, // see corresponding
TFT and QOS params in pgw_pcef_emulation.h
# 30 = SDF_ID_NGBR_DEFAULT_PREMIUM, // see corresponding
TFT and QOS params in pgw_pcef_emulation.h

```

```
# 31 = SDF_ID_NGBR_DEFAULT, // see corresponding
TFT and QOS params in pgw_pcef_emulation.h
# 32 = SDF_ID_TEST_PING // see corresponding
TFT and QOS params in pgw_pcef_emulation.h

DEFAULT_BEARER_STATIC_PCC_RULE = 31;
# SDF identifier for default bearer
PUSH_STATIC_PCC_RULES = (31);
# List of SDF identifiers

# Waiting for HSS APN-AMBR IE ...
APN_AMBR_UL = 500000;
# Maximum UL bandwidth that can be used by non guaranteed bit rate
# traffic in Kbits/seconds.
APN_AMBR_DL = 500000;
# Maximum DL bandwidth that can be used by non guaranteed bit rate
# traffic in Kbits/seconds.
};
};
```

### **eNB configurations**

The configuration file **enb.band7.tm1.usrpb210.conf** is exposed below:

```
Active_eNBs = ( "eNB_Eurecom_LTEBox");
# Asn1_verbosity, choice in: none, info, annoying
Asn1_verbosity = "none";

eNBs =
(
{
////////// Identification parameters:
eNB_ID = 0xe00;

cell_type = "CELL_MACRO_ENB";

eNB_name = "eNB_Eurecom_LTEBox";

// Tracking area code, 0x0000 and 0xffffe are reserved values
tracking_area_code = "1";

mobile_country_code = "214";
mobile_network_code = "91";

////////// Physical parameters:

component_carriers = (
{
node_function =
"eNodeB_3GPP";
```

```

node_timing
"synch_to_ext_device";
node_synch_ref
    frame_type          = "FDD";
    tdd_config          = 3;
    tdd_config_s        = 0;
    prefix_type         = "NORMAL";
    eutra_band          = 7;
    downlink_frequency   = 2660000000L;
    uplink_frequency_offset = -120000000;
    Nid_cell             = 0; // Default value was 2 This
parameter is related to PSS
    N_RB_DL              = 50;
    Nid_cell_mbsfn       = 0;
    nb_antenna_ports     = 1;
    nb_antennas_tx       = 1;
    nb_antennas_rx       = 1;
    tx_gain              = 90;
    rx_gain              = 108;
    //rx_gain            = 108;
//(best working value)
    prach_root           = 0;
    prach_config_index   = 0;
    prach_high_speed     = "DISABLE";
    prach_zero_correlation = 1;
    prach_freq_offset    = 2;
    pucch_delta_shift    = 1;
    pucch_nRB_CQI        = 1;
    pucch_nCS_AN         = 0;
    pucch_n1_AN          = 32;
    pdsch_referenceSignalPower = -24;
    pdsch_p_b            = 0;
    pusch_n_SB           = 1;
    pusch_enable64QAM     = "DISABLE";
    pusch_hoppingMode    =
"interSubFrame";
    pusch_hoppingOffset  = 0;
    pusch_groupHoppingEnabled = "ENABLE";
    pusch_groupAssignment = 0;
    pusch_sequenceHoppingEnabled = "DISABLE";
    pusch_nDMRS1         = 1;
    phich_duration       = "NORMAL";
    phich_resource       = "ONESIXTH";
    srs_enable           = "DISABLE";
/* srs_BandwidthConfig
    srs_SubframeConfig
    srs_ackNackST
    srs_MaxUpPts

```

```

    pusch_p0_Nominal          = -90;
    pusch_alpha               = "AL1";
    pucch_p0_Nominal          = -108;
    msg3_delta_Preamble       = 6;
    pucch_deltaF_Format1      = "deltaF2";
    pucch_deltaF_Format1b     = "deltaF3";
    pucch_deltaF_Format2      = "deltaF0";
    pucch_deltaF_Format2a     = "deltaF0";
    pucch_deltaF_Format2b     = "deltaF0";

    rach_numberOfRA_Preambles = 64;
    rach_preamblesGroupAConfig = "DISABLE";
/*
    rach_sizeOfRA_PreamblesGroupA = ;
    rach_messageSizeGroupA        = ;
    rach_messagePowerOffsetGroupB = ;
*/
    rach_powerRampingStep        = 4;
    rach_preambleInitialReceivedTargetPower = -104;
    rach_preambleTransMax        = 10;
    rach_raResponseWindowSize    = 10;
    rach_macContentionResolutionTimer = 48;
    rach_maxHARQ_Msg3Tx         = 4;

    pcch_default_PagingCycle     = 128;
    pcch_nB                      = "oneT";
    bcch_modificationPeriodCoeff = 2;
    ue_TimersAndConstants_t300    = 1000;
    ue_TimersAndConstants_t301    = 1000;
    ue_TimersAndConstants_t310    = 1000;
    ue_TimersAndConstants_t311    = 10000;
    ue_TimersAndConstants_n310    = 20;
    ue_TimersAndConstants_n311    = 1;
    ue_TransmissionMode           = 1;
}
);

srb1_parameters :
{
    # timer_poll_retransmit = (ms) [5, 10, 15, 20,... 250, 300,
350, ... 500]
    timer_poll_retransmit    = 80;

    # timer_reordering = (ms) [0,5, ... 100, 110, 120, ... ,200]
    timer_reordering         = 35;

```

```
# timer_reordering = (ms) [0,5, ... 250, 300, 350, ... ,500]
timer_status_prohibit    = 0;

# poll_pdu = [4, 8, 16, 32 , 64, 128, 256, infinity(>10000)]
poll_pdu                  = 4;

#                               poll_byte                               =                               (kB)
[25,50,75,100,125,250,375,500,750,1000,1250,1500,2000,3000,infinity(>100
00)]
poll_byte                  = 99999;

# max_retx_threshold = [1, 2, 3, 4 , 6, 8, 16, 32]
max_retx_threshold        = 4;
}

# ----- SCTP definitions
SCTP :
{
    # Number of streams to use in input/output
    SCTP_INSTREAMS = 2;
    SCTP_OUTSTREAMS = 2;
};

////////// MME parameters:
mme_ip_address            = (
    # 2 PCs configuration just OAI
    {ipv4 = "192.68.1.3";
     ipv6="192:168:30::17";
     active="yes";
     preference="ipv4";
    }
    # 1 PCs configuration 21491 network
    {ipv4 = "192.68.1.3";
     ipv6="192:168:30::17";
     active="yes";
     preference="ipv4";
    }
);

NETWORK_INTERFACES :
{
    # 2 PCs configuration just OAI
    ENB_INTERFACE_NAME_FOR_S1_MME      = "eth0";
    ENB_IPV4_ADDRESS_FOR_S1_MME        = "192.68.1.5/24";

    ENB_INTERFACE_NAME_FOR_S1U         = "eth0";
    ENB_IPV4_ADDRESS_FOR_S1U           = "192.68.1.5/24";
    # 1 PC configuration 21491 network
```

```

ENB_INTERFACE_NAME_FOR_S1_MME          = "eth0";
ENB_IPV4_ADDRESS_FOR_S1_MME            = "192.68.1.5/24";

ENB_INTERFACE_NAME_FOR_S1U              = "eth0";
ENB_IPV4_ADDRESS_FOR_S1U                = "192.68.1.5/24";

    ENB_PORT_FOR_S1U                     = 2152; # Spec 2152
};

log_config :
{
    global_log_level                      = "info";
    global_log_verbosity                  = "medium";
    hw_log_level                          = "info";
    hw_log_verbosity                      = "medium";
    phy_log_level                         = "info";
    phy_log_verbosity                    = "medium";
    mac_log_level                         = "info";
    mac_log_verbosity                    = "medium";
    rlc_log_level                         = "info";
    rlc_log_verbosity                    = "medium";
    pdcp_log_level                       = "info";
    pdcp_log_verbosity                   = "medium";
    rrc_log_level                        = "debug";
    rrc_log_verbosity                    = "high";
};

}
);

```

The configuration file **enb.band7.tm1.usrpb210.epc.conf** is exposed below:

```

Active_eNBs = ( "eNB_Eurecom_LTEBox");
# Asn1_verbosity, choice in: none, info, annoying
Asn1_verbosity = "none";

eNBs =
(
{
    /////////// Identification parameters:
    eNB_ID      = 0xe00;

    cell_type = "CELL_MACRO_ENB";

    eNB_name = "eNB_Eurecom_LTEBox";

    // Tracking area code, 0x0000 and 0xffffe are reserved values
    tracking_area_code = "1";

```



```

mobile_country_code = "214";
mobile_network_code = "12";

///// Max of 6 PLMN IDS can be broadcasted
multiple_OCN          = "True";
plmn_ids = ( {
    mobile_country_code = "214";
    mobile_network_code = "12";
},
{
    mobile_country_code = "214";
    mobile_network_code = "91";
}
);

////////// Physical parameters:
component_carriers = (
{
    node_function                = "eNodeB_3GPP";
    node_timing                  = "synch_to_ext_device";
    node_synch_ref               = 0;
    frame_type                   = "FDD";
    tdd_config                   = 3;
    tdd_config_s                 = 0;
    prefix_type                  = "NORMAL";
    eutra_band                   = 7;
    downlink_frequency           = 2660000000L;
    uplink_frequency_offset      = -120000000;
    Nid_cell                     = 0; // Default value was 2 This
parameter is related to PSS
    N_RB_DL                     = 50;
    Nid_cell_mbsfn               = 0;
    nb_antenna_ports             = 1;
    nb_antennas_tx               = 1;
    nb_antennas_rx               = 1;
    tx_gain                      = 90;
    rx_gain                      = 108;
    //rx_gain                    = 108; //(best working
value)
    prach_root                   = 0;
    prach_config_index           = 0;
    prach_high_speed             = "DISABLE";
    prach_zero_correlation       = 1;
    prach_freq_offset            = 2;
    pucch_delta_shift            = 1;
    pucch_nRB_CQI                = 1;
    pucch_nCS_AN                 = 0;
    pucch_n1_AN                  = 32;

```

```

        pdsch_referenceSignalPower          = -24;
        pdsch_p_b                          = 0;
        pusch_n_SB                        = 1;
        pusch_enable64QAM                  = "DISABLE";
        pusch_hoppingMode                    =
"interSubFrame";
        pusch_hoppingOffset                  = 0;
        pusch_groupHoppingEnabled            = "ENABLE";
        pusch_groupAssignment                = 0;
        pusch_sequenceHoppingEnabled        = "DISABLE";
        pusch_nDMRS1                        = 1;
        phich_duration                      = "NORMAL";
        phich_resource                      = "ONESIXTH";
        srs_enable                          = "DISABLE";
/*   srs_BandwidthConfig                    =;
    srs_SubframeConfig                      =;
    srs_ackNackST                          =;
    srs_MaxUpPts                            =;*/

        pusch_p0_Nominal                    = -90;
        pusch_alpha                        = "AL1";
        pucch_p0_Nominal                    = -108;
        msg3_delta_Preamble                  = 6;
        pucch_deltaF_Format1                = "deltaF2";
        pucch_deltaF_Format1b                = "deltaF3";
        pucch_deltaF_Format2                = "deltaF0";
        pucch_deltaF_Format2a                = "deltaF0";
        pucch_deltaF_Format2b                = "deltaF0";

        rach_numberOfRA_Preambles            = 64;
        rach_preamblesGroupAConfig          = "DISABLE";
/*
        rach_sizeOfRA_PreamblesGroupA        = ;
        rach_messageSizeGroupA                = ;
        rach_messagePowerOffsetGroupB        = ;
*/
        rach_powerRampingStep                = 4;
        rach_preambleInitialReceivedTargetPower = -104;
        rach_preambleTransMax                = 10;
        rach_raResponseWindowSize            = 10;
        rach_macContentionResolutionTimer    = 48;
        rach_maxHARQ_Msg3Tx                  = 4;

        pcch_default_PagingCycle              = 128;
        pcch_nB                              = "oneT";
        bcch_modificationPeriodCoeff          = 2;
        ue_TimersAndConstants_t300            = 1000;
        ue_TimersAndConstants_t301            = 1000;

```

```

        ue_TimersAndConstants_t310          = 1000;
        ue_TimersAndConstants_t311          = 10000;
        ue_TimersAndConstants_n310          = 20;
        ue_TimersAndConstants_n311          = 1;
        ue_TransmissionMode                  = 1;
    }
);

srb1_parameters :
{
    # timer_poll_retransmit = (ms) [5, 10, 15, 20,... 250, 300,
350, ... 500]
    timer_poll_retransmit      = 80;

    # timer_reordering = (ms) [0,5, ... 100, 110, 120, ... ,200]
    timer_reordering          = 35;

    # timer_reordering = (ms) [0,5, ... 250, 300, 350, ... ,500]
    timer_status_prohibit     = 0;

    # poll_pdu = [4, 8, 16, 32 , 64, 128, 256, infinity(>10000)]
    poll_pdu                   = 4;

    # poll_byte = (kB)
[25,50,75,100,125,250,375,500,750,1000,1250,1500,2000,3000,infinity(>100
00)]
    poll_byte                  = 99999;

    # max_retx_threshold = [1, 2, 3, 4 , 6, 8, 16, 32]
    max_retx_threshold         = 4;
}

# ----- Sctp definitions
Sctp :
{
    # Number of streams to use in input/output
    Sctp_INSTREAMS = 2;
    Sctp_OUTSTREAMS = 2;
};

////////// MME parameters:
mme_ip_address      = ( {ipv4 = "147.83.105.150";
                        ipv6="192:168:30::17";
                        active="yes";
                        preference="ipv4";
                        },
                        {ipv4 = "147.83.105.253";

```

```

        ipv6="192:168:30::18";
        active="yes";
        preference="ipv4";
    }
};

NETWORK_INTERFACES :
{
    ENB_INTERFACE_NAME_FOR_S1_MME                = "eth1";
    ENB_IPV4_ADDRESS_FOR_S1_MME                  = "147.83.105.220/32";

    ENB_INTERFACE_NAME_FOR_S1U                    = "eth1";
    ENB_IPV4_ADDRESS_FOR_S1U                      = "147.83.105.220/32";
    ENB_PORT_FOR_S1U                              = 2152; # Spec 2152
};
log_config :
{
    global_log_level                            = "info";
    global_log_verbosity                        = "medium";
    hw_log_level                                = "info";
    hw_log_verbosity                            = "medium";
    phy_log_level                               = "info";
    phy_log_verbosity                           = "medium";
    mac_log_level                               = "info";
    mac_log_verbosity                           = "medium";
    rlc_log_level                               = "info";
    rlc_log_verbosity                           = "medium";
    pdcp_log_level                              = "info";
    pdcp_log_verbosity                          = "medium";
    rrc_log_level                               = "debug";
    rrc_log_verbosity                           = "high";
};
}
);

```

## Annex 2. Manufacturer SIM parameters

The following table show the parameters fixed by the manufacturer in the different SIM card that we used. The vendor parameters are important to know the ADM1, among others.

No	IMSI	ICCID	PIN1	PUK1	Ki	OPC	ADM1
1	901700000 009910	8988211000000 099104	302 3	11849 994	4A8AD0DE2E4948EE9378 C8311906C73A	8940B111A8EE03EFF4938E E2E23AC056	6647630 0
2	901700000 009911	8988211000000 099112	536 1	25257 221	29C7D674257AF3B0C6C9F 7879663F5FB	926627AAB7C1B503F7F12B 6B93E2702B	8641619 4
3	901700000 009912	8988211000000 099120	415 0	99652 493	5B868E2B30C61190ABEFB A1CA6F6D56F	B3425076F23BA6054557FA 359B4F9C0C	2016846 2
4	901700000 009913	8988211000000 099138	965 1	50388 899	CF4D1ACE1C99C63FF0922 322D9CA740A	3FDEBD82A5B5B7B2C03086 33D882ECF0	7766877 2
5	901700000 009914	8988211000000 099146	307 7	86721 501	4403368D7F3ABEE7CBED 3155D134ABFA	2584E3BA38097A30BF4888 E352E998B8	7131224 5
6	901700000 009915	8988211000000 099153	120 6	12579 930	3EC6CED8F811540354558 A9BF18C631B	128FEFACC51ED5790FF05D EC522FC1AF	4706581 6
7	901700000 009916	8988211000000 099161	581 3	87727 663	3AFA3506B8F4775FBFA80 8F56F59D801	3E07743AD99ECBAC8AC881 10CE0496A3	0054538 7
8	901700000 009917	8988211000000 099179	510 0	02694 322	C4AC6C875021BD7BA08C4 FD6CB14072D	C3809EF8E9301B83AF7F162 DA05D2360	9564087 9
9	901700000 009918	8988211000000 099187	134 5	30447 692	41F9509BC349E8BB6EABE 4DAB7EA1B16	0B2AACBBE4F0FF468B2E45 343DE21877	4583359 4
10	901700000 009919	8988211000000 099195	785 7	06634 950	EE3701E7BDF200C42207A 073891F8202	E7FED2FE3B8CAFC4E31974 107A25D8C5	9196964 2

Figure 115. Manufacturer SIM card parameters.

## Annex 3. Configuration of database

In the following there is a tutorial which explains step by step how to add a new user in the database by means Linux commands. In this example we are going to add into the database the user of 21412 network identified by its IMSI 214120000009919.

1. Open a new terminal in the PC.
2. Introduce the user and password.
3. Execute the following command: `mysql -u root -p` and introduce the password: `"mysqlpa33w0rd"`.

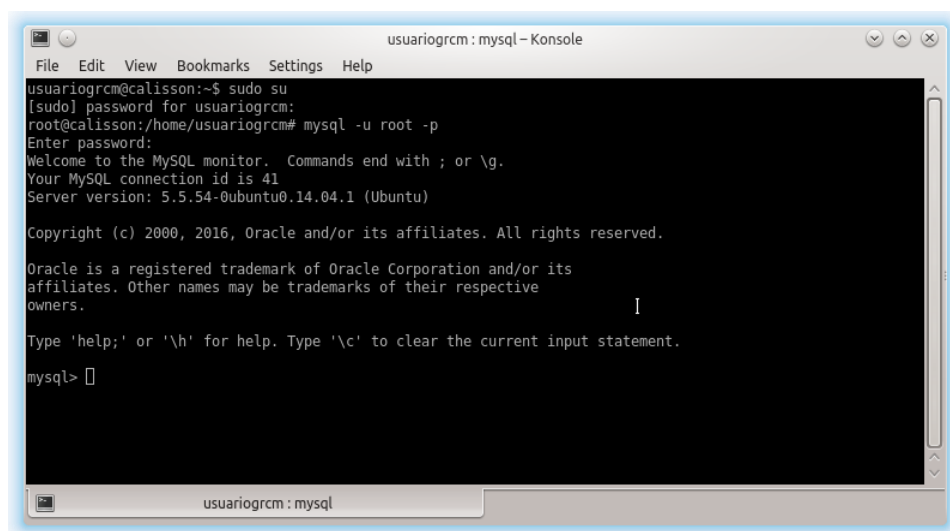


Figure 116. MySQL interface.

4. Introduce the command `use oai_db;` to enter in this database.

To add a new user in the database must be upgraded three tables in the OAI database:

- **mmeidentity**: It contains the record corresponding to my MME.
- **pdn**: It contains the association between a subscriber (user\_imsi) and an Access Point Name (APN) and its QoS parameters.
- **users**: It contains the information about the subscriber: IMSI, IMEI, key LTE K, SQN, operator key OP, QoS parameters, and the last known identity where the subscriber is registered.

5. The users table is the first one that we should modify introducing the following command:

```
INSERT INTO users (`imsi`, `msisdn`, `imei`, `imei_sv`, `ms_ps_status`,
`rau_tau_timer`, `ue_ambr_ul`, `ue_ambr_dl`, `access_restriction`,
`mme_cap`, `mmeidentity_idmmeidentity`, `key`, `RFSP-Index`, `urrrp_mme`,
`sqn`, `rand`, `OPc`) VALUES ('214910000009919', '34600000005', NULL,
NULL, 'NOT_PURGED', '120', '50000000', '100000000', '47', '0000000000',
'1', 'EE3701E7BDF200C42207A073891F8202', '1', '0', '',
0x00000000000000000000000000000000, '');
```

6. The pdn table is the second one that we should modify introducing the following command:

```
INSERT INTO pdn (`id`, `apn`, `pdn_type`, `pdn_ipv4`, `pdn_ipv6`,
`aggregate_ambr_ul`, `aggregate_ambr_dl`, `pgw_id`, `users_imsi`,
`qci`, `priority_level`, `pre_emp_cap`, `pre_emp_vul`, `LIPA-
Permissions`) VALUES ('5', 'oai2.ipv4', 'IPV4', '0.0.0.0',
'0:0:0:0:0:0:0:0', '50000000', '100000000', '3', '214910000009919', '7',
'7', 'DISABLED', 'ENABLED', 'LIPA-ONLY');
```

7. The mmeidentity table is the last one that we should modify introducing the following command:

```
INSERT INTO mmeidentity (`idmmeidentity`, `mmehost`, `mmerealm`, `UE-
reachability`) VALUES ('1', 'calisson.openair4G.eur', 'openair4G.eur',
'0');
```

To show the different tables through the terminal the command `show * "name_table"` should be introduced.

In Figures 98, 99 and 100 are shown the tables created via phpmyadmin.

## Annex 4. Launching the network

The procedure to run the entire network is the following. In this example we are working the 1 PC configuration (21491 network) and with the 2 platforms integrated.

1. From the 147.83.105.150 host make and ssh to remotely use the controller host (147.83.105.233).
2. Go to the /opt/empower/update directory and run the controller with administrator permissions

```
cd /opt/update/empower-runtime
```

```
sudo ./empower-runtime.py.
```

3. Then, run the EPC entity.

This entity is made of three blocks: HSS, MME and SPGW, so will be open 3 new terminals in the PC. It is recommended to start running the HSS block before than the others.

4. To run the HSS:

Go to the directory `/opt/openair-cn-new/SCRIPTS` executing the following command:

```
cd /opt/openair-cn-new/SCRIPTS
```

Finally, execute the next command to run the HSS:

```
./run_hss
```

5. To run MME:

Go to the directory `/opt/openair-cn-new/SCRIPTS` executing the following command:

```
cd /opt/openair-cn-new/SCRIPTS
```

Finally, execute the next command to run the HSS:

```
./run_mme
```

6. To run SPGW:

Go to the directory `/opt/openair-cn-new/SCRIPTS` executing the following command:

```
cd /opt/openair-cn-new/SCRIPTS
```

Finally, execute the next command to run the HSS:

```
./run_spgw
```

7. Finally, run the eNB:

Go to the directory `/opt/openairdebug/empower-openairinterface/cmake_targets/lte_build_oai/build` executing the following command:

```
cd /opt/openairdebug/empower-openairinterface/cmake_targets/lte_build_oai/build
```

Introduce the next command to run the eNB:

```
./lte-softmodem -O /opt/openairdebug/empower openairinterface/targets/PROJECTS/  
GENERIC-LTE-EPC/CONF/enb.band7.tm1.usrpb210.conf
```

Now, the entire network is running as shown the Figures 78 – 85.



## **Glossary**

ADM	Administrator PIN
APN	Access Point Name
CAPEX	Capital Expenditures
CC	Country Code
CN	Core Network
CPP	Click Packet Processor
eNB	evolved NodeBs
EPC	Evolved Packet Core
E-UTRAN	Evolved Terrestrial Radio Access Network
FDD	Frequency División Duplexing
FuN	Future Networks Unit
GRCM	Mobile Communications Research Group
GSM	Global System for Mobile Communications
GUMMEI	Globally Unique MME Identity
HSS	Home Subscriber Server
ICCID	Integrated Circuit Card Identifier
IMSI	International Mobile Subscriber Identity
LTE	Long Term Evolution
LTE-A	Long Term Evolution Advanced
LVAP	Light Virtual Access Point
LVNF	Light Virtual Network Function
MCC	Mobile Country Code
MIMO	Multiple Input Multiple Output
MME	Mobility Management Entity
MNC	Mobile Network Code
MVNO	Mobile Virtual Network Operator
NFV	Network Function Virtualization
OAI	OpenAirInterface
OPc	Operator key or code
OPEX	Operating Expenditures
OSA	OpenAirInterface Software Alliance
P-GW	PDN Gateway

PLMN Id	Public Land Mobile Network Identifier
QoS	Quality of Service
RAN	Radio Access Network
SDN	Software Defined Network
S-GW	Serving Gateway
TA	Tracking Area
TAI List	Tracking Area Identity List
TCP	Transmission Control Protocol
TDD	Time División Duplexing
UE	User Equipment
UMTS	Universal Mobile Telecommunications System
USRP	Universal Software Radio Peripheral
VBS	Virtual Base Station
VM	Virtual Machine
VNF	Virtualized Network Function

