Technische Universität Berlin

Institut für Telekommunikationssysteme
Fachgebiet Architektur der Vermittlungsknoten

Fakultät IV
Franklinstrasse 28-29
10587 Berlin
http://www.av.tu-berlin.de

Master Thesis

Design and Implementation of a flexible MME Load Balancing solution for Carrier Grade Virtual Mobile Core Networks

Rakesh Varudu

Matriculation Number: 0368410
08.02.2016

Supervised by
Prof. Dr. Thomas Magedanz

Assistant Supervisor
Prof. Dr. Axel Küpper

FOKUS Institute
Kaiserin-Augusta-Allee 31
10589 Berlin

This dissertation originated in cooperation with the Fraunhofer Institute for Open Communication Systems (FOKUS).

First of all I would like to thank Prof. Dr. Thomas Magedanz, Chair AV, TU Berlin for giving me the opportunity to carry out state of the art research in this field.

I would like to thank Prof. Dr. Axel Kupper for being the assistant supervisor for my thesis work.

A very special thanks to my industrial supervisor at Fraunhofer FOKUS Dr.-Ing. Marius-Iulian Corici for his excellent guidance throughout my thesis work and his help in making me understand the trends in 5G network. I would like to thank Jakub Kocur for his fantastic guidance in my day to day activities that made my job easier and lovable. I would also like to thank Eleonara Cau for sharing her experiences during her development activity at FOKUS.

Special thanks to Eichhorn, Fabian for his help in translating the abstract to German. A heartful thanks to the entire team of NGNI/Open5GCore and OpenSDNCore for their quick support whenever needed.

Furthermore, I would like to thank my parents Mr.Rambabu Varudu and Mrs.Sudha Rani Varudu, my brother Avinash Varudu, Varudu family, my paternal family, my maternal grand parents Mr. Poorna Chandra Rao Vadigina and Mrs.Dhana Lakshmi Vadigina, Vadigina family members Mr.Srinivas and Mr. Sai Ram Prasad for all their

continuous encouragement and love. Last but not the least, I thank my fellow students and EIT Digital Master School for this life changing experience.

I hereby declare that I wrote this thesis myself with the help of no more than the mentioned literature and auxiliary means.

Berlin, 08-02-2016

.....

(Signature [Rakesh Varudu])

Abstract

With every passing day there is a wide adoption of mobile connectivity by end-users. A variety of mobile applications are entering the market and the mobile subscribers need to have higher bandwidth and faster connectivity. While there is a growing list of requirements from the end-users, the network service providers need to equip themselves to address the needs of the users.

With legacy network infrastructure, in most of the cases, service providers overprovision their networks to address the peak hour demands. This leads to excessive costs as most of the time, the average load on network is much lesser than the peak load resulting in weak operation and maintenance of networks. Hence, they see a need to move to cloud based infrastructures that facilitate them to efficiently manage their networks. Service providers are moving to virtual infrastructures, where they can scale-in or scale-out their network resources on demand with the dynamically changing user needs. This thesis focuses on addressing the core network needs, especially towards the scaling of Mobile Management Entity (MME).

It is good to have a mechanism to scale MMEs in the core network, while ensuring that the end user doesn't see any changes in the core network. Other approaches like S1-Flex and random load balancing mechanisms exist to scale the core networks. However, the downside with these approaches is that they are not transparent to the user or eNodeB. Hence, a new load balancer on S1-MME interface level is needed to transparently scale the core network, while being as closely compliant as possible to the 3GPP standards. This thesis work focuses on design and implementation of an S1-MME load balancer, that can balance the incoming mobile subscriber traffic across multiple MMEs while still being able to maintain an affinity at transaction level.

For the purpose of integration, Open5GCore toolkit, developed by Fraunhofer FOKUS that provides all access network and core network elements was used. The implemented S1-MME load balancer was verified for its compliance with 3GPP standard and was tested against all the requirements, and performance was evaluated which shows that the designed and implented S1-MME load balaner works as expected by successfully processing S1 Setup, S1AP procedures like Attach, Detach and Handovers. It was able to redirect all the MME signalling messages for all the above stated S1AP procedures to

the core network with a delay latency overhead of around 10% during attach operations. Eventually, possible future enhancements to S1-MME load balancer were discussed.

Zusammenfassung

Da die meisten Leuten an der TU deutsch als Muttersprache haben, empfiehlt es sich, das Abstract zusätzlich auch in deutsch zu schreiben. Man kann es auch nur auf deutsch schreiben und anschließend einem Englisch-Muttersprachler zur Übersetzung geben.

Jeden tag verwenden mehr und mehr Nutzer mobile Endgeräte. Eine Vielfalt von mobilen Applikationen strömt auf den Markt und Mobilfunkkunden benötigen schnellere Verbindungen mit höheren Datenraten. In Anbetracht der steigenden Anforderungen der Endnutzer, müssen Netzwerkbetreiber ihre Systeme aufrüsten.

Die meisten legacy Netzwerke werden von ihren Betreibern überprovisioniert, um Lastspitzen bedienen zu können. Dies führt zu exzessiven Kosten, da die durchschnittliche Auslastung der Netzwerke den Großteil der Zeit deutlich geringer ist. Daher wenden diese Betreiber sich cloud basierten Infrastrukturen, welche eine effizientere Steuerung der Netzwerkresourcen ermöglichen. Mobilfunkanbieter stellen ihre Netzwerke auf virtuelle Infrastrukturen um, mit denen sie ihr Netzwerk dynamisch den aktuellen Lastanforderungen anpassen können. Diese Arbeit konzentriert sich auf die Anforderungen des Kernnetzes.

Es ist von Vorteil eine Mechanismus zur Skalierung von MMEs im Kernnetz zu haben, der für die Endnutzer nicht detektierbar ist. Bereits bestehende Ansätze, wie S1-Flex und random load balancing, können das Kernnetz skalieren. Diese sind jedoch nicht transparent für den Endnutzer oder die eNodeB. Daher ist ein Lastausgleichsverfahren auf der S1-MME Protokollebene notwending, welcher den Lastausgleich transparent und dem 3GPP Standard so konform wie möglich realisiert. Diese Arbeit befasst sich mit dem Design und der Implementierung eines S1-MME Loadbalancers, welcher eingehenden Mobilfunkverkehr über mehrere MMEs verteilen kann ohne die Affinität auf der operativen Ebene zu verlieren.

Contents

List of Figures xv

List of Tables xvii

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Objective . . . 2
- 1.3 Scope . . . 4
- 1.4 Methodology . . . 6
- 1.5 Outline . . . 8

2 State of the Art 9
- 2.1 3GPP EPS Architecture . . . 9
- 2.2 S1 Setup procedure . . . 11
  - 2.2.1 S1 Setup Request . . . 12
  - 2.2.2 S1 Setup Request Response . . . 12
- 2.3 S1AP Attach . . . 12
- 2.4 S1AP Detach . . . 19
- 2.5 S1AP Handover procedure . . . 22
  - 2.5.1 Overview of S1 Handovers . . . 22
  - 2.5.2 S1- Handover message flow . . . 23
- 2.6 S1-Flex . . . 29
- 2.7 Load Balancing Techniques . . . 31
- 2.8 Conclusion . . . 33

3 Requirements 35
- 3.1 Problem Statement . . . 35
  - 3.1.1 Focus Areas . . . 36
    - 3.1.1.1 Good Network coverage . . . 36
    - 3.1.1.2 Uninterrupted service . . . 36
    - 3.1.1.3 Quick response for application needs . . . 37
  - 3.1.2 Existing approach and its problems . . . 37
- 3.2 Functional Requirements . . . 38
  - 3.2.1 Message Flow Handling . . . 38
  - 3.2.2 Transparency . . . 39
  - 3.2.3 Load Balancer Components . . . 39

3.2.3.1 Core Message Handling . . . 39
3.2.3.2 Balancer Policing . . . 41
3.2.3.3 External Management support . . . 41
3.2.4 Configuration Management . . . 42
3.2.5 3GPP Standard Compliance . . . 42
3.3 Non-Functional Requirements . . . 43
3.3.1 Modularity . . . 43
3.3.2 Performance . . . 43
3.3.3 Interoperability . . . 44
3.3.4 Portability . . . 44
3.3.5 Usability . . . 45
3.4 Conclusion . . . 45

4 Design and Specification **47**
4.1 Architectural Overview of EPC with S1-MME LB . . . 47
4.2 S1-MME Load Balancer Design . . . 48
4.3 Interface Specifications . . . 50
4.3.1 S1 Interface Support . . . 50
4.3.2 Deviations from 3GPP standard . . . 51
4.4 S1AP Message Handling . . . 53
4.4.1 S1AP Message handling from eNodeB to MME direction . . . 53
4.4.2 S1AP Message handling from MME to eNodeB direction . . . 54
4.5 S1AP procedures . . . 55
4.5.1 S1AP Setup . . . 55
4.5.1.1 S1AP Setup Request . . . 55
4.5.1.2 S1AP Setup Response . . . 56
4.5.2 S1AP Attach . . . 56
4.5.3 S1AP Detach . . . 59
4.5.4 S1AP Handover Procedure . . . 60
4.6 Conclusion . . . 63

5 Implementation **65**
5.1 Environment . . . 65
5.2 Overview of Project Structure . . . 66
5.3 S1-MME Load Balancer components and Operations . . . 68
5.3.1 S1AP Load Balancing . . . 69
5.3.2 Session Management . . . 69
5.3.3 Balancer Policing . . . 70
5.3.4 Configuration Management . . . 72
5.3.5 Console Management . . . 73
5.3.6 Module Management . . . 74
5.4 Virtualization . . . 75
5.5 Conclusion . . . 76

6 Evaluation **77**
6.1 Test Environment . . . 77
6.2 Test Scenarios . . . 80
6.2.1 Test Cases . . . 80
6.2.2 Test Execution Procedures . . . 81
6.3 Compliance with 3GPP Standards . . . 83
6.3.1 S1AP Attach and Detach Procedures . . . 83
6.3.2 S1AP Handover Procedure: . . . 83
6.3.3 Observations from the wireshark traces: . . . 84
6.4 Performance Measurements: . . . 85
6.5 Delay analysis for Attach operations: . . . 85
6.5.1 Delay analysis for Detach operations: . . . 87
6.6 Conclusion: . . . 88

7 Conclusion **89**
7.1 Summary . . . 89
7.2 Dissemination . . . 90
7.3 Problems encountered . . . 91
7.4 Outlook . . . 91

List of Acronyms **93**

Bibliography **97**

List of Figures

1.1 Effect on growth in mobile traffic due to smart devices 1
1.2 Control plane and data plane split in EPS and E-UTRAN 2
1.3 Load Balancer enabling RAN Sharing and Network slicing 3
1.4 Functional Block design 4
1.5 S1-MME Load Balancer Overview 5
1.6 Methodology Overview 7

2.1 Mobile Architecture according to 3GPP standards 9
2.2 S1 Setup procedure 11
2.3 Summary of S1AP Attach procedure-part 1 13
2.4 Summary of S1AP Attach procedure-part 2 14
2.5 Initial UE message 15
2.6 Initial Context Setup Request and Response 17
2.7 UE Initiated Detach procedure 20
2.8 Summary of S1AP Detachment procedure 21
2.9 Overview of S1 Handover procedure 22
2.10 Intra-LTE Handover message flow using S1 interface-part 1 24
2.11 Intra-LTE Handover message flow using S1 interface-part 2 25
2.12 Handover Required message 26
2.13 Handover Request Message 26
2.14 Handover Command Message 27
2.15 eNodeB status transfer 27
2.16 MME status transfer 28
2.17 Handover Notify 28
2.18 S1-Flex Mechanism 29
2.19 S1-Flex handling MME Failover 30
2.20 Load Balancing application traffic between servers in the cloud 31

4.1 Load Balancer Integration in EPC 47
4.2 S1-MME Load Balancer Design 48
4.3 S1-MME Protocol Stack 49
4.4 S1-MME LB Interface Support 50
4.5 S1AP Message Handling from eNodeB to MME 53
4.6 S1AP Message Handling from MME to eNodeB 54
4.7 S1AP Setup Request 55
4.8 S1AP Setup Response 56

4.9 LTE Attachment with S1-MME LB part 1 57
4.10 LTE Attachment with S1-MME LB part 2 59
4.11 LTE Detach Procedure 60
4.12 S1AP LTE Handover Procedure (Part 1) 62
4.13 S1AP LTE Handover Procedure (Part 2) 63

5.1 Project Structure 67
5.2 Basic components of s1ap-loadbalancer module 68
5.3 S1-MME Load Balancer Configuration 72
5.4 Console support for S1-MME Load Balancer 74
5.5 S1-MME Load Balancer running VM WorkStation 75

6.1 S1-MME Testbed environment 79
6.2 S1-MME Testbed Network Configuration 80
6.3 Traffic capture of Attach and Detach operations on S1-MME interface . . 83
6.4 Traffic capture of Handover operations on S1-MME interface 84
6.5 Delay analysis for Attach operations 85
6.6 Delay analysis for Detach operations 87

List of Tables

3.1 S1AP Procedures Supported 39
3.2 S1AP Messages Supported 40
3.3 Management Operations Supported 41

4.1 Deviations from 3GPP standard 52

1 Introduction

1.1 Motivation

Since the last decade, we are experiencing an immense growth of the mobile traffic. This increase in mobile traffic is due to increase in usage of smart devices. The number of mobile devices connected to IP networks exceeded the population of the world in the year 2014 and is expected to grow beyond 3 times by end of 2019. The main reason for this are due to changing communication paradigms like increase in Internet usage on mobiles, cloud storage and computing, Internet of things, always-on connectivity. With the growing availability of smartphones at a lower cost, their users tend to run different-bandwidth hungry applications like playing high resolution videos, graphic-rich multimedia online games, interactive audio and video, high quality audio streaming. It is expected that the global mobile traffic will grow 11-fold at a compounded annual growth of 61% from 2014 to 2019 [CIS14].

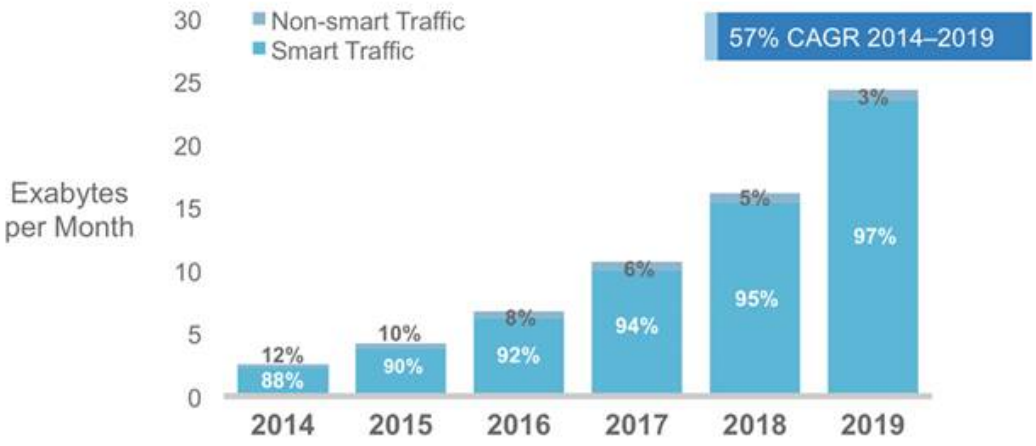


Figure 1.1: Effect on growth in mobile traffic due to smart devices

Figure 1.1 depicts the impact of growth in global mobile traffic due to increase in usage of smart mobile devices. Traffic due to smart devices is expected to reach 97% of the total global mobile traffic by 2019 from its current volume of 88%. This is a reduction of non-smart mobile traffic from 12% to 3% in a span of 5 years starting from 2014. The trend shows that there is 57% compounded annual growth in global mobile traffic between 2014-2019.

All these changes are fueling the need for a more scalable and efficient mobile network architecture where focus has been increasing on adopting emerging software trends like cloud computing, software defined networking (SDN)[Ope12] and network function virtualization (NFV)[ETS] into mobile architectures in order to address the growing needs of users, while minimizing the expenses for the mobile network service providers.

The users' requirements keep changing very dynamically and a general observation in mobile networks is that the usual average network load is much lesser than the peak traffic load. Earlier due to this, mobile network operators have to over-provision their networks. With the changing paradigm in mobile network architectures, it is good to have a mechanism where based on the traffic needs, the number of virtualized MME elements could be either scaled out or scaled in instead of reserving extra fixed hardware resources to handle peak load that rarely appears in the network [Fuj14]. This pushes the need for the development of scalable and flexible core networks within virtualized environment that adapts to the varying situation in the system due to changing subscriber requirements.

1.2 Objective

According to 3rd Generation Partnership Project (3GPP), a body that works with telecommunication standard development organisations to standardise telecommunication standards, it has introduced a new architecture for mobile networks that has EPC which has two important parts, i.e., Evolved Packet System (EPS) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access. This architecture supports to allow the split between control plane traffic and data plane traffic as described in the figure 1.2.

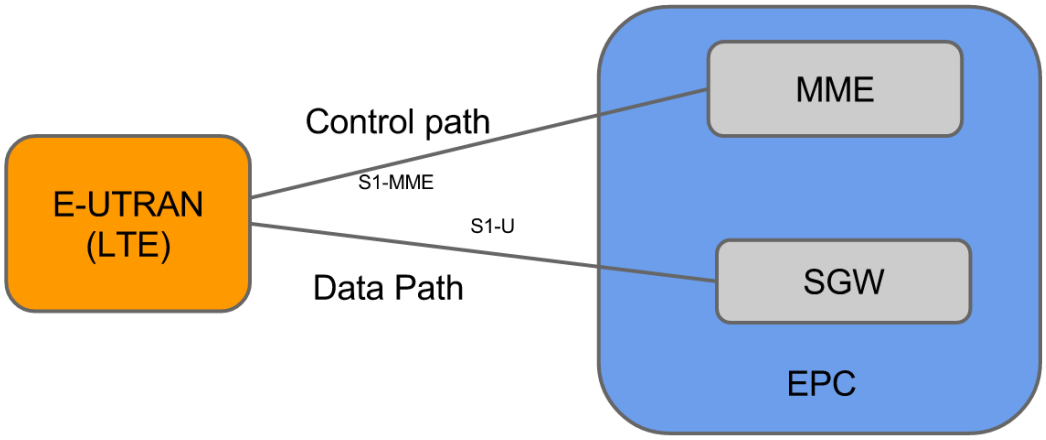


Figure 1.2: Control plane and data plane split in EPS and E-UTRAN

The main objective of the work is to enable communication with the control entity

MME distributed among multiple virtual machines. This allows to have a dedicated core network for different subscriber needs like machine-to-machine services (M2M), general subscriber usage, enterprise networks.
Proposed solution will also help to solve the following challenges in the current mobile networks as shown in figure 1.3.

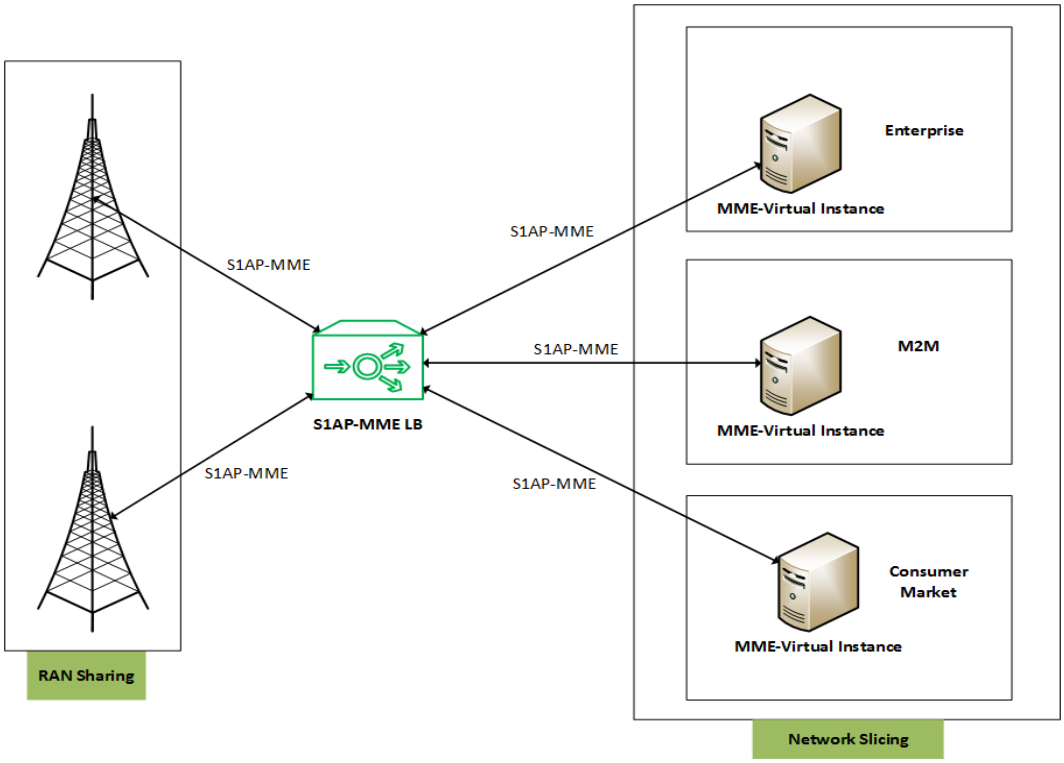


Figure 1.3: Load Balancer enabling RAN Sharing and Network slicing

1. **RAN Sharing:** It is challenging for operators to handle the growing business needs, obtaining spectrum resources, reducing the operating costs and reaching the market on time. This is pushing the network operators to collaborate with one another by sharing their access network resources thus mutually benefiting from each other. RAN sharing becomes more meaningful when the operators are able to split their core traffic between different virtualized MME control entities running as software elements in cloud.

2. **Deployment of NFV:** Service providers are moving towards cloud-based infrastructure, software based mobile networks to provide more robust and cost-effective services. MMEs can be deployed as virtual elements following ETSI NFV standard.

3. **On demand scaling:** The split in MME control plane functionality across multiple virtual machines in cloud allows to scale in or scale out the virtual MME

instances based on the traffic needs without making any changes in the access network because of the transparency provided by the S1-MME Load balancing entity.

1.3 Scope

The scope of the work includes implementing a S1-MME load balancer that resides between eNodeB and virtualized MME instances as shown in figure 1.4. This allows the traffic coming from eNodeB to be distributed across the instances of MME.

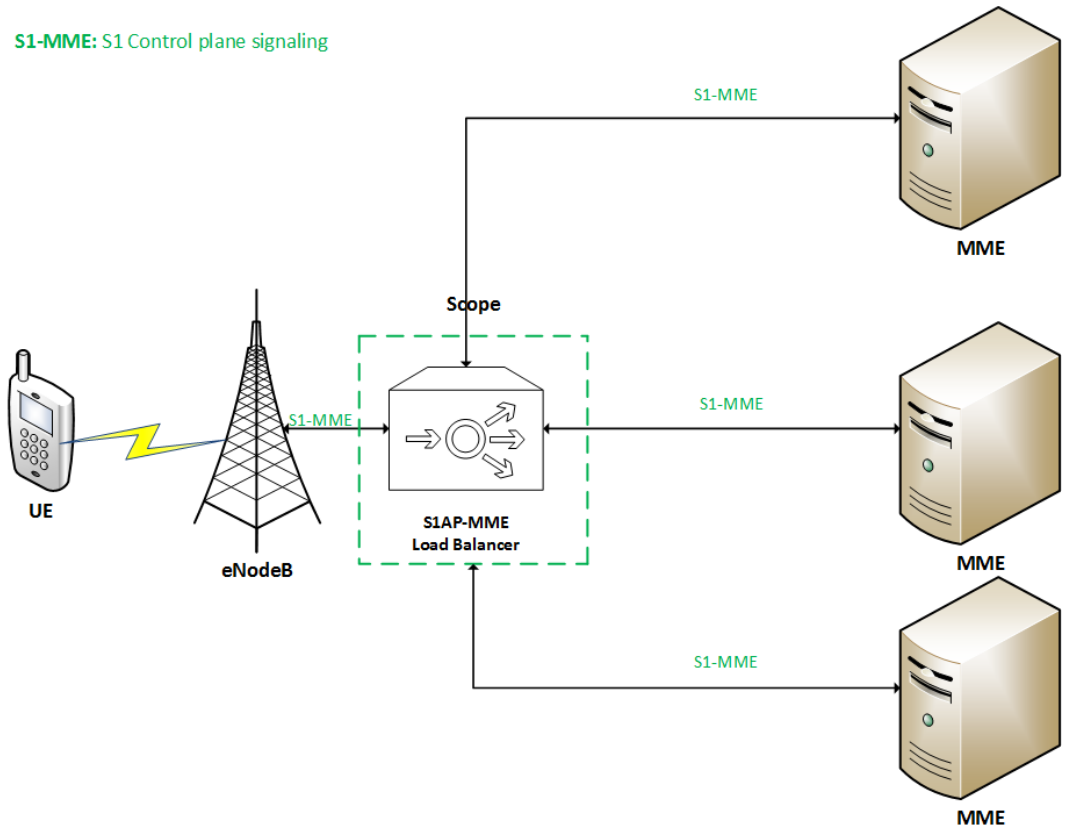


Figure 1.4: Functional Block design

Considering the need for changes in mobile networks, Fraunhofer FOKUS has developed a platform named Open5GCore. According to the latest developments in the mobile network the Evolved Packet Core functionality is being moved towards the cloud, leveraging

the NFV and generic implementations of software on a single hardware, reducing the costs. The MME is one of the examples, where it's functionality is placed as a software in the cloud, which is scaled out/in based on demand.

However, this novel approach of transforming the specialized EPC hardware elements to general purpose software elements in cloud introduces new challenges that have to be addressed before deploying such a solution in a commercial network. One of the major challenges is to keep the scaling transparent to the other components. As of now, an eNodeB connects to one of the MMEs by an S1 interface in the cloud as shown in figure 1.5. As the MME instances can be scaled out/in, it is good to have a S1-MME load balancer (LB) that dynamically decides which MME the subscriber connects to. A LB has to be implemented as a separate software component that can be run in a cloudified environment as a front end. It resides in front of multiple MMEs in a data center(s) as shown in figure 1.5. The Load Balancer should be able to receive external management commands for reacting to the changes in the network, like adding/removing newly created/deleted MME instances.

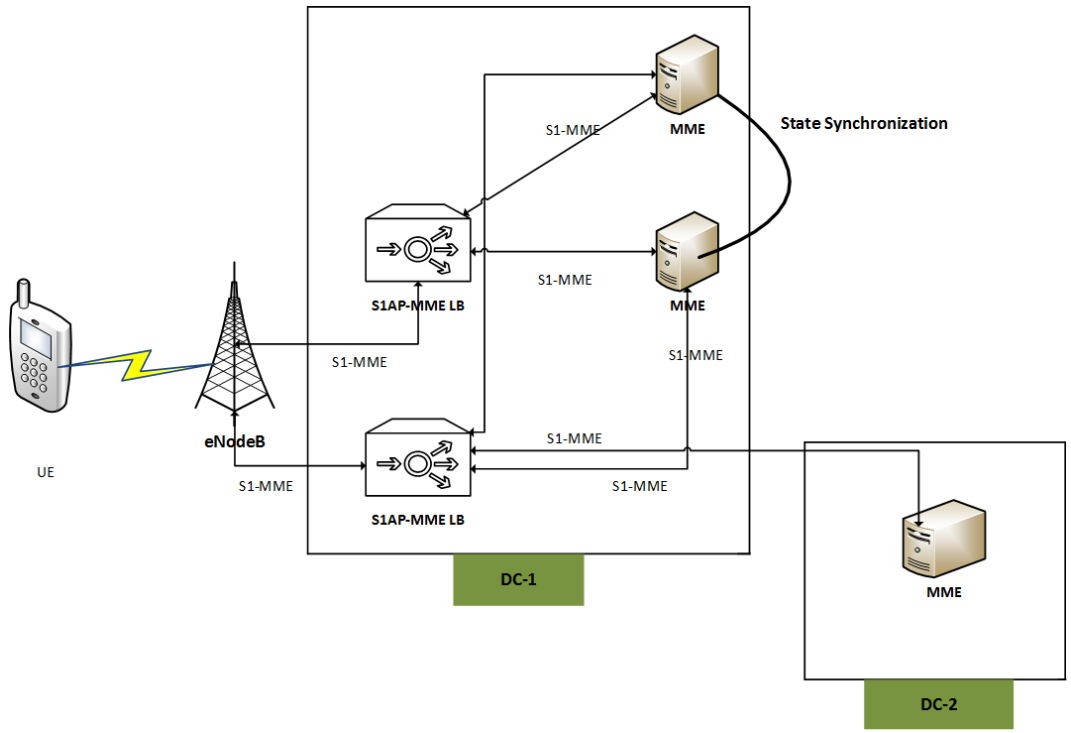


Figure 1.5: S1-MME Load Balancer Overview

The aim of the thesis is to design and implement a prototype of a S1-MME load balancer, as described in the previous paragraph, evaluate the performance and accuracy of the prototype on a running testbed. For the practical part of the thesis, the Open5GCore

platform developed by Fraunhofer FOKUS will be used as an experimentation environment, enabling functional end-to-end system testing, and in which required evaluation will be performed.

This graceful transition from one MME to the other MME requires synchronization of the state information as seen in figure 1.5, which can be realized by a new type of memory shared between various network element instances. Care should be taken such that this implementation changes are transparent to eNodeB and EPC components, requiring no additional changes to the standardized interfaces between those elements. The result is a transparent network where eNodeBs don't see any scaling effects in MME. Multiple LBs should be able to run simultaneously and eNodeB can be connected to any of them. Also, the LB should be able to balance between different MME software instances running in different data centers.

Due to limited time constraints and considering the complexity of the system to be built, the scope of the thesis is restricted to the implementation of an S1-MME load balancer. The rules for traffic splitting in S1-MME load balancer, state sync between MMEs are taken as a separate work.

1.4 Methodology

For the given problem statement, a comprehensive study will be done on the state-of-the-art that can possibly solve the problem. The state-of-the-art includes latest existing approaches towards the problem, current research activities that might help in addressing the problem. Any gaps in solving the problem are identified. A detailed list of requirements are brainstormed keeping in view of the changing users expectations. The requirements are defined taking into view the 3GPP standards for the mobile network architectures.

Once, all the requirements are captured, the S1-MME load balancer is designed. The design of the load balancer is done in such a way that it as closely compliant as possible to the 3GPP standard. Any limitations in standard compliance should at least fit in the context of Open5GCore test-bed platforms at Fraunhofer FOKUS. An implementation is done based on the the design choices made. Then the implemented load balancer is evaluated against the requirements set. Any issues identified will be addressed. Depending on the seriousness of the gaps identified, changes are done either at implementation or design level. If the issue is at very core of the design, the design is changed to suit the requirements and the implementation follows any design changes as explained in figure 1.6.

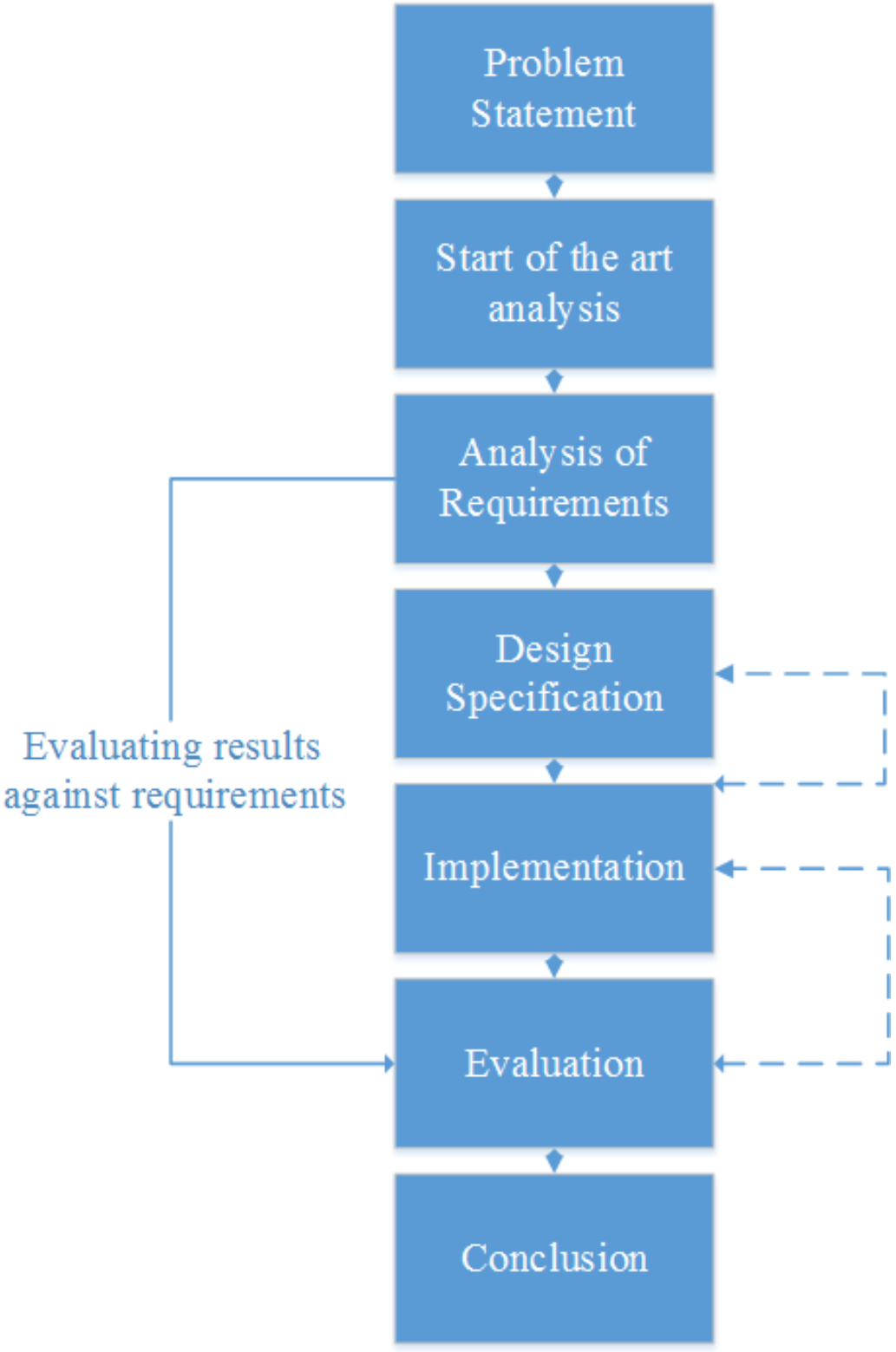


Figure 1.6: Methodology Overview

By following the iterative process as described in figure 1.6, a fully functional S1-MME load balancer is developed. Eventually, after the evaluation process, the merits and demerits of the work, lessons learnt during the journey, and an outlook of the possible future work are discussed.

1.5 Outline

This section gives a brief introduction to the main chapters of the work.

This thesis is separated into 7 chapters.

Chapter 2 starts with an overview followed by an introduction to SDN and NFV, their importance in 5G networks, followed by description of the S1AP attachment procedure and the state of the art for network redundancy and load-balancing using S1-Flex, followed by their analysis in the context of previously described technologies.

Chapter 3 discusses the problem statement and analyses the requirements for the S1-MME load balancer to address the problem.

Chapter 4 discusses the design and specification of the S1-MME load balancer. It describes the architectural overview of the load balancer, followed by its design.

Chapter 5 describes the environment and technologies used for the prototype. The implementation details of Load balancer components, it's operations and configuration management are explained.

Chapter 6 evaluates the S1-MME load balancer prototype implementation against the requirements.

Chapter 7 describes the problems encountered during the thesis work, learnings during this phase, any solutions to overcome the problems encountered. It also summarizes the results and includes possible future work.

2 State of the Art

This chapter gives an overview of the Evolved Packet System (EPS) architecture, describes the functionality of each of its elements, with the focus on the communication between these elements, especially between eNodeB and MME. The communication between eNodeB and MME takes place on s1 interface defined by 3GPP standard. S1AP procedures like S1AP Setup, S1AP Attach, S1AP Detach, S1AP Handover are explained. Finally, other existing approaches and their shortcoming to solve the problems in core network scaling are discussed briefly.

2.1 3GPP EPS Architecture

Two primary components of 4G network is Evolved UMTS terrestrial Radio Access Network (LTE) and Evolved Packet Core (EPC), as shown in figure 2.1.

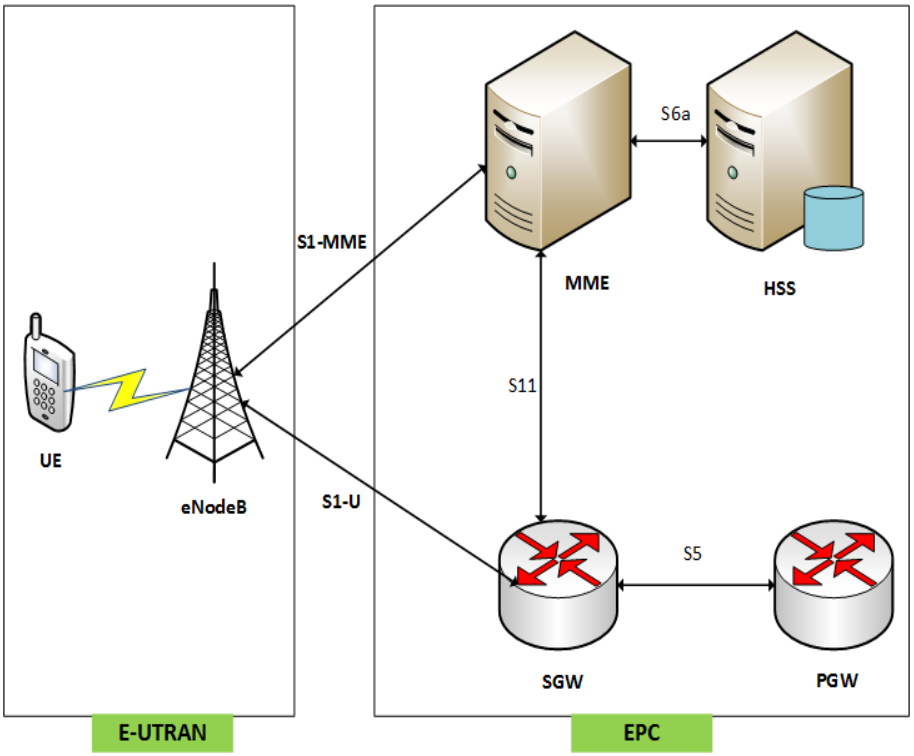


Figure 2.1: Mobile Architecture according to 3GPP standards

The User Equipment (UE) in an LTE network connect with the E-UTRAN which is composed of a number of eNodeBs [STK11]. The eNodeB in a LTE network, combines the function of a RNC/BSC and NodeB/BTS in a typical 3G/2G network.

The eNodeBs are responsible for all of the radio related functions of LTE such as radio resource management including resource scheduling, mobility control, paging, air interface security and MME selection.

EPC was introduced by 3GPP in Release 8 of the 4G standard[Fir]. The user data and the signaling are also separated by using EPC to make the scaling independent. EPC is the main part of an LTE network which comprises of MME (Mobility Management Entity), S-GW (Serving Gateway) and Packet Data Network or PDN Gateway (P-GW) along with the HSS (Home Subscriber Server). The EPC connects with the E- UTRAN using an S1 interface.

The E-UTRAN consists of multiple eNodeBs which provide E-UTRAN user plane and control plane protocol terminations towards the user equipment (UE). The eNodeBs are optionally interconnected with each other by means of X2 interface [Fir]. The eNodeBs are connected with the EPC by using different types of S1 interfaces. All the eNodeBs are connected individually to both the MME and the S-GW through an S1 interface. The control part of the S1 interface or the S1-MME interface lies between the eNodeB and the MME and is controlled by the S1 Application Protocol (S1AP). The user part of the S1 interface or S1-U interface lies between the eNodeB and the S-GW and uses GPRS Tunneling Protocol (GTP). S1AP facilitates the exchange of signaling messages between the eNodeB and the MME while GTP helps carry user traffic such as email and web data between the eNodeB and the S-GW.

The MME handles control plane signaling like Non-Access Stratum (NAS) signaling [Mul09]. The MME tracks and maintains the current location of all UEs that have registered to it via an Attach procedure. This allows the MME to page an UE [Fir]. The MME interacts with the 2G/3G overlay legacy MSC so that Circuit-Switched Fallback (CSFB) may be realized. MME also coordinates the establishment of EPS bearers, which help carry user traffic between the UE and the P-GW. MME, on the other hand, accesses the subscriber database which is stored in the HSS for validating the user. LTE uses the Diameter protocol on the S6a interface.

HSS stores the subscription profile of the subscriber and records the current location of the UE. The HSS typically performs user authentication and authorization [Fir].

The S-GW and P-GW handles the user plane data. S-GW has GTP-U tunnels with the eNodeB and the P-GW to carry user traffic [Mul09]. A UE has only one serving gateway at any instance. The S-GW establishes GTP-U tunnels based on the instructions of the MME received via the S11 interface. The S-GW takes care of user plane packets by

forwarding them between the eNodeB and the P-GW via the S1-U and S5 interfaces, respectively. S-GW takes care of user's IP packets routing and forwarding, servers as an anchor point for handovers, transport level packet marking [Fir] [3GP13b].

In LTE, IP address allocation is performed by the PDN Gateway (P-GW). The P-GW gives an IPv4 address, or an IPv6 address, or both IPv4 and IPv6 addresses to the UE [Fir]. The P-GW is connected to the S-GW using the S5 interface which uses GTP-C for control plane signaling[3GP13f] and GTP-U for the user plane [3GP13a] [Mul09]. The P-GW is used to connect to external Packet Data Networks identified by unique Access Point Names (APNs). The interface between the S-GW and the P-GW is called an S5 interface when both are located in a home network. When the UE roams into a visited network, the user might connect from the visited S-GW to the home P-GW through an S8 interface [GSM13].

2.2 S1 Setup procedure

A S1 setup message is triggered by the eNodeB so that the MME takes it into service[RK09] as described in figure 2.2. S1 setup procedure allows the eNodeB and MME to exchange application level data required to interoperate on the S1 interface. On completion of the S1AP procedure, a connection is established between the eNodeB and the MME and it also initializes any existing UE related contexts on both eNodeB and the MME. It clears the existing configuration data if any exists and replaces it with the new application information received in the setup message[3GP13e].

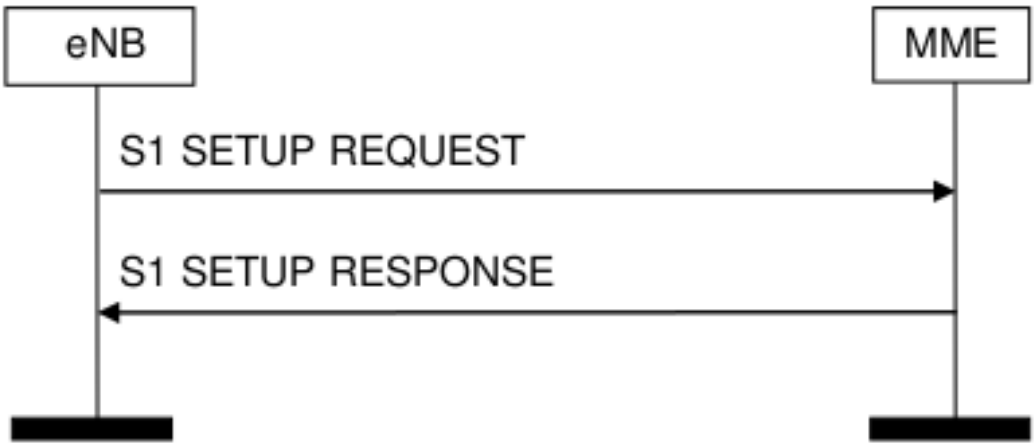


Figure 2.2: S1 Setup procedure

2.2.1 S1 Setup Request

It is eNodeB that initiates the S1 setup procedure by sending an S1 setup request message to the MME as shown in figure 2.2. It shares all the eNodeB configuration details to the MME in this message [3GP13e]. An eNodeB is uniquely identified by a global eNodeB Id, which is a combination of Mobile Country Code (MCC), Mobile Network Code (MNC) and macro ENB-ID. Furthermore, the message also has details about the configured Tracking Area Code (TAC) of the eNodeB along with the ip address of the eNodeB. With these details, a MME identifies unambiguously each eNodeB that is taken into the service [RK09].

2.2.2 S1 Setup Request Response

In response to the S1 setup request sent by the eNodeB, MME sends a s1 setup response message back to the eNodeB identified by it's ip address as shown in figure 2.2. MME in it's response message informs the Globally Unique Mobile Management Entity Identifier (GUMMEI) information for the eNodeB so that here afterwards, eNodeB can identify itself with this information[RK09]. GUMMEI contains the MME code and the MME group ID information in it.

The data retrieved by each node will be stored by them for the duration of session. On successful s1 setup response by the MME, the s1 layer is ready for communication between eNodeB and MME for other control messages to be exchanged[3GP13e].

eNodeB can share optional field eNodeB name while MME can share MME name in the corresponding messages.

2.3 S1AP Attach

A mobile subscriber that needs services from a network should first register with the it. After the successful completion of the registration, a network allows the subscriber to access its services based on the services the subscriber has paid for. The access to these services is controlled the HSS by informing the MME about the QoS that it has to offer to a specific subscriber.

A mobile subscriber that needs services from a network should first register with the it. After the successful completion of the registration, a network allows the subscriber to allow its services based on the services the subscriber has paid for. The access to these services is controlled the HSS by informing the MME about the QoS that it has to offer to a specific subscriber.

The summary of the S1AP Attach procedure can be seen in figure 2.3. It shows an overview of the involved elements and the messages exchanged between these elements.

GUTI is used to identify UE for control plane communication between UE and MME. A mapping is established between GUTI and IMSI of the UE. On S1 control plane, a connected UE is identified by a pair of S1AP IDs, i.e., eNB-UE-S1AP-ID and MME-UE-S1AP-ID. The details about these messages are discussed elaborately in further sections.

Usually, a Cell Radio Network Temporary Identifier (C-RNTI) is used to identify each UE connection on radio side. eNodeB separates the control plane and data plane packets from the UE. RRC protocol is used for communication on the radio side between UE and the eNodeB

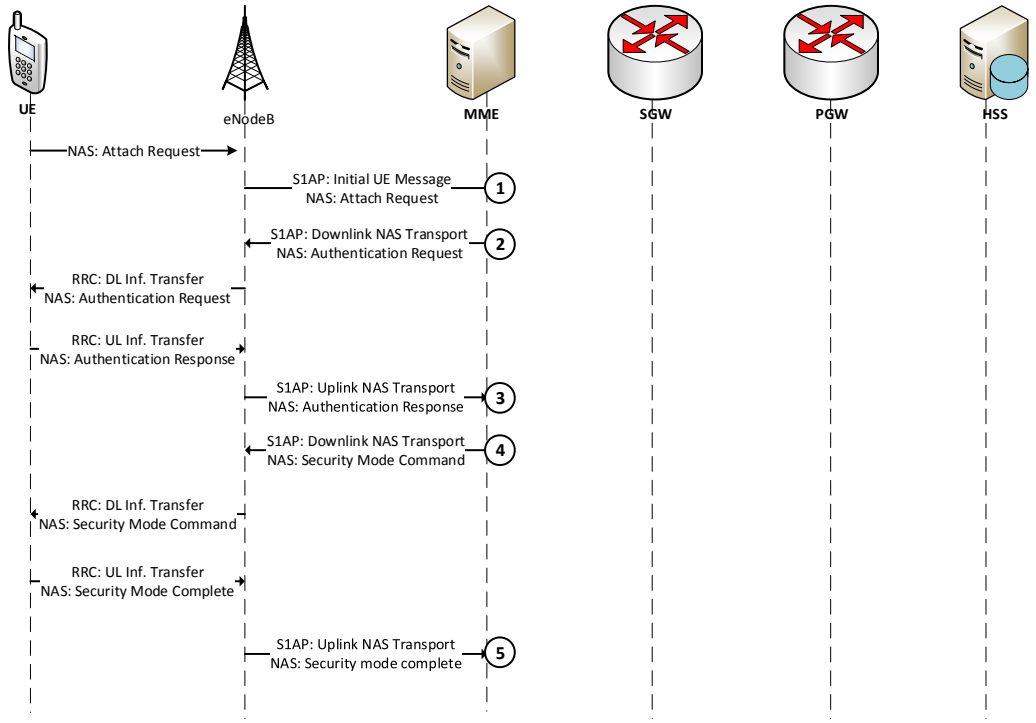


Figure 2.3: Summary of S1AP Attach procedure-part 1

Several NAS Uplink and Downlink messages are exchanged between eNodeB and MME in order to realize the S1AP UE Attach procedure.

As shown in figure 2.4, MME checks with the HSS to verify if the user is allowed to access the services provided by the network. Once it knows that it is allowed, it creates the GTP sessions between MME and SGW and in turns triggers further requests to establish GTP sessions between SGW and PGW. This sessions carry the data traffic coming from the users to the external network through the gateway. A brief summary

of this part of the S1AP Attach procedure is described in figure 2.4.

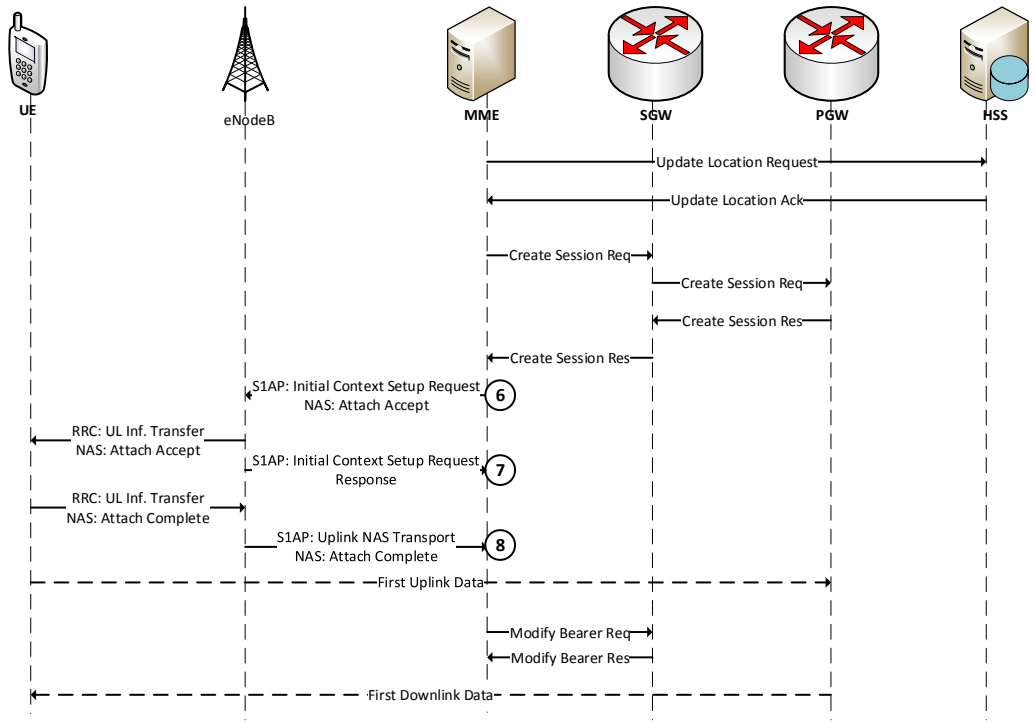


Figure 2.4: Summary of S1AP Attach procedure-part 2

eNodeB sends the control plane signalling information to the MME, while the data traffic to the S-GW. The control plane and user plane tunnels are identified by F-TEID and transport layer addresses (ipv4/ipv6) of the involved end points.

As shown in 2.5, an initial attach message is triggered when user switches on their handset, while in case of data cards or USB sticks that are used to connect to mobile networks, a connect button initiates the attach. UE in its attach message indicates that it is of type "EPS attach" in the EPS attach attach type IE. The UE will include old Globally Unique Temporary ID (GUTI) if it is earlier already connected with the network, as this simplifies the registration process with the core network. If the UE is not already connected to the network, it will send International Mobile Subscriber Identity (IMSI) information to the core network. The UE requests a PDN connectivity by sending a PDN CONNECTIVITY REQUEST message in a EPS Session Management (ESM) message container element along with the ATTACH REQUEST message. The UE shall integrity protect the message if a valid NAS security context exists [3GP13c]. The details of the NAS Security context are excluded from discussion, as they don't fall in

the scope of the project. A default S1-U bearer is immediately created for the subscriber even before the attach procedure is complete. This reduces the delay in accessing the network. Along with a S1-U bearer, PDP connection is established in core network so that payload packets can be exchanged back and forth between UE and the PDN-GW [RK09].

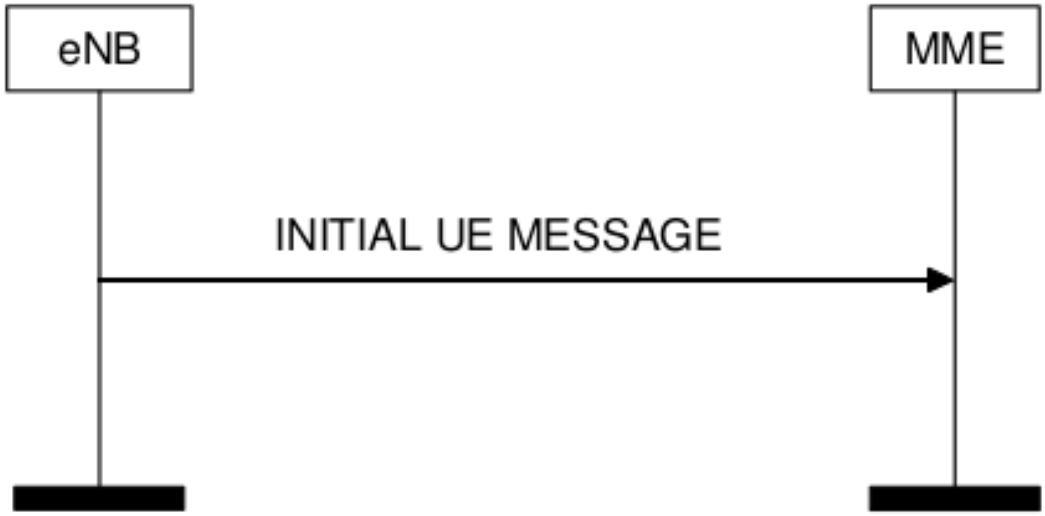


Figure 2.5: Initial UE message

An UE initiates an attach message by sending a NAS attach request message to the MME via eNodeB. The contents of a NAS message are not interpreted by the eNodeB, but the message is just forwarded by it to the MME. It is the Universal Subscriber Identity Mobile (USIM) card in the UE that sends the NAS attach request. This includes the attach type, PDCP connection request and the UE identity. If USIM card is already registered with a 4G network, it sends the GUTI information stored in it's memory, if not it will send the IMSI identifier in the request message. On receiving the NAS message transmitted on a Radio Resource Control (RRC) connection by the eNodeB, it invokes a NAS transport message called the initial UE message. eNodeB in its message includes the eNodeB S1AP ID that uniquely identifies this ue session on the eNodeB. The initial message contains current location of the subscriber in the form of tracking area ID and Evolved UTRAN (EUTRAN) Cell Global Identity (CGI) to the MME [RK09] [3GP13e].

The procedure code information in the message indicates the type of the NAS message. An initial UE message will have the procedure code id-initialUEMessage. For this subscriber, the eNodeB uses the same eNodeB S1AP ID throughout its communication with the MME.

The initial UE message contains current location of the subscriber in the form of tracking

area ID and EUTRAN CGI to the MME.

On receiving an ATTACH REQUEST, the network triggers EMM procedures like authentication, identification and security mode procedures based on values like IMSI, GUTI and Key Set Identifier (KSI) in the ATTACH REQUEST message[3GP13c]. These procedures are not discussed in detail considering considering that they don't fall into the scope of the thesis.

MME on reception of the initial UE message extracts the UE information like EUTRAN CGI, Tracking area ID and contacts the HSS using the DIAMETER update location procedure. This allows the HSS to store the newly detected UE information in its database. HSS replies MME whether the subscriber is allowed to use the services offered by the network, together with the QoS parameters, in case access is provided to the subscriber. In the message, HSS is identified by Origin Host and Origin Realm, while the receiver i.e., the MME is identified by destination Host and destination Realm. The field DIAMETER result code informs whether the diameter update is successful or not. The message from HSS to MME contains a field called Application-ID which indicates that this is a message on the S6a interface between the HSS and the MME. HSS also informs the maximum allowed uplink and downlink bit rates allowed for uplink and downlink bearers that carry the traffic. It also contains the QoS parameters indicated by QoS Class Identifier (QCI) value. HSS also assigns an IP address to the UE, while a dynamic IP address will be assigned by the PDN-GW later. HSS informs the PDN-GW address and the Access Point Name (APN) details to the UE, which serves as a default route to the external networks.

On receiving a successful DIAMETER update location answer, the MME responds to the eNB with a NAS Downlink transport message on the S1 interface. In this message contains the MME UE S1AP ID which uniquely identifies the UE from a specific eNodeB. Both the eNodeB UE S1AP ID and the MME UE S1AP ID forms a context on these nodes that uniquely identify a subscriber. On reception of the DOWNLINK NAS TRANSPORT message, the eNodeB will store the s1ap context that contains both the S1AP IDs. eNodeB forwards the NAS-PDU which contains the MME-UE message to the eNodeB without interpreting its contents.

Once the HSS grants access for the subscriber to use the services of the network, MME triggers S-GW a GPRS Tunneling Protocol Create Session Request on the S11 interface. The message contains details like IP address of UE, IMSI, TAI, ECGI, Fully Qualified Tunnel Endpoint Id (F-TEID) and ip address of the MME, PDN-GW, APN allocated for the UE , EPS bearer ID that uniquely identify each bearer created for a given UE along with the other details like guaranteed and maximum bits rates, QCI parameters of the session, configuration options of the protocols like username and password for a service provider. Additional information about some flags that indicate whether the session request is triggered due to a regular attach procedure or new bearer request or

due to a handover procedure [RK09].

On receiving the GTP Create Session Request from the MME, S-GW sends a new create session procedure on the S5 interface between the S-GW and P-GW. The details of the message include the ip address of the S-GW, EBI information, UE related information like its ip address and the APN allocated for this UE by the HSS. PDN-GW creates a session between itself and the S-GW with the information received from the S-GW and thus completes the S5 bearer setup. Now PDN-GW signals the completion of the S5 bearer setup with the create session response message [RK09].

On successful creation of the GTP session between P-GW and S-GW, S-GW informs the MME on the S11 interface about the completion of the bearer creation. This message includes not only the details of S-GW endpoint identifiers of the session for the control plane traffic but also the user plane identifiers of S-GW as the eNodeB directly transfers the user plane traffic to the S-GW. It also informs the MME about the route to PDN-GW. This gives more flexibility for the MME to select user plane paths for the user plane traffic based on the load in the core network [RK09].

After the bearers in the core network are established, the user plane transport functions have to be established on the s1 interface as well as the radio interface so that the UE will be able to send and receive data from the external networks.

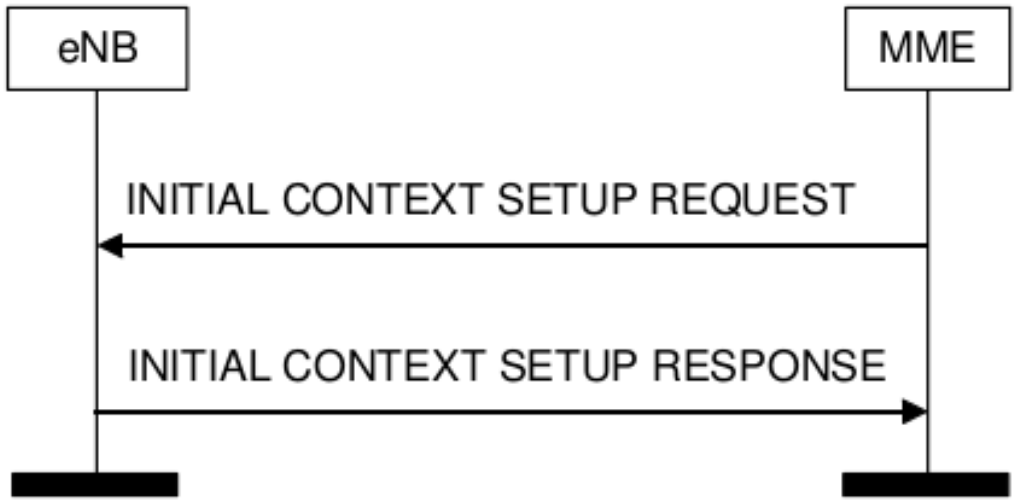


Figure 2.6: Initial Context Setup Request and Response

MME sends an S1AP initial context setup message to the eNodeB as shown in figure 2.6. Initial Context Setup procedure is need to create the needed UE context. The UE context includes UE radio capability, UE security capability, EUTRAN Radio Access

Bearer (E-RAB) context, Handover restriction list if any and security key[3GP13e]. This message contains all the information related to the UE that have to be stored on the eNodeB in order to establish the UE context between eNodeB and MME. The message also contains the details about user plane Tunnel Endpoint ID (TEID) of the S-GW so that the eNodeB can establish a GTP tunnel on the S1-U, the user plane interface. Along with the Initial context setup request, Activate Default Bearer Request message and NAS Attach Accept messages are forwarded to the UE transparently over the radio interfaces[RK09].

The Initial Context Setup Request message contains parameters like eNB-UE-S1AP-ID, which is sent by the eNodeB to MME during initial UE message, MME-UE-S1AP-ID, which is created by the MME. Both these ids make the unique context identifiers for a given UE. Hereafter Wards, all the NAS messages for this UE connection that are exchanged back and forth between eNodeB and MME will communicate using the same unique S1AP Id pair combination. The message further contains details like AMBR values for the Uplink and Downlink transport that are received from the HSS on the S6a by the MME. Other details of the message include e-RAB-IDS and QoS related parameters for each RAB. It also contains the transport layer addresses in either a bit string of 32 bits or 128 bits in hexadecimal format depending on whether it is a ipv4 address or an ipv6 address. If the system uses only ipv4 address, the ipv6 format address is set to 0. S-GW's user plane side is identified by a transport layer address along with the GTP TEID. MME sends the eNodeB details like security key and the security capabilities of the UE which helps to provide integrity protection and ciphering for the transmission on the radio interface [RK09].

The other section of the message contains the NAS messages that belongs to the EMM like NAS Attach Accept and Activate Default EPS Bearer Context Request. The EPS attach accept message contains the timer value for T 3412 that is used to control the periodicity of Tracking Area Updates. The timer value is expressed in deci hours. Each deci hour is equal to 6 minutes. Upon receival of this information by the UE, it understands that it has to do a periodic tracking area update as specified in the T3412 timer.

The message includes the Tracking Area List, GUTI and GPRS timer values. An UE can be registered with multiple tracking areas at the same time. Tracking Area List informs either one or more tracking areas, depending on how many tracking areas the UE is connected to. GUTI is the temporary identifier for the UE within the EPS. The GPRS timer value is used by UE to determine when to go to STANDBY state when there it is idle without doing any data transfer. The value of timer value is expressed in multiples of minutes.

The other NAS message in the S1AP Initial Context Setup Request is Activate Default EPS Bearer Context Request sent by the MME to the UE. A Radio bearer on the user plane interface is created after the UE receives this message that has details such as

APN, IP PDN address assigned to the UE by the HSS, QoS assigned the HSS, indicated by a Channel Quality Indication (CQI) parameter, usually with the default value of CQI=9 [RK09].

eNodeB on receiving the Initial Context Setup Request, if there does not exist an UE S1 logical connection already, UE establishes the GTP tunnel with the S-GW on the S1 user plane which is a part of the RAB[3GP13e]. As shown in figure 2.6, eNodeB responds to MME with an Initial Context Setup Response message that contains E-RAB setup list. This list contains for each created e-RAB-ID, the user plane transport layer address (ipv4/ipv6) and the GTP TEID on the eNodeB side of the tunnel.

On receiving the NAS message Attach Complete message, the UE sends a UE Accept message. After receiving the other NAS message called the Activate Default EPS Bearer request, the UE sends an Activate Default EPS Bearer Accept message that informs the MME that it is now in a state where it can send or receive data on the user plane with the QoS as specified by the HSS.

Now that the MME has received an UE Accept message, and both eNodeB and MME are ready to send or receive data, S-GW should be updated about the same, so that everything falls in place for data transfer across the user plane. S-GW will be informed about the eNodeB user plane IP address and GTP TEID to establish S1-U communication [3GP13a]. This information is passed to the S-GW by the MME using a GTP Modify bearer request which contains above mentioned details. S-GW in response to this request, sends a GTP Modify bearer response.

2.4 S1AP Detach

A Detach procedure can be used by the UE in the following scenarios:

- When the UE decides to detach from the EPS services
- When the UE wants to disconnect from the existing PDN it is connected to
- When the UE is switched off
- When the USIM in the UE is removed

A Detach procedure can also be initiated by the network when the network wants to inform the UE will no longer have access to the EPS services. For the sake of simplicity to explain the basic idea, the thesis document focuses on explaining the UE initiated detach.

EPS bearer context(s) for the given UE are deleted locally after performing the detach procedure for EPS services. This doesn't require any peer-to-peer signalling between the UE and the MME. Also UE and the MME will delete EPS mapped security context, if any on successful completion of detach procedure.

UE initiated detach procedure is triggered by a DETACH REQUEST message from the UE. The message contains Detach type IE which indicates whether the detach is caused due to switch off or not. It also indicates if the detach is to the EPS or non-EPS services or for both. The scope of the thesis work only deals with EPS only services. Also in this message the UE indicates the type of security context flag. It sets the flag to "mapped security context", in case if UE has a mapped EPS security context. If not, it will set the flag to "native security context".

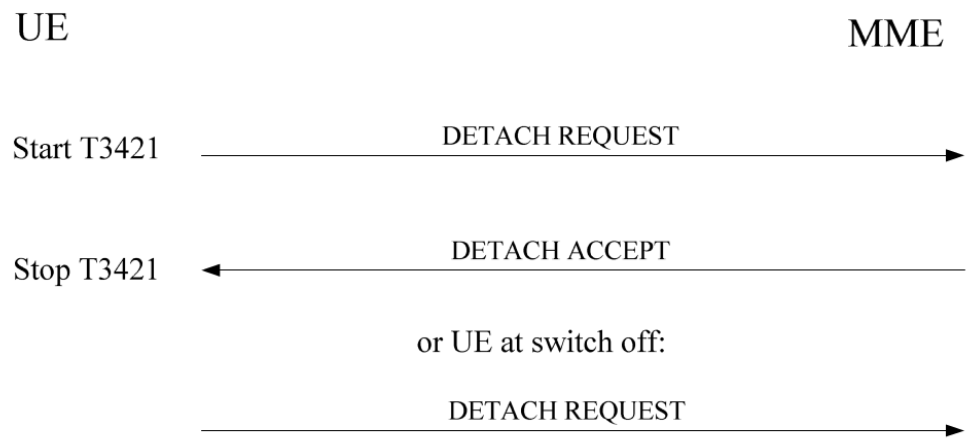


Figure 2.7: UE Initiated Detach procedure

When the UE is in EMM-REGISTERED Or EMM-REGISTERED-INITIATED, and if the detach is not triggered due to switch off, a timer T3421 is started on UE as soon as the UE sends the DETACH REQUEST message. Now, the UE enters into a state called EMM-DEREGISTERED-INITIATED.

When the UE is about to be switched off, UE will try to send a DETACH REQUEST message for a period of 5 seconds. The UE may be switched off after sending the DETACH REQUEST message. After sending the message, the UE removes the EPS security context, if any.

On receiving a DETACH REQUEST message by the network, the network shall react in two different ways depending on the Detach type as described below:

- If the Detach type indicates a "switch off", current EPS security context is removed by the MME, if the context is different from the EPS native security context
- else, a DETACH ACCEPT message is sent to the UE by the MME on , post removing the and stores the EPS security context.

MME on receiving the UE-initiated detach as described in step 1 in Figure 13, it stores the user's current NAS security context, it requests for termination of the activated EPS session. This request triggers P-GW-initiated EPS termination, releasing all the network/radio resources allocated to the user by releasing all the associated EPS bearers and S1 signalling connections.

On receiving the DETACH ACCEPT message as described in step 2 of figure 2.8, the UE will stop the T3421 timer and enters into a EMM-DEREGISTERED state. In Step 3, MME request the eNodeB to release all the resources allocated for the UE session. eNodeB responds to MME in step 4 after the UE context is removed from the eNodeB. The summary of the sequence of steps that complete the detach process are described in the figure 2.8 below:

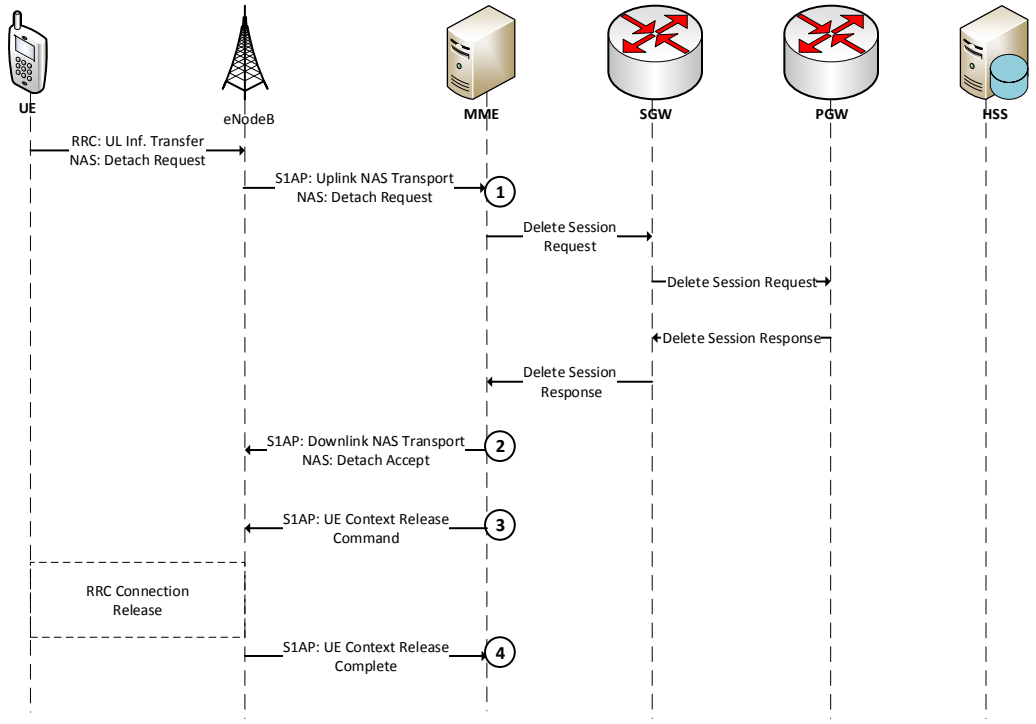


Figure 2.8: Summary of S1AP Detachment procedure

2.5 S1AP Handover procedure

In LTE, mobility of an active UE from one cell to another is called a handover. Handovers can take place through S1 interfaces or X2 interfaces. S1 interfaces deal with indirect communication between source eNodeB and target enodeB through MME, while X2 deals with direct communication between source and target eNodeBs, which is not in the scope of the discussion of our thesis.

2.5.1 Overview of S1 Handovers

Typically, an intra LTE S1 handover is performed in three steps namely handover preparation, handover resource allocation and modification of the S1-U bearer[RK09] as described in figure 2.9. LTE supports hard handovers which means that the connection with the current serving cell is broken before a new connection is made with the target cell.

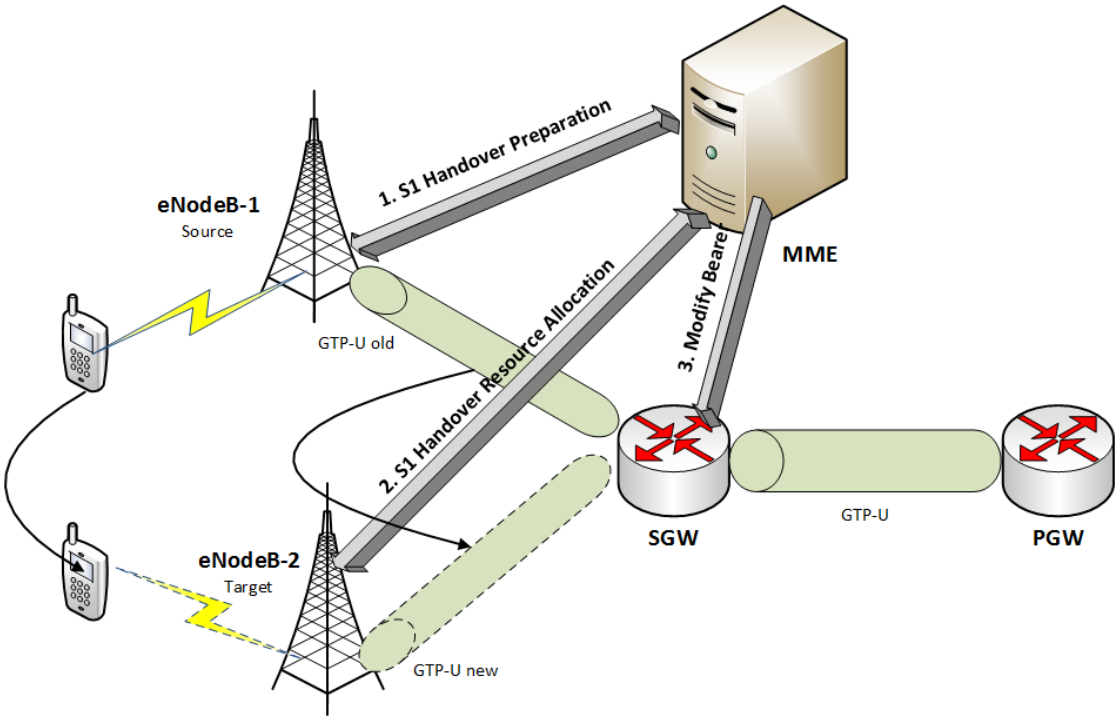


Figure 2.9: Overview of S1 Handover procedure

The UE in advance has the details from serving cell on what parameters it has to measure while it is moving across cells [3GP13d]. The UE sends the measurement report to serving eNodeB if any predefined criteria set by the eNodeB is met during signal strength measurement while it is moving from one cell to the other. Based on the received measurement report, serving enodeB will decide if a switchover is required.

The UE will send a Measurement Report to the source eNodeB during Handover preparation stage. When the Measurement Report is received, the eNodeB will make the decision to handover to one of the neighboring cells, which is the target eNodeB as shown in figure 2.9

Now, the Handover resource allocation phase starts, during which the source eNodeB, say eNodeB 1 as in figure 2.9 sends a relocation request using Handover request message during preparation phase to MME which then checks with the target eNodeB (which is designated by eNodeB 2 in figure 2.9) if the required resources are available to service the UE. If the target eNodeB has enough resources to service an UE, it then agrees to the request made by MME by acknowledging it with an S1AP Handover Request Acknowledge message. This triggers the Modify bearer part of Handover procedure where MME has to modify the existing GTP-U bearers which carry the UE traffic from the current bearer between S-GW and eNodeB 1 (source) to a new bearer between S-GW and eNodeB 2 (target) as described in figure 2.9.

2.5.2 S1- Handover message flow

This section explains all the messages that are exchanged in the network to realize the handover process. It outlines the sequence of messages transferred between different network elements in order to realize the Handover of the call from one eNodeB to the other eNodeB. The purpose of each message between the network elements in explained in detail in further sections.

Initially a UE is being serviced by S-eNB, the source eNodeB which in turn is connected to MME for control plane signaling and S-GW for user plane forwarding. At this moment, uplink and downlink data traffic between UE and S-GW keeps flowing. Source eNodeB gives the UE information of Measurement control parameters and criteria UE has to follow through a Radio Resource Control (RRC): Measurement Control message. When the UE moves from one geographical location to the other, it sends an RRC: Measurement Report to the source eNodeB.

The S-eNB based on the measurement report it received, decides whether to make a handover of UE to the new cell. If the S-eNB decides to do a handover, it first sends a S1AP: Handover Required message to the MME as shown in figure 2.10. Figure 2.10 describes further summary of steps involved in moving the UE from the source eNodeB to target eNodeB. This involves several S1AP procedure calls as shown in figure 2.10. Each S1AP message is explained in detail in further sections.

This part of the summary as described in figure 2.11 of handover procedure explains all the s1ap messages that are exchanged after the UE is completely moved to the target eNodeB and starts being operational. The next procedures indicate to remove the

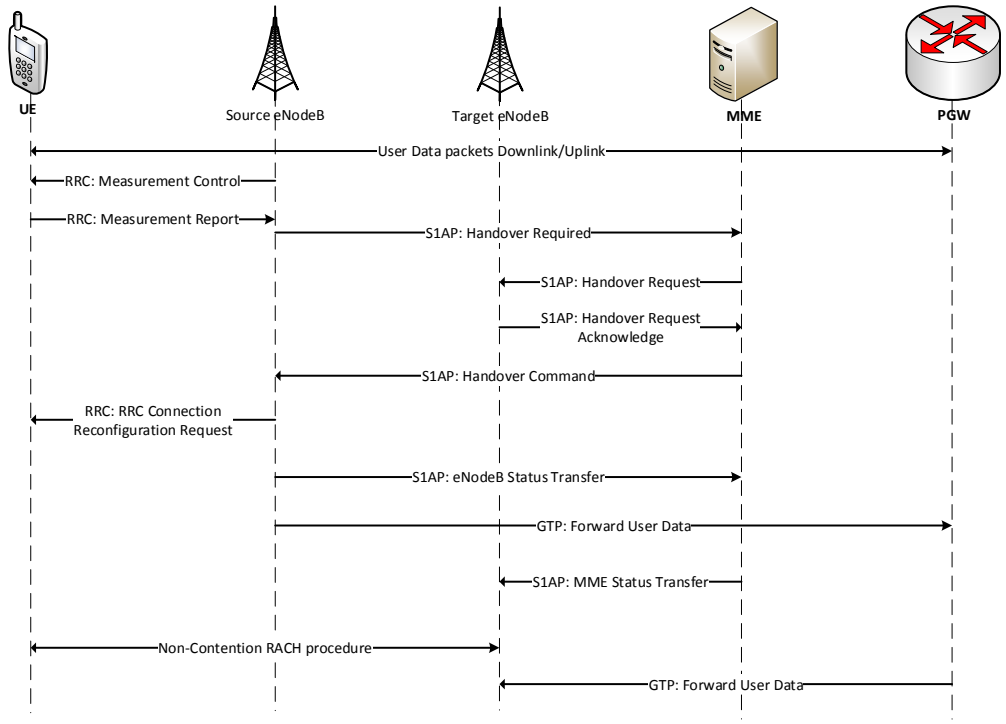


Figure 2.10: Intra-LTE Handover message flow using S1 interface-part 1

resources allocated for this UE as requested the source eNodeB side. Target eNodeB indicates the MME that there is a transition of the UE from source eNodeB to target eNodeB, which triggers MME to release the resources allocated for the MME. Step by step procedures are explained in detail in further sections.

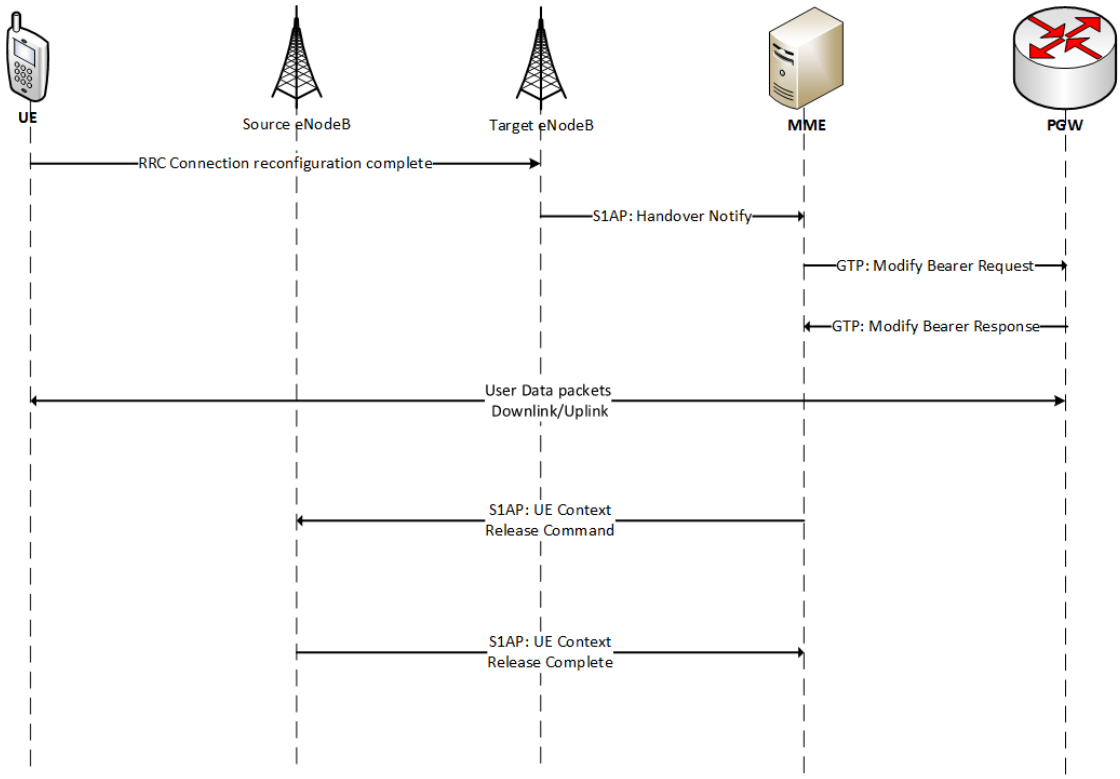


Figure 2.11: Intra-LTE Handover message flow using S1 interface-part 2

As explained in summary of S1 Handovers, the source eNodeB after identifying that a handover is required, it sends a Handover Required message to the MME as shown in figure 2.12.

This message contains procedure code "Relocation preparation", handover type (eg: Intra-LTE handover), cause, and target ID (eNodeB ID) and TAI (tracking area identity) and source-to-target-Transparent container that contains UE details that will be needed by the target eNodeB (T-eNB) to continue the ongoing UE connection to the EPC.

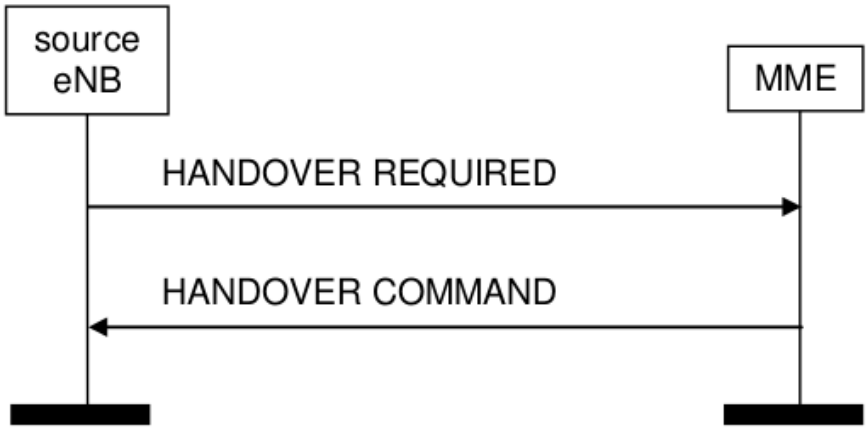


Figure 2.12: Handover Required message

Based on the details in Handover Required message received by MME, it sends a S1AP: Handover request message to the T-eNB as described in figure 2.13.

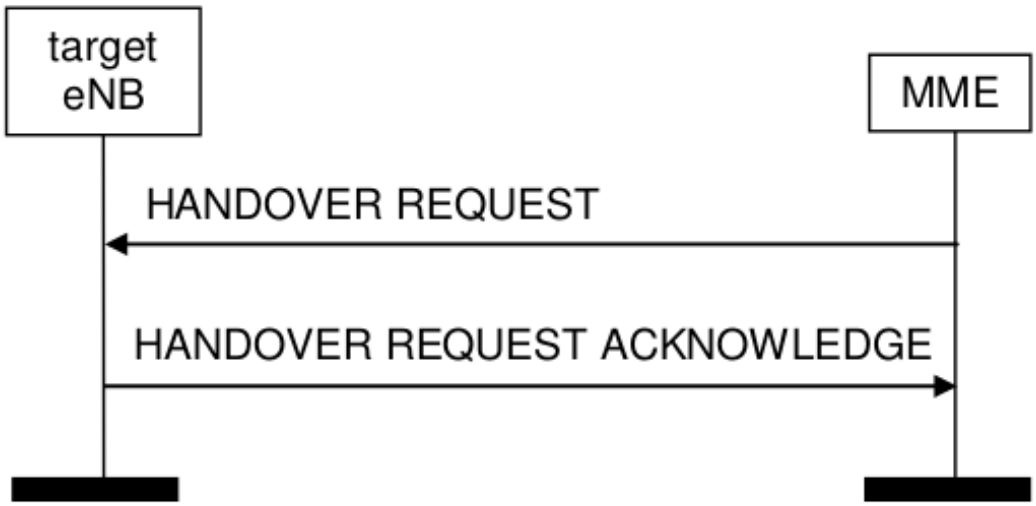


Figure 2.13: Handover Request Message

Handover Request contains the same details as sent by S-eNB along with list of e-RABs and the corresponding QoS values for which S1 user plane GTP tunnel needs to be switched.

On receipt of the Handover request message by the T-eNB, it allocates the required resources to UE based on the availability and then sends the RRC connection reconfiguration details in RRC handover message embedded in a Handover request acknowledge

message to the MME.

MME, on receipt of the Handover request acknowledge message, sends the S1AP: Handover command to the S-eNB as shown in figure 2.14.

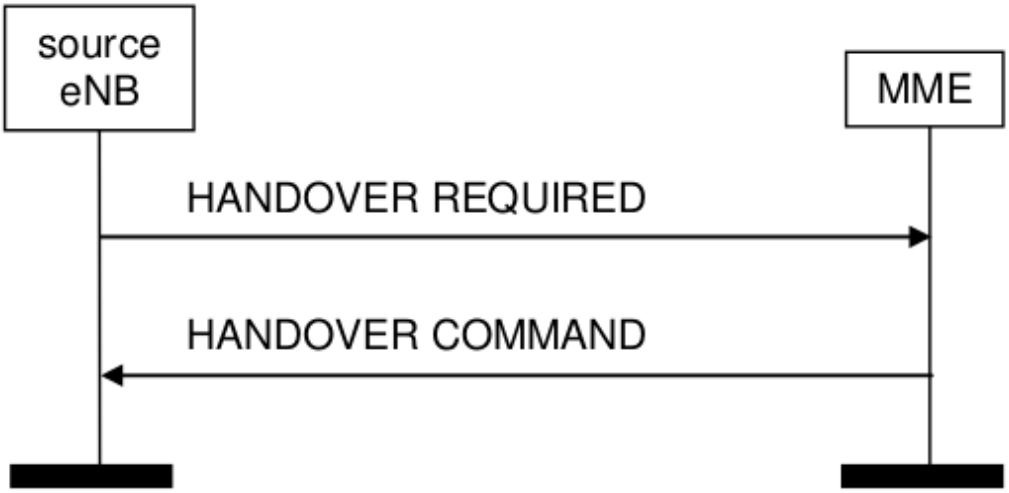


Figure 2.14: Handover Command Message

Now the S-eNB forwards the Handover command message transparently to the UE thus allowing the UE to execute the handover procedure on the radio interface layer.

Once the UE leaves the S-eNB, it sends a status S1AP: eNB status transfer message to the MME as shown in Figure 2.15. This message contains last PDCP SNs (sequence numbers) sent or received by the S-eNB in both the uplink and downlink direction.

Figure 2.15: eNodeB status transfer

On receipt of the S1AP: eNB status transfer message by the MME, the contents of the message are forwarded to the T-eNB using a S1AP: MME Status Transfer message as shown in figure 2.16. This allows the T-eNB start transferring or receiving the data from the point S-eNB has stopped.

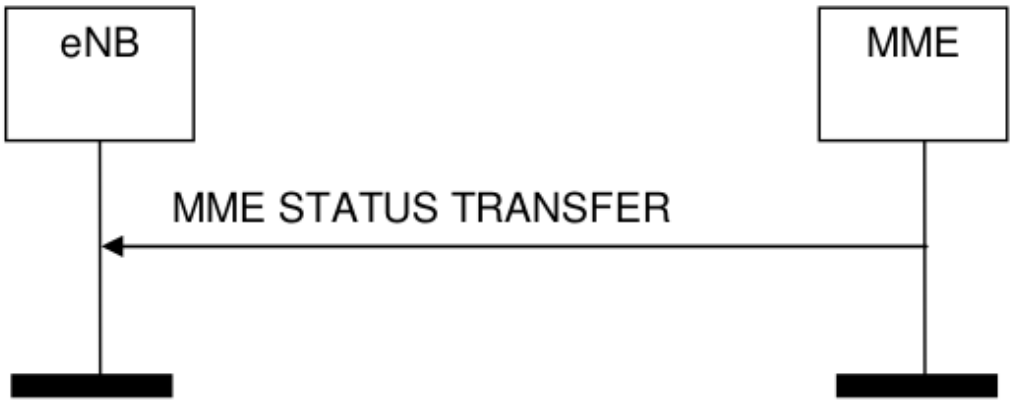


Figure 2.16: MME status transfer

When the UE finally moves to the T-eNB, indicated by a RRC connection reconfiguration complete message from UE to the T-eNB, which transitions the UE to a state where it is fully functional with both its control and data plane traffic being operational; T-eNB indicates the MME that handover is successful by sending a S1AP: Handover Notify message as described in figure 2.17.

Figure 2.17: Handover Notify

This triggers the MME to send modify GTP bearer request to S-GW to change its tunnel endpoint identifiers. After changing the tunnel endpoint identifiers in the downlink direction, the S-GW acknowledges the MME by sending a GTP: Modify bearer response.

Now, MME send a S1AP UE context release command to the S-eNB, so that it can release all the transport and radio resources it allocated to the UE, which are no longer required as the UE moved to the T-eNB. It indicates the release of resources by a S1AP UE context release complete message to the MME.

2.6 S1-Flex

S1-Flex mechanism allows the eNBs to be connected to multiple core network elements on S1 interface. S1-Flex requires a full mush connection between eNodeBs and core network elements like MME and SGW[KS11] as shown in figure 2.18.

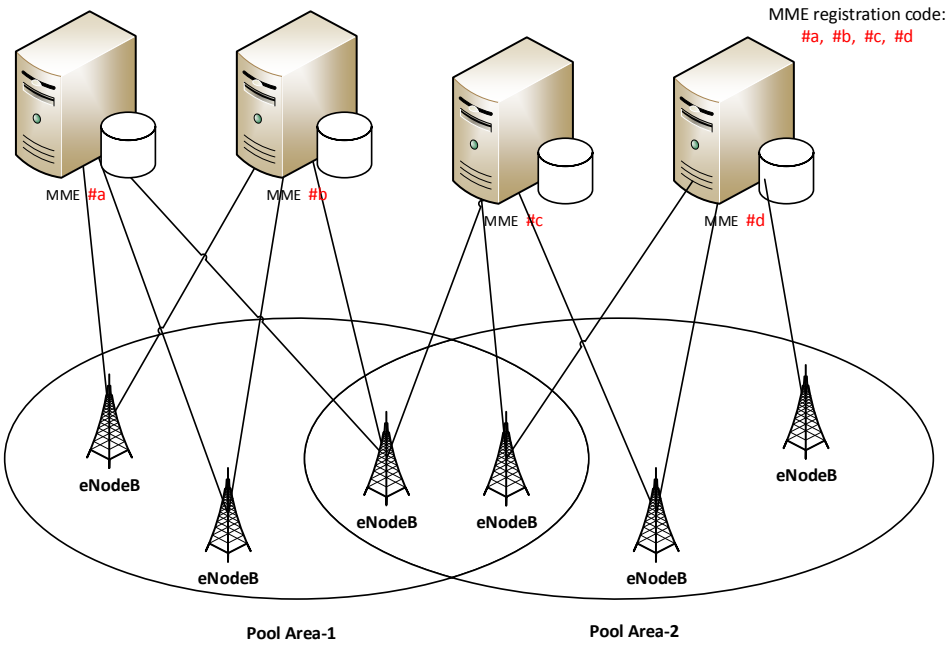


Figure 2.18: S1-Flex Mechanism

S1-Flex provides the following benefits to service providers:

- It allows load sharing of traffic.
- It provides network redundancy.
- It allows the Network operators to share the access network.

A pool area is created for a set of MMEs or SGWs. A MME pool area allows a mobile to shift between multiple MMEs without the need to change the serving MME [3GP12].

Each eNodeB is connected to every core network element of the pool area as shown in figure 2.18. S1-Flex increases the reliability of the network as any incoming calls will be handled by one of the working MMEs, even if some of the other MMEs in the MME pool are out of order for some reason.

In S1-Flex, when a UE sends a registration request, the eNodeB selects the MME from a MME pool area. As shown in figure 2.19 below, eNodeB selects MME with registration code a, after which MME a sends a registration response. When next time UE sends a service request to the network and in case if previous MME i.e., MME a went down for some reason, eNodeB switches the service request to MME with registration code b. This switch is possible only when the access network element eNodeB is aware of the failure of MME a, thus forcing it to take corresponding action of switching the UE to MME b by terminating its previous sessions with MME a.

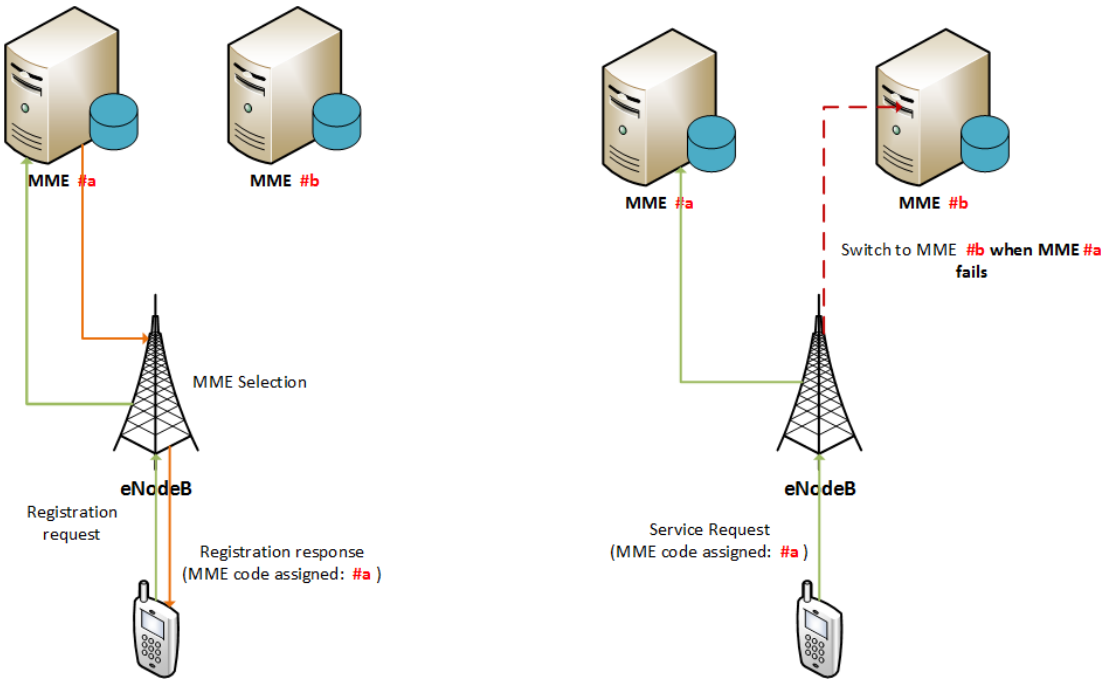


Figure 2.19: S1-Flex handling MME Failover

Though S1-Flex provides some advantages, but they come only at the expense of some overload in communication between access network and core network elements. It requires any changes in the core network elements like MME to be announced to the eNodeB. This involves an expensive round trip time operation for the signaling information. This reduces the flexibility of effectively scaling in/out the MME instances in

the core network. eNodeBs receive duplicate warning messages as they are connected to multiple MMEs of the pool using the S1-Flex mechanism[3GP12]. But these messages are not allowed to be transmitted on the radio interface as they will be discarded after interpreting the message identifier and sequence numbers.

2.7 Load Balancing Techniques

A load balancer is an network element that distributes application or network traffic between computing resources, such as generic computers, cluster of computers, processing units, I/O devices and network resources. The goal of load balancing is to optimize resource utilization, maximize throughput, minimize response time [WIK16].

A load balancer can be broadly classified into two categories [F5 16]:

- Layer 4 - A layer 4 load balancer splits the traffic based on the data in network and transport protocols such as IP, TCP, FTP, UDP etc.
- Layer 7 - A layer 7 load balancer distributes the traffic after inspecting data from application layer protocols like http.

Figure 2.20 explains an example where the application traffic between different servers in the cloud is load balanced by the balancer element.

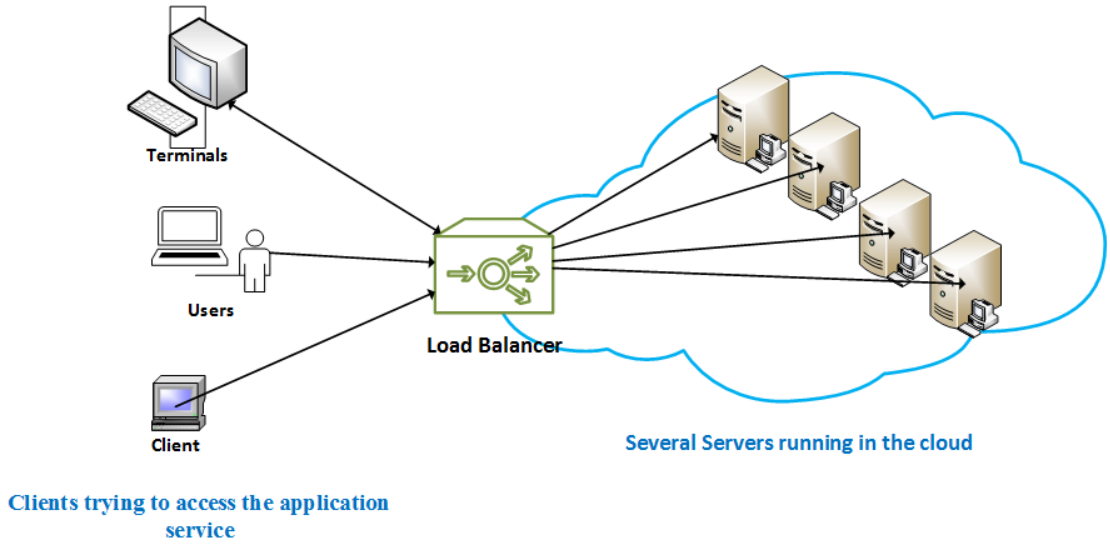


Figure 2.20: Load Balancing application traffic between servers in the cloud

Usually these load balancers are connected to some sort of monitoring tools which informs it the status of the servers in the cloud. As an example when the load balancer comes to know that a specific server is down, it will direct the requests to some other

server in the cloud.

In general, a load balancer provides the following benefits:

- **Redundancy** - A Load Balancer facilitates the provision of introducing additional resources like computing elements, alternate network paths to handle the application data or network traffic. In case of any failure, the redundant resource can handle the incoming traffic or request thus improving the availability of the resources for the application/network traffic processing. It serves as a backup mechanism in the event of unplanned infrastructure outages.
- **Scalability** - A Load Balancer allows to scale the computing resources that addresses growing load due to the number of service requests or network traffic. The number of computing resources can either be increased or decreased on a need basis.
- **Reliability** - Load Balancers can improve the reliability of the application service by ensuring that the requests are sent only to computing elements that can respond in a time bound manner, thus improving the reliability of the application service or network. It improves the overall efficiency of the network or application service.
- **Removes the bottlenecks** - In the absence of load balancer, there arises a situation where all the application or network traffic requests need to handled by a single resource and this makes it a bottleneck. This can be alleniated by introducing a load balancer in the network or application data traffic processing environments.

The following are some of the basic characteristics of a Load Balancer:

- **Transparent** meaning that, the client that is connected to a load balancer for seeking a service doesn't know how the load balancer is dealing with the request. In reality, the load balancer simply acts as some sort of proxy while the actual work is handed over to some other resource in the backend. This resource processes the request and replies to the load balancer and in turn the load balancer forwards it to the corresponding client.
- **Persistent** meaning that, a load balancer always sends its data traffic/service request to the same computing/network element as distributed by it earlier. When a packet is arrived at the load balancer, based on some characteristic of the incoming packet, it forwards the packet to a specific resource. At a later point of time, if a packet arrives with same characteristics identified earlier, it sends it to the same resource.
- **Light in weight** meaning that a load balancer should be as simple as possible without involving complexing computing. It should be able forward the incoming traffic or service requests to the backend resources as fast as possible. This needs the load balancer not to interpret the incoming packet in full and not to maintain

full session management for all the field of the packet. Only some important fields of the packet that uniquely identify its direction back and forth between the client and the backend resource should be maintained. This helps the load balancer to minimize the amount of processing involved at loadbalancer and thus reducing the overall delay in the end to end communication.

- **Scheduling policies** A load balancer is usually a software program that listens on a specific port to incoming requests. The clients usually connect to this port for sending requests. The load balancer forwards this request to one of the backend resource based on preconfigured scheduling algorithm. Several algorithms are used based on the requirement. The simple ones are such as random selection, round robin etc. The simple ones are easy to implement, but the downside is that they may not able to address some special requirements given some situation. There are several other complex ones designed taking into consideration several factors like up/down status of backend resources, geographic location, traffic load, recently used resource etc [WIK16]. These complex ones are hard to implement but they can cater the needs of several special purpose application needs.

2.8 Conclusion

In this chapter, the technological foundations relevant to the thesis are briefly explained. It explains 3GPP EPC Architecture, S1AP Setup, Attach, Detach and Handover procedures. Furthermore, it explains how the S1-Flex and generic load balancing mechanisms work and their inability in addressing the current core network requirements. Eventually this sections lays a foundation to the arguments that justify the need for a new mechanism to handle the rapidly evolving mobile core network requirements.

3 Requirements

This chapter explains several driving facets of the problem explored in the thesis. It analyses the currently available solutions in the market, and identifies their limitations and discusses the need for MME load balancing solution which can provide an efficient and flexible core network running in virtualized environments that can adapt to the changing needs.

This chapter also discusses the functional and nonfunctional requirements of the above discussed load balancer, considering the needs for a flexible core network while trying to follow the recommendations in 3GPP standards. Functional requirements define the components and the set of functions that load balancer has to achieve, while the non-functional requirements define the attributes that measure the quality of the system.

3.1 Problem Statement

Recent studies and the past trends forecast reveals that in the next decade time, there will be a 500 to 1000 fold increase in total mobile data traffic[Eur15]. With every changing day, it is becoming more and more easier for a common man to buy a smart mobile. These statistics assume that there will be as high as 10 times increase in broadband mobile subscribers. With the advancements in technology, the market is offering several smartphones and tablets at very low price, but can run several applications that support multimedia and sensing capabilities. Users running these applications need good broadband experience. This in turn puts pressure on service providers to cope up with the growing demand.

With the introduction of high-end smartphones and their support for different applications and services, the mindset of the mobile users is also changing with every passing day. Some of the requirements from the user perspective are:

- Always-on connectivity
- Ubiquitous user experience
- Quick response time for applications
- Competitive pricing
- Good QoS and QoE

It is really important for the service providers to satisfy the user requirements while trying to effectively utilize their network infrastructure. There are reports that the total expenditure by global mobile operators have exceeded 800 billion USD[Jun15], while some of the major service providers might run into a situation where their expenses exceed their revenues in a decade time, which is really alarming. Therefore, the service providers should take a look into the way they operate and maintain their infrastructure.

With the changing paradigm in mobile network architectures, it is really important for the operators to try several novel ideas on their networks that can improve the efficiency of their networks that can meet customers' expectations while trying to reduce the costs involved.

In order to give users a good experience it is important for the service providers to manage and operate their networks in a certain manner. There are some focus areas that needs attention from service providers. Mention below are few of the focus areas and desirable mechanisms to address the problem.

3.1.1 Focus Areas

3.1.1.1 Good Network coverage

Users expect to have good network coverage wherever they go. It requires the service provider to maintain their own access network infrastructure everywhere. But, this incurs huge expenses in building and maintenance of the infrastructure. Also it is time consuming to install infrastructure everywhere and not beneficial to have own infrastructure everywhere especially in areas with less users.

In situations like this, it is desirable to have RAN sharing which allows the access network to be shared between different operators while having their own core networks. This is mutually beneficial to operators and help them in cutting down their costs and time to market, thus giving them an edge over their competitors.

3.1.1.2 Uninterrupted service

Service interruption can be caused either due to the failover of network elements in access network or core network. Since the scope of the thesis doesn't cover the scaling capabilities of access network, the discussion related to its research is skipped and focused on core network. Sometimes service disruption is caused due to failover of core network elements like MME.

This problem can be addressed if there is some mechanism by which the duties of a failed MME is taken up by another MME. This requires the service provider to maintain

redundant MMEs to handle the failover nodes.

3.1.1.3 Quick response for application needs

Some applications run by the user need quick response from the resources in the network. As an example, real time traffic needs quick response. These applications in the background might require some new paths to be established in both the access network and core network. This in turn requires some control plane signaling between the UE and MME via eNodeB. MME needs to quickly respond to the control plane signaling messages to establish the required paths. This needs some sort of load sharing between MMEs so that the delay in response to the requests coming from the UE is reduced and thus the user have a good quality of experience.

3.1.2 Existing approach and its problems

So, in order to address these focus areas, service providers can use S1-Flex mechanism. S1-Flex provides an ability to load share the control plane signalling among multiple MMEs, provide network redundancy and helps the service providers to share the radio access networks. But this only comes with extra cost. In S1-Flex, it is mandatory to have a eNodeB connected to all the MMEs in the MME pool thus exposing all core network structure making its functionality more complicated. Any failover in any of the MMEs is completely visible to the eNodeB and eNodeB should act accordingly. For instance, though S1-Flex allows to move all the UEs from one MME to another MME. First all the sessions between these UEs and the failed MME should be completely killed and then they need to reestablish connections with the new MME. This involves a lot of messages being exchanged between MME and the eNodeB which makes it a time consuming process. It is not possible to hide the changes in the core network. This in turn reduces the flexibility in scaling the core network. Also since the eNodeB is connected to several MMEs in the MME pool, it receives several duplicate messages from the other MMEs which is an overhead on the access network as it involves some processing to discard those messages.

Some of the major reasons why mobile operators are seeing an increase in their expenses are over provisioning of networks, which means that extra resources are allocated in order to address the traffic needs in peak hours. Service providers in its previous architectures have fixed infrastructure, leading to higher cost of equipping and maintaining the infrastructure. But, It is good to have a mechanism where based on the traffic needs, the number of computing elements in the network could be either increased or decreased dynamically based on the incoming traffic instead of having fixed infrastructures. This reduces the cost involved in reserving extra fixed hardware resources to handle peak load that rarely appears in the network [Fuj14]. This pushes the need for development of scalable and flexible core networks within virtualized environment that adapts to the

varying mobile subscriber needs. This can be achieved by following the latest trends in 5G Mobile architectures where Software defined Networks (SDN) and Network function Virtualization are used to build, operate and maintain the networks.

For the above mentioned reasons, most of the service providers are moving their network elements to cloud running them as virtual elements. While it is good to move them to the cloud, but that in itself is not sufficient to extract the full scale benefits of deploying the infrastructure in the cloud. For example, in the current approach, service providers can connect their eNodeB directly to a virtualized instance of MME running in the cloud, but when the MME fails, the details are exposed to eNodeB and thus as explained earlier as in the case of S1-Flex, which reduces the flexibility in core network.

Hence, a new S1AP-MME load balancer is needed between eNodeB and MME. This load balancer hides all the details of the core network. eNodeB assumes the load balancer as MME, while the MME which is connected to load balancer assumes it to be eNodeB. This gives a lot of flexibility to scale in or scale out the MME instances in the core network. A network orchestrator can continuously monitor the health of MMEs in the core network and divert the traffic to alternate MME in case if the working MMEs goes down for some reason.

This avoids all the overhead due to control plane signaling between eNodeB and MME before the eNodeB is taken into service by a new MME, while at the same time, the load balancer provides an ability for the service providers to do RAN Sharing.

3.2 Functional Requirements

This section describes the functional requirements of the S1AP-MME Load Balancer. It describes several components that need to be supported by S1AP-MME load balancer and the functionality each of it has to provide. This section further discusses the inputs each component receives and expected output from them.

3.2.1 Message Flow Handling

S1AP-MME load balancer comes in between the eNodeB and the MME who communicate on the S1 interface. S1 interface is split into S1-MME and S1-U. S1-MME carries S1 control plane traffic while S1-U carries user plane traffic. S1AP-MME load balancer supports the balancing of control plane traffic going between different eNodeBs in access network to the MME elements in the core network. The load balancer reads each packet on the S1 interfaces and forwards it to the corresponding MME based on policing functionality, whose functionality is described as part of the Load Balancer Component functions.

3.2.2 Transparency

It is required that S1AP load balancer gives flexibility to the core network. This is possible when any changes in the core network is hidden from the access network by the load balancer. The very basic need of this load balancer is to avoid any direct communication between eNodeB and the MME so that the changes in the core network element doesn't need to be informed to the eNodeB, unlike the S1-Flex approach where eNodeB and MME are directly connected to each other and thus the access network will see any changes in core network. S1-MME load balancer providing transparency means that the eNodeB in the access network assumes load balancer to be MME, thus making it a proxy node between eNodeB and MME.

3.2.3 Load Balancer Components

This section discusses different components needed by the S1-MME load balancer and explains in details what functionality is expected from each of these components.

3.2.3.1 Core Message Handling

The load balancer should support different S1 setup procedures as listed in the table 3.1 below:

Serial No	Procedure support needed
1	S1AP Setup
2	S1AP Attach
3	S1AP Detach
4	S1AP Handover

Table 3.1: S1AP Procedures Supported

These procedures in turn need support for handling specific messages carried on the s1 interface as listed in table 3.2. These messages are transferred between eNodeB and MME to realize the support for the above stated procedures as in table 3.1. Hence the S1-MME LB should support these messages and be able to comprehend them and take necessary action based on the type of the message received.

Serial No	Message type support needed
1	S1 SETUP REQUEST
2	S1 SETUP RESPONSE
3	INITIAL UE MESSAGE
4	S1 INITIAL CONTEXT SETUP REQUEST
5	S1 INITIAL CONTEXT SETUP RESPONSE
6	UPLINK NAS TRANSPORT
7	DOWNLINK NAS TRANSPORT
8	UE CONTEXT RELEASE REQUEST
9	UE CONTEXT RELEASE COMMAND
10	UE CONTEXT RELEASE COMPLETE
11	HANDOVER REQUIRED
12	HANDOVER REQUEST
13	HANDOVER COMMAND
14	HANDOVER REQUEST ACKNOWLEDGE
15	eNB STATUS TRANSFER
16	MME STATUS TRANSFER
17	HANDOVER NOTIFY

Table 3.2: S1AP Messages Supported

3.2.3.2 Balancer Policing

Traffic policing is a method by which network traffic is monitored for agreement of specific set of predefined traffic rules. It enforces that the incoming traffic complies to those rules. S1AP-MME load balancer receives the messages coming from eNodeB and inspects certain fields of the message and forward them to the concerned MME.

Given the scope of thesis, the following are required:

- Any incoming S1AP Setup Request coming from a new eNodeB must be assigned to some MME. For now, it is good enough to have some sort of round robin mechanism where any first packet coming from eNodeB can be assigned to a random MME.
- Any next request coming from the same eNodeB should always be sent to the same MME as assigned earlier for the first packet coming from eNodeB.
- The S1-MME LB should ensure that incase an assigned MME is failed to receive the forwarded request, the load balancer should reassign another MME, till it finds a working MME that can accept the request forwarded by loadbalancer.

3.2.3.3 External Management support

A network orchestrator keeps track of the changes in the core network and informs the load balancer some actions to be taken so that access network still continues to use the services from MME on the core network without any interruption.

In order to achieve this, the S1AP-MME load balancer should support the following operations on the console:

- An option to add additional MME to which the load balancer can forward the incoming messages to.
- An option to replace an existing MME with a new MME. This support is specially needed when the orchestrator identifies that a specific MME went down and it wants the users sessions on this MME to be handled by another redundant MME.

- Delete all the user related mappings on the load balancer.

The set of Console operations that should be supported by a S1-MME can be summarized as shown in table 3.3 below:

Serial No	Operation support needed
1	add_mme_ip
2	replace_old_mmeip with new_mme_ip
3	delete_all_sessions

Table 3.3: Management Operations Supported

3.2.4 Configuration Management

The load balancer needs to be uniquely identified by access network elements like eNodeB and core network elements like MME using some transport layer addressing so that they exchange messages back and forth between them. Also, the load balancer needs default configuration to startup the node. Some of the details that the node needs to know include the ip address by which it is identified, the ports on which it will listen to communicate with eNodeB and MME. It also needs to know if there is any default MME to which it can forwards the incoming requests from the access network.

3.2.5 3GPP Standard Compliance

The S1AP-MME load balancer processes the messages that are carried on the S1 interface. Required signaling information is exchanged between eNodeB and MME so that the mobile users can utilize the mobile network services.

Since the load balancer operates on the s1 interface, it is important that it adheres to the 3GPP standards that define the S1 interface. While it is possible that some changes might be required to accommodate the load balancer in between eNodeB and MME, it is important to ensure that these changes as minimal as possible and as closely compliant as possible to the 3GPP standard.

3.3 Non-Functional Requirements

This section describes the non-functional requirements of the S1AP-MME load balancer. These requirements helps to measure the quality of the s1ap load balancer from different dimensions as discussed in the section below:

3.3.1 Modularity

Modularity refers to decomposition of software package into a number of independent components that can interact and exchange resources through some standard interfaces. Each component achieves only a specific aspect of the required functionality. When these components collectively work together, they achieve a generalized bigger functionality needed. These modules expose only the those details to each other in order for them to perform their independent tasks. This sort of modularity in software development helps in understanding the functionality in a very easier way. It helps to quickly debug the issues as it is easy to identify root cause of any issues in the implementation. This is especially helpful when there are some design decisions in implementation that can be changed in future for improving the functionality. In this case, it makes it easier to change only certain modules where new changes are necessary and thus reusing the remaining modules

It is important to implement the S1AP-MME load balancer functionality in a modular fashion as it makes it easy to maintain the implementation considering the continuous improvements in the mobile network architecture. As an example case, the current requirement for the balancer policing is that the load balancer assigns an MME to each eNodeB by using a simple round robin like algorithm. Since the balancer policing is implemented as a separate sub module, in future it is easy to change the MME assignment procedure to eNodeB considering several other complex parameters like load on the MME, delay to reach a specific MME etc, only by making changes in balancer policing module without disturbing the other functionality implemented in other sub modules of the load balancer.

3.3.2 Performance

Performance describes the parameters that helps to evaluate any system. It is one of the key indicators that evaluates the quality of the system. Performance can be evaluated based on several parameters like load testing, stress testing, throughput testing, delay etc. Since this is a prototype implementation, the focus is emphasised on proving the point that S1AP-MME load balancer can serve its purpose while being lightweight, meaning that it will not introduce much delay in the communication path between eNodeB and MME, while achieving the required functionality at the same time. It is desirable that the load balancer doesn't introduce a delay not more than 10% of the

time it usually takes for the direct communication between an eNodeB and the MME.

3.3.3 Interoperability

Interoperability can be defined as the ability of two systems to interoperate or communicate with each other irrespective of the manufacturer of the systems. This characteristic of the systems should allow them to exchange the data through some common protocol. It helps the operators to choose different network elements from different vendors without the need to worry about the compatibility issues between the systems. Hence, it is important to design the S1AP-MME load balancer keeping in view the interoperability requirements.

There is a continuous evolution in mobile network architectures due to the growing needs from customers. With every passing day, mobile users tend to use several bandwidth-hungry applications. Apart from application needs from humans, there are several new applications like vehicle-to-vehicle communication, machine-to-machine communication, increased automation in several fields that are pushing the need for the networks to be more flexible to adapt to the changing needs. This is pushing the operators to slowly move towards a cloud based environment while still trying to use their legacy hardware platforms in the best possible way to get their return on investment on these legacy systems.

Given the situation, S1AP-MME load balancer should be flexible enough to fit into this kind of mixed mobile network architecture, which is based on a combination of both the cutting edge cloud based infrastructure and legacy hardware infrastructure. The load balancer should be able to communicate with either a hardware based eNodeB or a virtualized eNodeB in the access network and be able to connect with either a hardware based MME or a virtualized instance in the core network. This makes the S1AP-MME load balancer flexible enough to fit into any mobile network architecture that needs rapid innovation with the changing requirements.

3.3.4 Portability

Platform independence of a system refers to a method in which the technology is developed independent of the underlying platform. It is needed that S1AP-MME load balancer should be able to run on any x86 processor hardware. The load balancer implementation should run on any commercial off the shelf (COTS) hardware instead of having the need for any specialized hardware.

3.3.5 Usability

Usability refers to how easily the users can use your application/service. Given the context of this thesis, usability is of just minor importance. It is good enough that the S1AP-MME load balancer component has a command line interface using which the users can interact with the system to configure it.

3.4 Conclusion

This chapter introduced the non-functional and functional requirements for the design of S1-MME load balancer. The requirements for the load balancer are thoroughly studied considering the current generation Mobile network architecture needs, and several requirements were proposed justifying the need for each of them.

The major requirements set for a S1-MME load balancer were as below:

- It should be as much 3GPP standard compliant as possible, meaning that only very minimal changes (or no changes) to the 3GPP standard are allowed, to accommodate it as a new element in the core network.
- It should be transparent to Access network, meaning that the eNodeB doesn't know the existence of Load Balancer, and continue to work with load balancer as if it has connected to the MME, while the MME knows the existence of load balancer, which remains as part of the core network.
- It should be as light as possible in terms of delay latency, meaning that it should not introduce higher delay in the communication path between eNodeB and MME. It is desirable to have only around 10% overhead in the delay while a eNodeB attaches to the MME through the load balancer.

4 Design and Specification

In this thesis work, an S1-MME Load Balancer(LB) is developed in order to dynamically scale the MME in core network. The focus of the work is to see that the S1-MME load balancer (LB) sits into the existing EPC environment and ensure that it supports all the basic S1AP procedures. Design is made to ensure that it doesn't introduce much delay during attaches. Design of the S1-MME LB also considered the need for being 3GPP standard compliant.

4.1 Architectural Overview of EPC with S1-MME LB

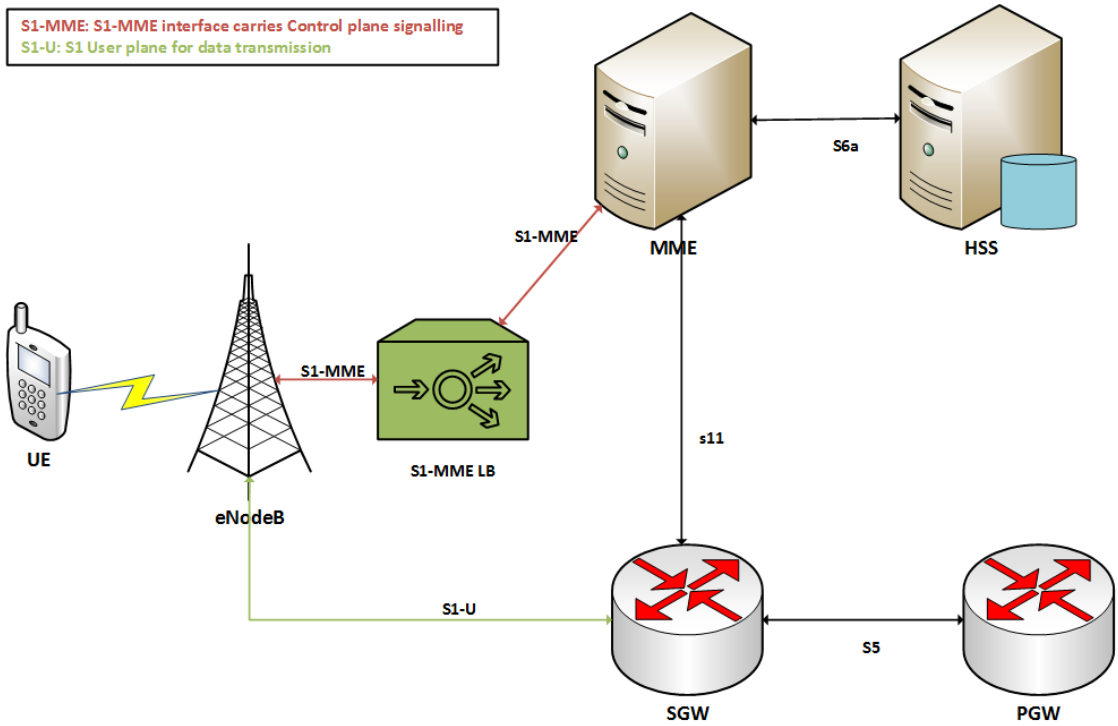


Figure 4.1: Load Balancer Integration in EPC

S1-MME LB enables transparent redirection of MME signalling. As shown in figure 4.1, it is located between a eNodeB and a MME. The usual communication between access network and the core network happens through S1AP protocol. S1-MME is used for

signalling between eNodeB and MME, while S1-U carries the data plane traffic between eNodeB and SGW. S1-MME LB deals with the control plane traffic on the S1-MME layer. S1-MME LB balances the incoming s1 traffic by maintaining a session mapping that determines to which MME the message has to be forwarded to. It maintains the affinity to an MME on a transaction level, meaning that the load balancer forwards the messages from a specific UE always to the same MME.

4.2 S1-MME Load Balancer Design

Figure 4.2 depicts the design of S1-MME load balancer from its protocol standpoint along with possible future extensions for it. S1-MME Load Balancer functionality works on top of the S1AP layer. The load balancer acts as a proxy node which transparently adds itself as representative of the correspondent parties in a back-to-back fashion, meaning that an MME sees the load balancer as eNodeB while the eNodeB sees the S1-MME LB as MME.

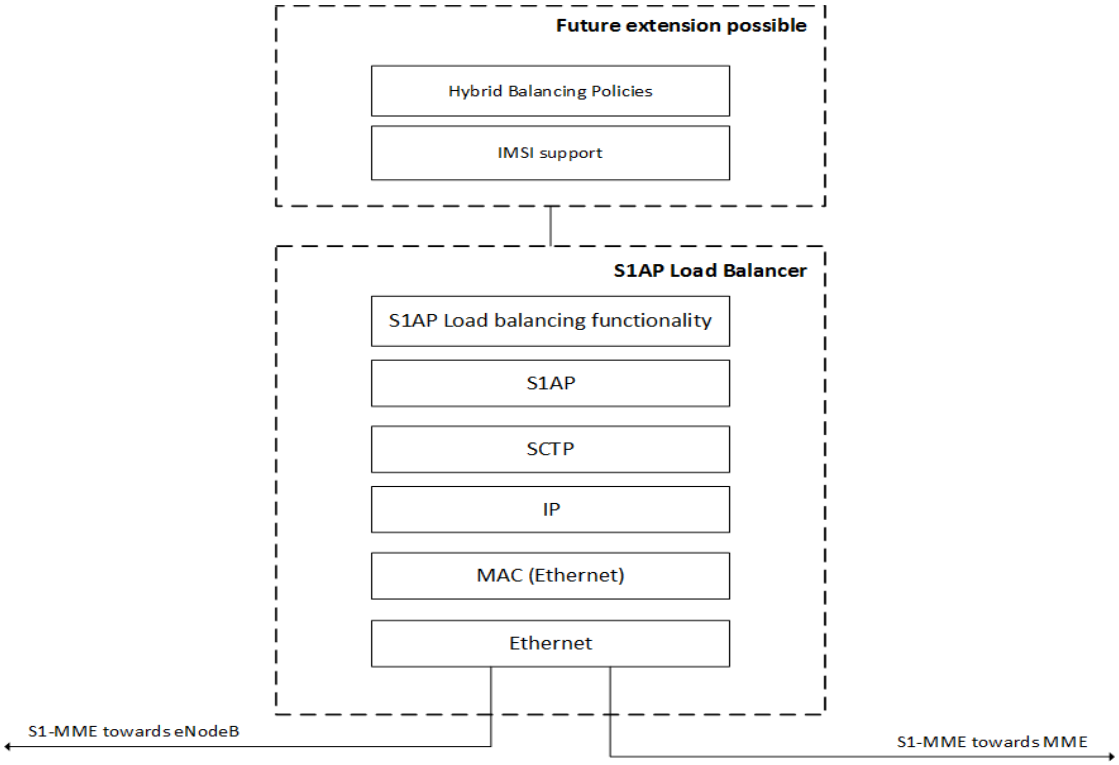


Figure 4.2: S1-MME Load Balancer Design

For now, a simple round-robin balancing policy is used to decide where to forward the incoming s1ap messages. But, a session map is maintained in order to persistently forward the S1 messages belonging to the same UE to the same MME, as serviced for the predecessor messages. Thus choosing an MME in the current design is straightforward, while this in future can be extended to support some hybrid balancing policy such as load on MME, the latency involved in communicating with the MME etc. As of now, all the subscribers connected to the same eNodeB will be addressed by the same MME, but this can be in future extended to distribute the load from the different types of subscribers (general mobile users, M2M, enterprise subscribers) to a different MME, as each of these subscribers have a different need.

Figure 4.3 depicts how a S1-MME load balancer communicates with eNodeB and MME back to back at protocol level.

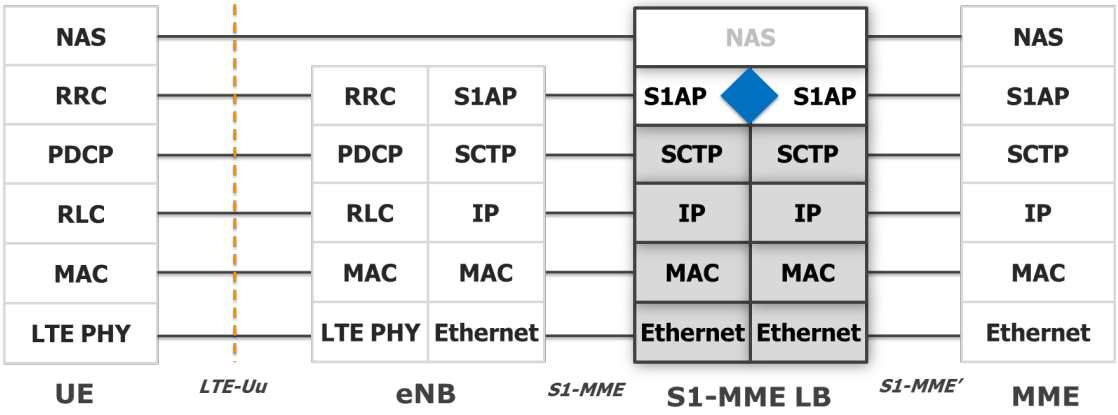


Figure 4.3: S1-MME Protocol Stack

For the lower layer protocols (Ethernet, MAC, IP and SCTP) a back-to-back solution was chosen. Specifically, the data packets received from the eNB or from the MME are processed up to the SCTP level inclusively and then forwarded either to the network or to the load balancing component. It is assumed that the processing up to this level does not incur a lot of resources.

On the other side for the Non-Access Stratum (NAS) protocol, the parsing of these messages, although they will give the identity of the subscriber consumes a large amount of resources (and represent more than 30% of time consumed by an MME while communicating over the S1-MME interface). Thus, it was assumed that the NAS protocol part of the message remains unmodified. This is possible only when all the MMEs have a synchronized subscriber state.

Instead, a matching at S1AP protocol level was chosen. S1AP does not include subscriber specific information or any other type of information from which the specific

subscriber identity can be derived. However, the S1AP protocol identifies in a unique manner a transaction which is equivalent to one specific LTE procedure through the usage of the ENB-UE-S1AP-ID which is included by the eNB for each transaction and the MME-UE-S1AP-ID which is added by the MME to the specific transaction. The message flow of each S1AP Message is explained in detail in further sections.

4.3 Interface Specifications

This sections discusses the interface specification of the S1-MME LB along with any other deviations from the 3GPP standard.

4.3.1 S1 Interface Support

S1-MME Load Balancer operates on S1-MME interface balancing the control plane traffic that is exchanged back and forth between eNodeB and MME as shown in figure 4.4. The basic procedures supported by the interface include S1AP Setup Request, S1AP Setup Response, S1AP Attach, S1AP Detach and S1AP Handover. Since the Load Balancer follows a standard interface, it allows any network equipment to connect to it irrespective of the vendor as long as they support the S1AP protocol. This kind of design facilitates quick time to market and reduced development costs to those vendors who want take S1-MME LB proof of concept to industry grade product.

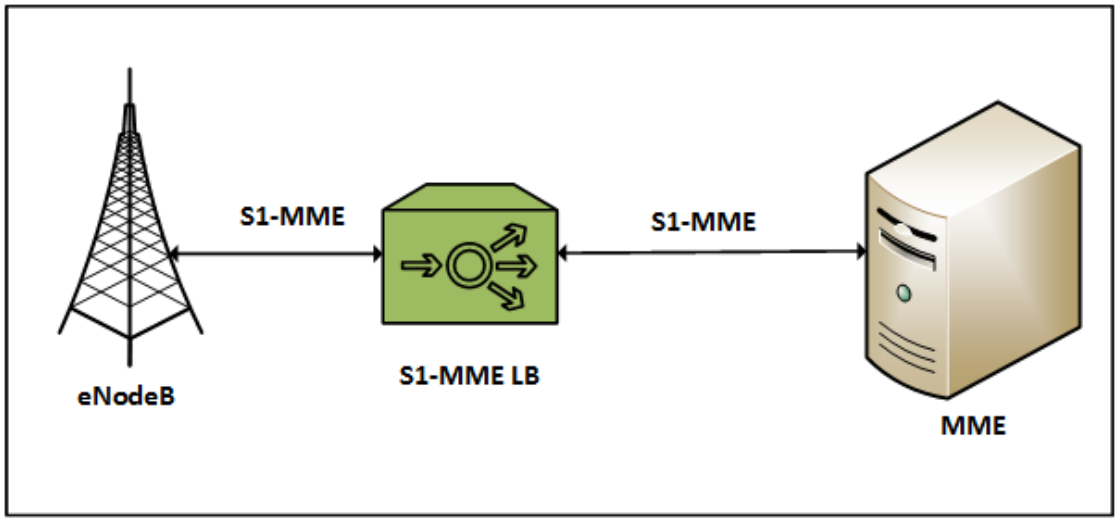


Figure 4.4: S1-MME LB Interface Support

4.3.2 Deviations from 3GPP standard

The deviations from the existing standard design are explained in table 4.1 with reasoning. The second column of the table describes the deviation from standard while the third column explains the reasons why such a deviation from the standard are taken. Apart from the changes in the design did as part of this work, it is also expected that there is another important change that is expected to be taken care as part of the MME state synchronization work. It is to make sure that all the MMEs are in sync with each other and generate MME-UE-S1AP-IDs that are unique across all the MMEs in the system. This way, it makes the S1-MME LB simpler.

Serial No	Deviation	Reason
1	The field enbname received in the s1 setup request message by the MME is used as mmename in its s1 setup response message	• **Without S1-MME LB:** eNodeB and MME are connected to each other directly on S1 interface and they identify their end points uniquely with the ip address and port no. • **In presence of S1-MME LB:** Since a load balancer is introduced in between these two, the MME replies to Load Balancer and Load Balancer should have some unique field to identify where the S1 Setup Response has to be forwarded to • **Solution:** Hence, MME uses same name as enbname so that the LB knows to which eNodeB it has the forward the S1 Setup Response to as it has a enbname to enb-ip address mapping
2	MME uses the pointer field mme-ue-s1ap-id-2 to store the enb-ue-s1ap-id that it has received during the S1 Handover Required message	• **Without S1-MME LB:** In the absence of S1-MME LB, When the MME Receives a Handover Required message from source eNodeB, MME sends a Handover request message to target eNodeB directly. • **In presence of S1-MME LB:** But, in the presence of the S1-MME LB, MME sends the reply to it instead • **Solution:** Pointer mme-ue-s1ap-id-2 which contains the enb-ue-s1ap-id will be useful for S1-MME LB as it has a mapping between enb-ue-s1ap-id and target eNodeB id

Table 4.1: Deviations from 3GPP standard

4.4 S1AP Message Handling

This sections gives an overview of how the supported S1AP messages are handled. Different fields of the S1AP message are used for redirection of MME signalling. As a characteristic requirement of the S1-MME LB, it is expected to send messages from a subscriber to always the same MME. Hence It is required that the S1-MME LB maintains some mappings between different fields of S1AP message to eNodeB ip and MME ip address, so that it knows where to forward the message to. The following sections describe how the S1AP messages are handled in eNodeB to MME direction and MME to eNodeB direction.

4.4.1 S1AP Message handling from eNodeB to MME direction

The following figure 4.5 describes how a message flows from eNodeB to MME.

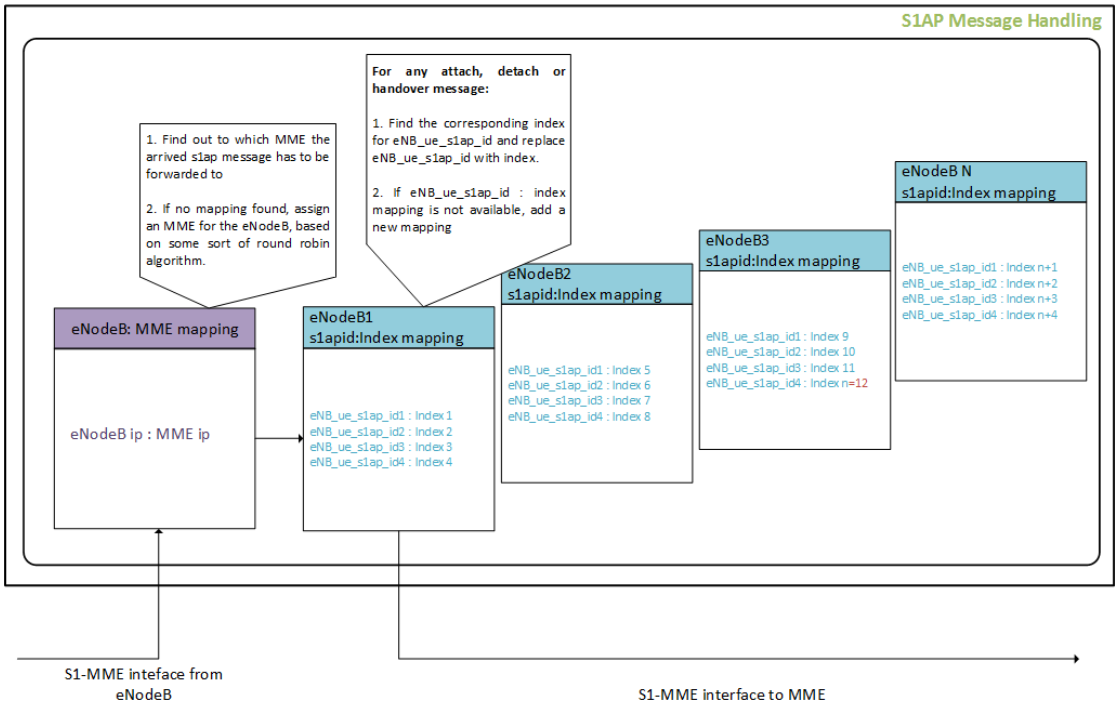


Figure 4.5: S1AP Message Handling from eNodeB to MME

Whenever a message arrives at the S1-MME LB, it needs to identify for a given eNodeB, which MME has to be chosen, if it is from a new MME, a round robin like scheduling algorithm is run to decide to which MME the messages from a specific eNodeB has to be forwarded. After the assignment, make an entry in the session mapping between eNodeB to MME. For the next round of messages coming from the same eNodeB, look for the already available eNodeB to MME mapping based on enbname. This applies

to S1 Setup Request/Response messages, which doesn't have any subscriber related information for mapping. The message for each transaction are uniquely identified by an identifier called eNB-UE-S1AP-ID. But this value is unique for all transactions originated from a given eNodeB. Hence, It is possible that several eNodeBs can use the same eNB-UE-S1AP-ID for serveral of it's transaction. since they generate these ids independently. Hence the S1-MME LB, using a combination of eNB-UE-S1AP-ID and eNodeB ip address, generates a unique index. Index serves as a duplicate eNB-UE-S1AP-ID and can be termed as a new identifier called eNB-UE-S1AP-ID". S1-MME LB replaces eNB-UE-S1AP-ID with this value while forwarding the messages to the MME. The S1-MME LB will now maintain a mapping between eNB-UE-S1AP-ID" and eNB-UE-S1AP-ID. Here afterwards, for the other uplink transport messages for the same UE that travel from eNodeB to MME, the S1-MME LB simply looks for the above stated mapping and replaces the eNB-UE-S1AP-ID with eNB-UE-S1AP-ID". All the set of messages that contains the same eNB-UE-S1AP-ID are considered to belong to a same transaction and hence assigns the same MME, instead of assigning it a new MME.

4.4.2 S1AP Message handling from MME to eNodeB direction

Figure 4.6 explains the message handling from MME to eNodeB direction.

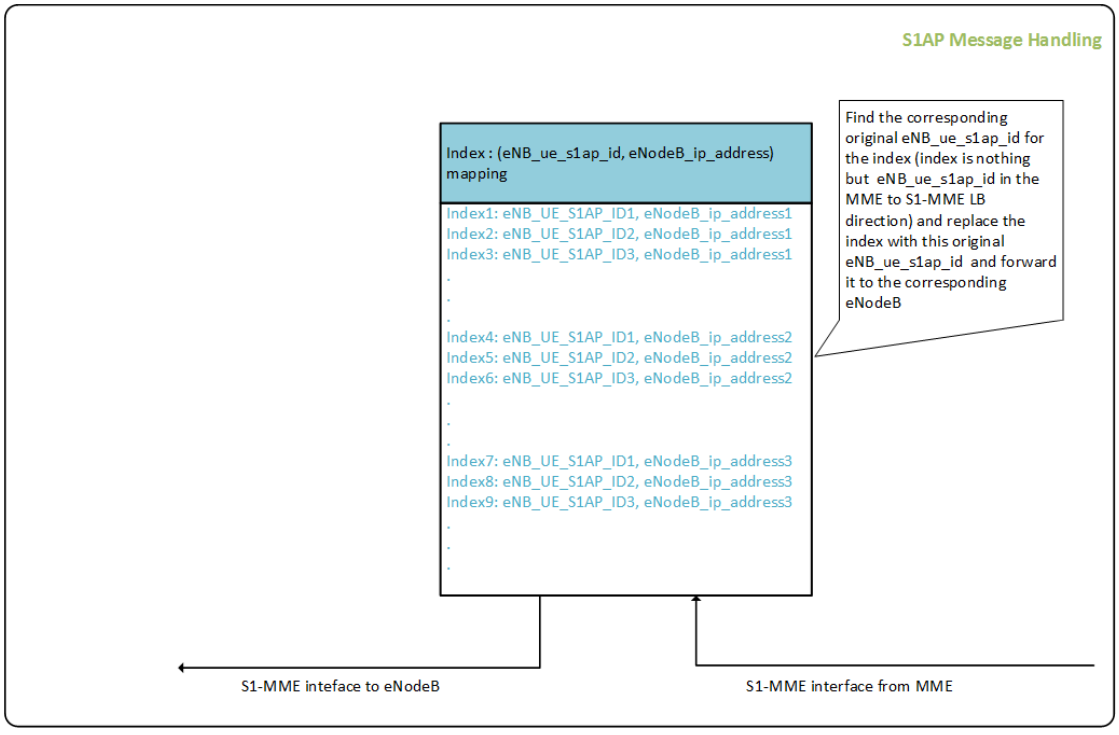


Figure 4.6: S1AP Message Handling from MME to eNodeB

Note that index values in the above figure are nothing but the unique identifier eNB-UE-S1AP-ID" given by the S1-MME while the messages are travelling from eNodeB to MME.

Any s1ap message going from MME to eNodeB will have index value (eNB-UE-S1AP-ID") when arrived at the S1-MME LB. Since this value is not the original s1ap-id given by eNodeB, the Load Balancer has to map it back to its original value in order for the eNodeB to identify the user session properly. The index value (eNB-UE-S1AP-ID") received in the s1ap message is mapped back to the actual s1ap index i.e., eNB-UE-S1AP-ID from the mapping table as described in figure 4.6

4.5 S1AP procedures

This section describes the detailed flow of all the procedures such as S1AP Setup, S1AP Attach, S1AP Detach, S1AP Handover.

4.5.1 S1AP Setup

The following section explains how both S1 Setup Request and S1 Setup Response messages are dealt by the S1-MME LB.

4.5.1.1 S1AP Setup Request

As shown in figure 4.7, a eNodeB sends a S1 Setup Request message to the MME in order to connect to the Core Network. In this since eNodeB assumes the S1-MME LB as the MME, it sends the request to the LB. The message includes details like global_enb_id and enbname. Based on the balancer placing rules, LB assigns eNodeB a specific MME. A mapping between enbname to mme_ip address is maintained on the LB, so that this will be useful for doing a reverse mapping for the replies coming from MME to eNodeB direction. Along with this information, it also includes a mapping between global_enb_id to enb_ip address as this information is needed by the load balancer while handling the Handover messages, which will be seen in detail in further sections.

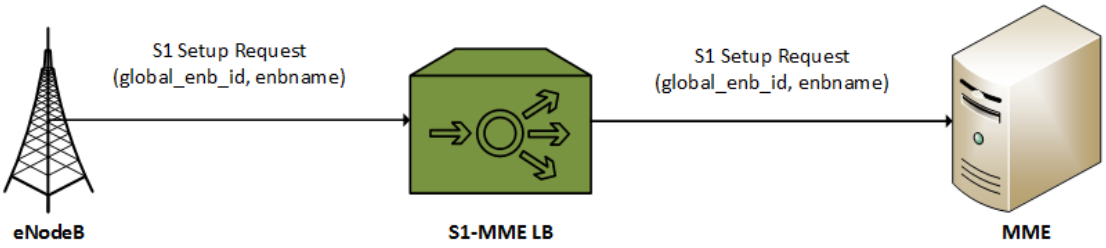


Figure 4.7: S1AP Setup Request

4.5.1.2 S1AP Setup Response

An MME on reception of S1 Setup Request as showin in figure 4.8, will respond to the S1-MME LB with an S1 Setup Response message. In the general case, where there is no LB, MME replies directly to the eNodeB, but in this case, it responds to the LB. This in itself is not sufficient for the LB to forward the message to the right eNodeB. The LB requires some unique identifier in the S1 Setup Response message that can help it to look in the available mapping tables to determine to which eNodeB this S1 Setup Response message has to be forwarded to. For this reason, the MME copies the enbname to mmename while sending the S1 Setup Response message.

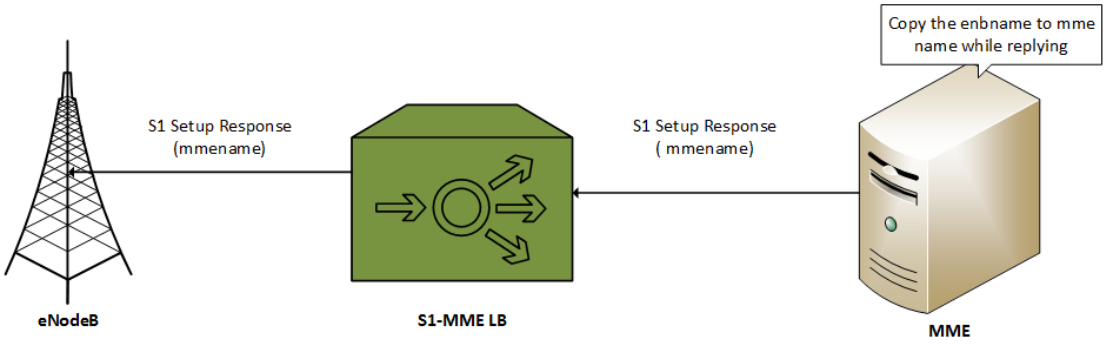


Figure 4.8: S1AP Setup Response

4.5.2 S1AP Attach

This section describes the LTE Attachment procedure when an S1-MME Load Balancer is used. The procedure follows the LTE attachment from ETSI standard 3GPP TS 23.401 [3GP13b] and details the S1AP message information from ETSI standard 3GPP TS 36.413 [3GP13e]. The complete sequence of messages can be seen in attach summary figures 4.9 and 4.10.

1. When a UE attaches to the network, it sends to the eNB a NAS Attach Request message. The message is encapsulated by the eNB into an S1AP: Initial UE Message and it is send to the S1-MME LB. The message includes as identifier an eNB-UE-S1AP-ID field.

2. The S1-MME LB receives the message from the eNB. The eNB identity can be derived from the source IP address of the message together with the eNB-UE-S1AP-ID identify in a unique manner. However, the S1AP identifier is unique only per eNB. The S1-MME LB creates a new unique session identifier for all the eNBs named eNB-UE-S1AP-ID", replaces the previous identifier in the message and sends it to the selected MME. The MME is selected based on a weighted round robin algorithm at transaction level.

3. When the MME receives the response, it transmits a NAS Authentication Request towards the device. The message is included into an S1AP: Downlink NAS Transport message and includes the eNB-UE-S1AP-ID" previously received from the S1-MME LB and a newly generated MME-UE-S1AP-ID. The MME is aware only of the S1-MME LB as a proxy for the eNB, thus it sends the response back (i.e. the S1-MME LB is a proxy for the eNBs making them transparent to the MME). This message is conformant to the 3GPP standard.

4. Based on the eNB-UE-S1AP-ID" the S1-MME LB determines that the message pertains to the specific UE attachment transaction. It replaces the eNB-UE-S1AP-ID" with the eNB-UE-S1AP-ID previously received from the eNB and sends it to the appropriate eNB. The eNB transfers the NAS message to an RRC: DL Inf. Transfer message.

5. The UE responds with a NAS Authentication Response included into an RRC: UL Inf. Transfer. The eNB transfers the message into an S1AP: Uplink NAS Transport message in which the session is identified by the eNB-UE-S1AP-ID and the MME-UE-S1AP-ID and sends it to the S1-MME LB.

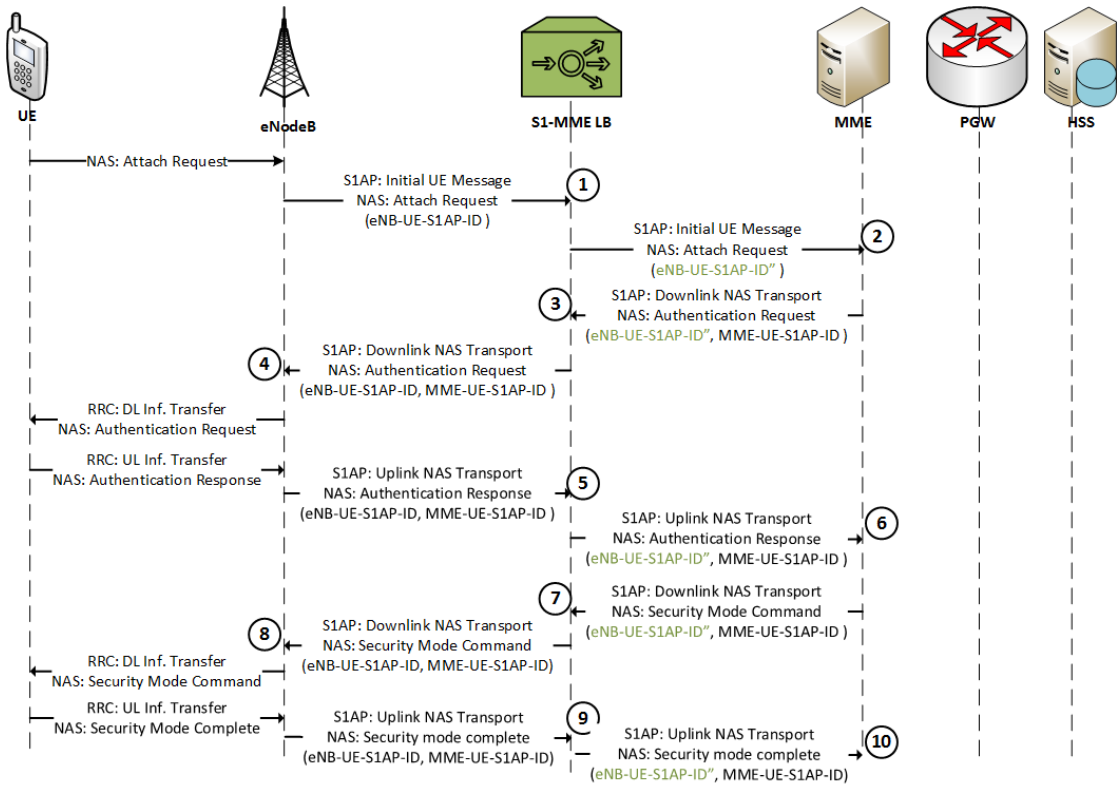


Figure 4.9: LTE Attachment with S1-MME LB part 1

6. The S1-MME LB replaces the eNB-UE-S1AP-ID with the eNB-UE-S1AP-ID" and

sends the message to the MME selected in Step 1. This ensures that all the messages of the same transaction are sent to the same MME.

7. The MME responds with a NAS Security Mode Command including the security credentials request based on the information received from the HSS. The message is sent to the S1-MME LB. Similarly to Step 2, this is a standard message, not requiring any modifications of the MME.

8. The S1-MME LB replaces the eNB-UE-S1AP-ID" with the eNB-UE-S1AP-ID and sends the message to the eNB. The eNB transfers the message into an RLC DL Inf. Transfer and sends it to the UE.

9. The UE responds to the MME with a NAS: Security Mode Complete message which is received by the S1-MME LB from the eNB in an S1AP: Uplink NAS Transport message. The S1-MME LB is processed in the same way as Step 5.

10. The MME receives the message from the S1-MME LB with a modified UE-eNB identifier which it processes according to the standard. Following, the MME executes the rest of the EPC attachment procedure including the Update Location Request/Acknowledgement messages exchanged between the MME and the HSS and the chained Create Session Request/Response between the MME, SGW and PGW. At the end of these steps, the location is updated into the HSS and a data path through the core network is established.

11. The MME responds to the UE with a NAS: Attach Accept message which is included into an S1AP: Initial Context Setup message identified by the eNB-UE-S1AP-ID" and the MME-UE-S1AP-ID.

12. The S1-MME LB identifies the transaction based on the eNB-UE-S1AP-ID", replaces it with the eNB-UE-S1AP-ID received in the first message from the eNB and sends it to the eNB. In its turn the eNB encapsulates the NAS message into an RRC: UL Inf. Transfer and sends it to the UE.

13. The eNB responds to the S1-MME LB with an S1AP: Initial Context Setup Response identified by the eNB-UE-S1AP-ID and the MME-UE-S1AP-ID.

14. The S1-MME LB identifies the transaction based on the source of the message as well as on the eNB-UE-S1AP-ID, replaces it with its own ip address and eNB-UE-S1AP-ID" respectively and sends it to the MME.

15. The UE responds to the MME with a NAS Attach Complete which is encapsulated by the eNB into an S1AP: Uplink NAS Transport message and sent to the S1-MME LB.

16. The S1-MME LB identifies the transaction, replaces the eNB-UE-S1AP-ID with the eNB-UE-S1AP-ID" and sends the message to the MME. The transaction can be removed from the binding data base as it is completed. The same identifiers

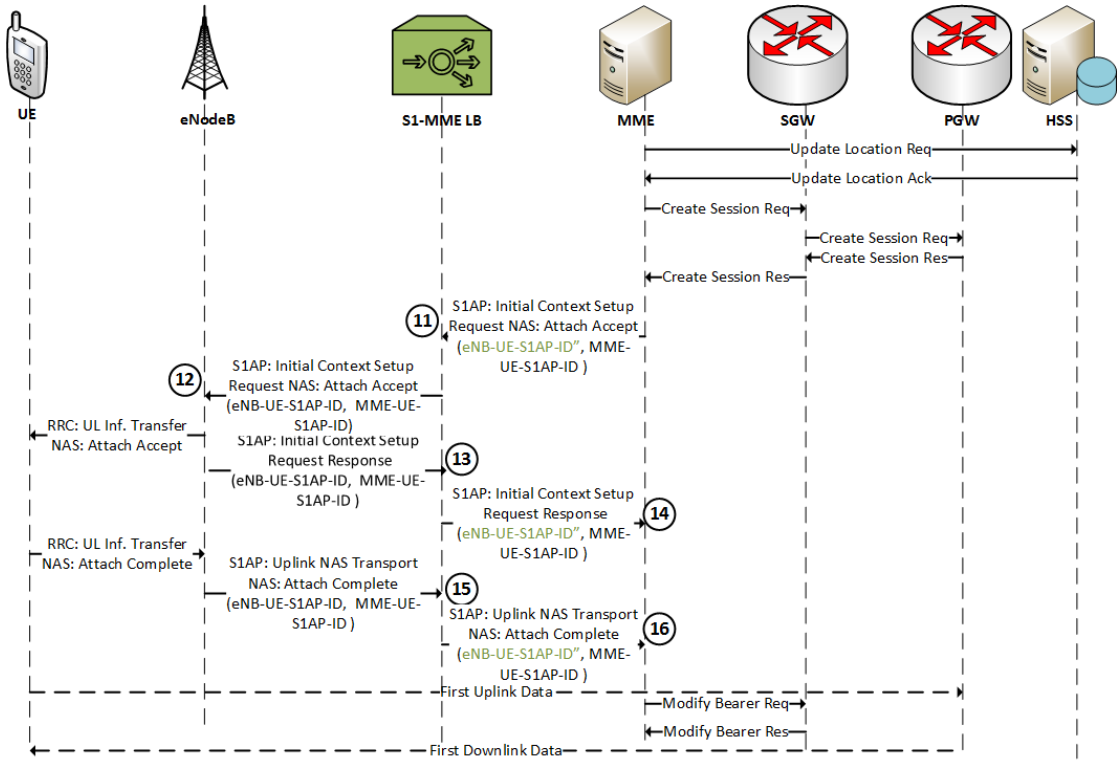


Figure 4.10: LTE Attachment with S1-MME LB part 2

will not be used by any other transaction. The MME completes the attachment by executing the Modify Bearer Request GTP procedures with the SGW.

4.5.3 S1AP Detach

In this section, the LTE detachment procedure considering the S1-MME LB is described. The detachment procedure can be triggered by a NAS: Detach Request message sent by the UE to the MME.

1. The NAS: Detach request message is encapsulated by the eNB into an S1AP: Uplink NAS Transport message including a newly generated eNB-UE-S1AP-ID

2. The S1-MME LB will create a new transaction for the message. The transaction will be identified by the eNB-UE-S1AP-ID and the eNB identifier. The S1-MME LB generates a new eNB-UE-S1AP-ID" which is uniquely identifying the system. The S1-MME LB selects an MME based on a selection algorithm such as weighted round robin. The message is forwarded to the selected MME.

3. The MME executes the detachment procedure including the chained GTP Delete Session Request/Response with the SGW and PGW, resulting in the removal of the

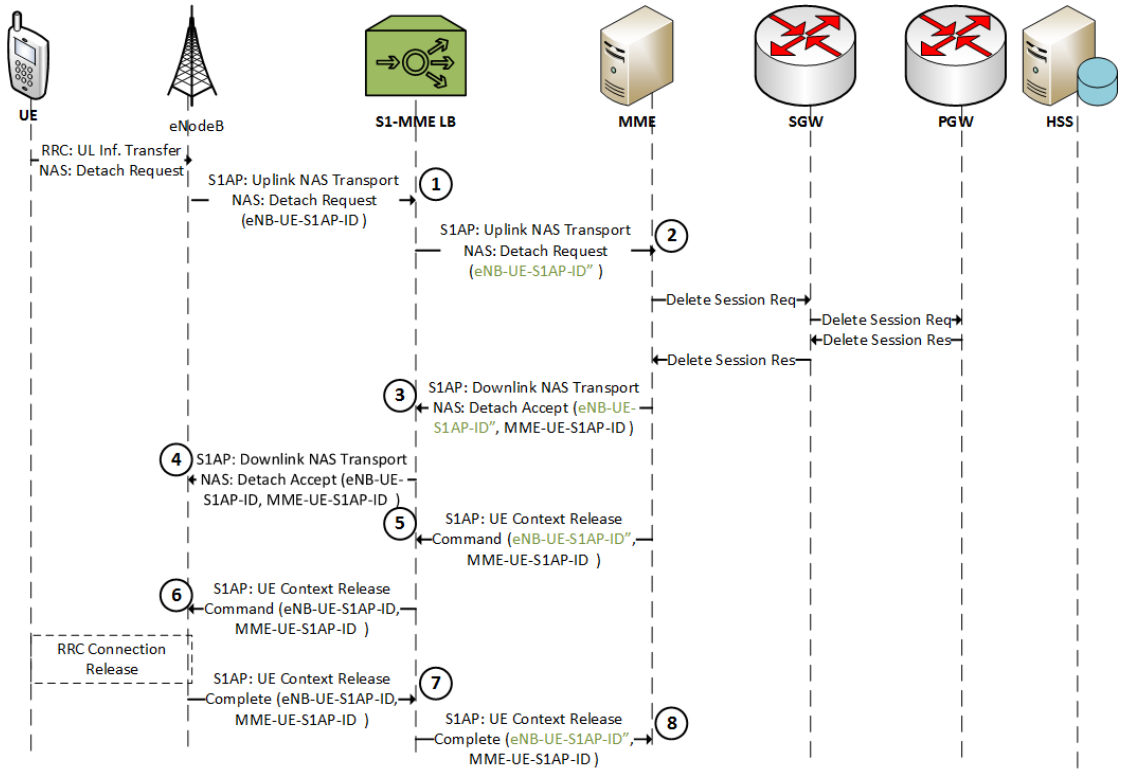


Figure 4.11: LTE Detach Procedure

data path. Upon the success of the procedure, the MME sends a standard S1AP: Downlink NAS Transport message which includes the NAS: Detach Accept.

4. The S1-MME LB identifies the session based in the unique eNB-UE-S1AP-ID", replaces it with the eNB-UE-S1AP-ID and sends it to the source eNB

5. The MME sends consecutively an S1AP: UE Context Release to the S1-MME LB.

6. The MME executes the same operation as in Step 4.

7. The eNB executes the RRC Connection release. When it is completed, the eNB sends to the S1-MME LB an S1AP: UE Context release message.

8. The S1-MME LB matches the specific transaction, sends the appropriate S1AP: UE Context release message and removes the information related to the transaction.

4.5.4 S1AP Handover Procedure

In this section the S1 LTE Handover Procedure is described from the perspective of the S1-MME LB. The UE is communicating through the source eNB by exchanging uplink and downlink data packets through the source eNB to the SGW through a GTP tunnel.

The RRC: Measurement Control from the source eNB triggers an RRC: Measurement Report from the UE from which the source eNB determines that a handover has to be executed. This acts as a trigger for the Handover procedure.

The following steps summarizes the sequence of steps involved during S1AP Handovers:

1. The source eNB sends an S1AP: Handover Required message to the S1-MME LB which includes a newly generated eNB-UE-S1AP-ID identifier as well as target eNB related details.

2. The message is received by the S1-MME LB which determines that it is a new handover transaction. It creates a new transaction identification binding which includes the eNB-UE-S1AP-ID, the source eNB identity determined from the IP address from which the message was sent and the target eNB details. Additionally, it generates a system unique eNB-UE-S1AP-ID" identifier. The eNB-UE-S1AP-ID is replaced in the message with the eNB-UE-S1AP-ID". The S1-MME LB sends the message to MME that was taken into service for the eNodeB from where this handover message is originated.

3. The MME receives the S1AP: Handover Request. It determines based on its content the global identity of the target eNB and attempts to send the message to it. However, from the perspective of the MME, all the eNBs are represented by the S1-MME LB component. The MME will send an S1AP: Handover Request to the S1-MME LB. Standard wise, the message will include a newly generated MME-UE-S1AP-ID-Tgt. However, in order to be able to determine to which transaction the message pertains to, the MME has to add to the message the received eNB-UE-S1AP-ID". This can be done using the optional message field *MME-UE-S1AP-ID2 in the S1AP Handover Request message.

4. The S1-MME LB determines to which transaction the message pertains to based on the eNB-UE-S1AP-ID" which was included in the field *MME-UE-S1AP-ID2 by the S1AP: Handover Request. By checking the target eNB information, it determines the identity of the target eNB. Note that currently this operation is executed in the MME and if the standard conformity is maintained for the MME, it results in a procedure duplication. The S1AP: Handover request message is sent to the target eNB including the MME-UE-S1AP-ID generated by the MME.

5. The target eNB responds with an S1AP: Handover request acknowledge including a new eNB-UE-S1AP-ID generated by the target eNB. The message is sent to the S1-MME LB.

6. The S1-MME LB matches the message with the specific handover transaction based on the mapping between IP address of the target eNB and the MME to which it is connected to. It generates a new eNB-UE-S1AP-ID" which is replacing the identifier received from the target eNB. The message is sent to the MME.

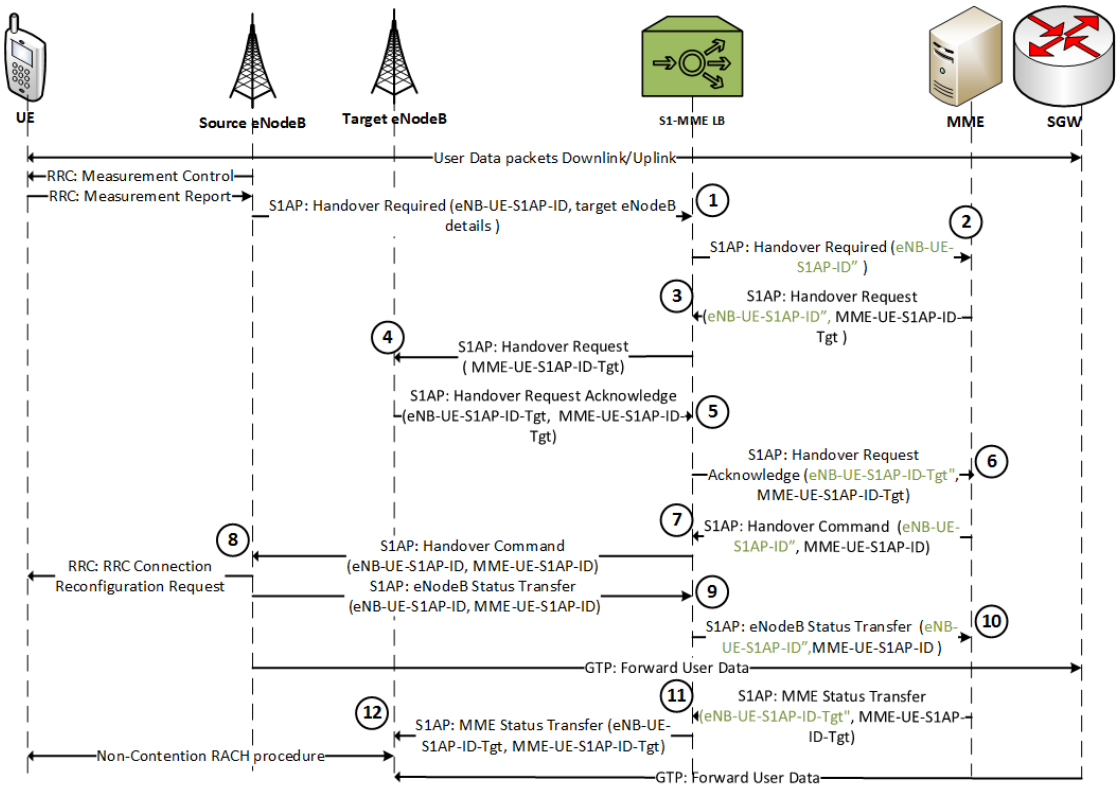


Figure 4.12: S1AP LTE Handover Procedure (Part 1)

7. The MME responds with a standard S1AP: Handover Command which include the eNB-UE-S1AP-ID" and the MME-UE-S1AP-ID.

8. Based on this information, the S1-MME LB is determining the specific transaction, replaces the eNB-UE-S1AP-ID" and sends it to the appropriate source eNB. This results into an RRC reconfiguration request.

9. The eNB sends to the S1-MME LB an S1AP: eNB Status Transfer

10. The S1-MME LB sends to the MME the message on the eNB Status Transfer.

11. The MME responds with an S1AP: MME Status transfer to the target eNB. Instead the message is sent to the S1-MME LB.

12. The MME sends the Status Transfer message to the target eNB.

13. The handover is executed and completed at radio level. The success of the procedure is notified to the target eNB as an RRC Connection reconfiguration complete. Following, the target eNB is sending an S1AP Handover Notify message to the S1-MME LB identified by the eNB-UE-S1AP-ID-Tgt. And the MME-UE-S1AP-ID-Tgt).

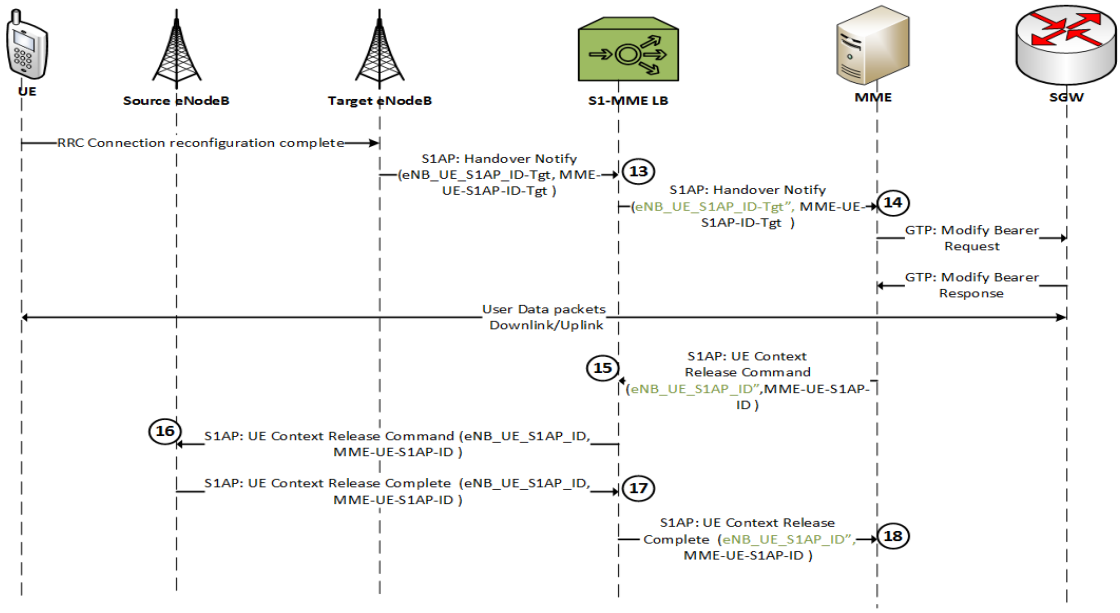


Figure 4.13: S1AP LTE Handover Procedure (Part 2)

14. The S1-MME LB identifies the transaction and forwards to the message to appropriate MME. The MME executes the GTP Modify Bearer Request/Response with the SGW, resulting into a new end-to-end data path through the target eNB.

15. The MME completes the procedure by sending towards the source eNB an S1AP: UE Context Release Command.

16. The S1-MME LB sends the message to the source eNB.

17. The source eNB responds with an S1AP: UE Context Release Complete message.

18. The context release is forwarded to the appropriate MME. The S1-MME LB can now safely remove the transaction information, as there are no message to be exchanged and as the identifiers will not be again used.

4.6 Conclusion

This chapter discussed several design aspects of the S1-MME LB that was considered for implementation as part of this thesis work. It focused on design issues for handling each type of S1AP message such as S1AP Setup, Attach, Detach and Handovers at the S1-MME LB. The design is made to ensure that S1-MME LB is compliant to the 3GPP standards. Furthermore, the design considered to keep the S1-MME as simple as possible, meaning that it parses very few parameters to decide to which MME the MME signaling messages has to be forwarded in a persistent manner. This approach allows the S1-MME LB to introduce very less delay latency in the path between eNodeB

and MME during the user attach and detach operations. One of the major decisions is to ensure that load balancer is transparent to eNodeB and hence it is not necessary to inform the access network any changes in the core network. But, MME knew the existence of the load balancer, in the core network. Two minor changes are made in the design of MME to accommodate the presence of S1-MME LB in the EPC network as discussed in the table 4.3.2. These changes allow the S1-MME LB to handle the S1 Setup and Handover procedures properly in the presence of the S1-MME LB in between eNodeB and MME. Though changes are done to the MME, it is ensured that no new elements are introduced to the 3GPP Standard S1AP message strucuture, thus making the implementation still adhering to the 3GPP standards.

5 Implementation

This chapter discusses the implementation aspects of S1-MME Load Balancer (S1-MME LB). The first sections describes the environment in which the project is realized, followed by brief introduction to the tools and softwares used. Furthermore, emphasis is given on explaining the internal relations between several modules, along with the details of functionality performed by each module. Several other important implementation aspects like the programming framework and other tools used are discussed. This sections also discusses the virtualization platform used to validate the software network functions implemented. Focus is given also on discussing the configuration management and command line support provided for the S1-MME LB.

5.1 Environment

As part of the thesis work, the software functions of S1-MME LB are developed in C programming language using wharf framework. Wharf provides easy to use APIs to achieve the required functionality. The details of wharf framework are discussed in further sections. The software implemented runs on a commercial off-the-shelf hardware running linux operating system. There is no need for high-end commercial hardware to run the software functions developed. They simply run on virtual machines running on a linux machine as virtualized software components.

Apart from implementing some core network functions using the c programming language, some scripts written in bash and Python are used to automate the system configuration such as different network settings that deals with DHCP, DNS and routing. In addition to this, some configuration files are written in Extensible Markup Language (XML) that are used to configure the S1-MME LB.

The following are the list of important softwares and tools used during the process of realizing the S1-MME LB as a software network function:

- **Ubuntu 14.04.3 LTS:** One of the most widely used open source linux distribution is used as a platform to run the software components. To be specific, the S1-MME LB is run as a software network function on Ubuntu 64-bit distribution with kernel version 3.13.0-40-generic.
- **Eclipse C/C++ Development Tool (CDT):** During the project, Eclipse Luna (Release 1) with CDT plugin is used as an Integrated development environment.

CDT supports development in programming languages C and C++, with the following support:

1. Creation of project
2. Build Management
3. Code editor with syntax highlighting
4. Source Navigation

- **GNU Compiler Collection (GCC):** A compiler supporting various programming languages that includes C and all its associated libraries. It's sources are free and available to download. Hence GCC is used as the compilation environment for this project.
- **VMWare Workstation version 10:** It is a virtualization software that allows to run multiple virtual machines on a physical machine. A licensed version of VMWare workstation version 10 is used to run instances of LB and other elements of EPC during Evaluation.
- **SVN versioning tool:** Subversion is an open source Version Control System (VCS). It is used for creating, editing and easy maintenance of the files and directories, during the course of the project development. Hence, It is used for maintaining the code developed for S1-MME LB.

5.2 Overview of Project Structure

The Project structure of S1-MME LB implementation follows the Wharf framework. Wharf consists of several modules. Each of these modules delivers some specific functionality. S1-MME LB is designed as a new module in wharf. Its functionality is placed in a module called s1ap_loadbalancer which sits just above the s1ap module in the protocol stack.

s1ap_loadbalancer is implemented as a wharf module and devloped in a such a fashion that it reuses the existing functionality provided by the already existing module s1ap and adds the required new functionality on top of it to serve the purpose of this project. The implementation follows a design that ensures the new module s1ap_loadbalancer can be reused by another module just like the way it is using the other existing modules in wharf. Also, s1ap_loadbalancer module is split into internal modules, each performing a specific task. This gives the flexibility of adapting changes in these internal modules without the need to worry about the impact of changes in one module over the other.

The overview of the project structure can be seen in figure 5.1:

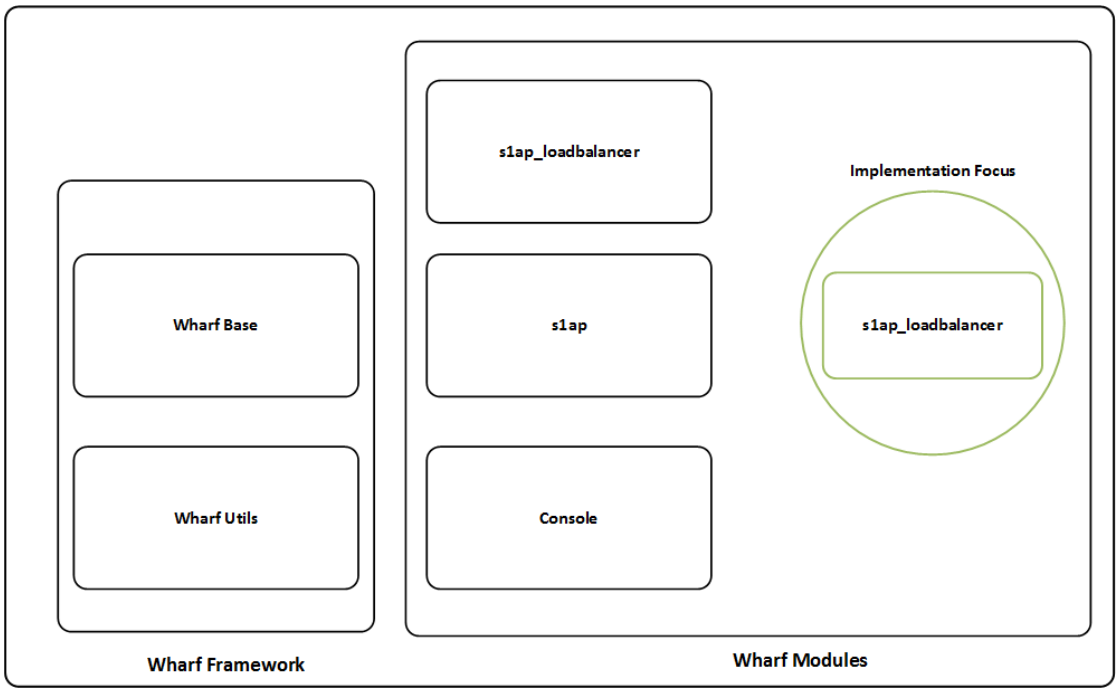


Figure 5.1: Project Structure

s1ap_loadbalancer module uses the existing functionality provided by modules such as s1ap and console, along with other basic wharf modules like base and utils. Each module provides a specific set of functionality.

The functionality achieved by each of these wharf modules are explained below:

- **Wharf base:** It provides core functions such as process management, worker pool management, memory management, timers and schedulers. Wharf base module provides core functions such as s1ap module provides all the signaling procedures between eNodeB and MME.
- **Wharf Utils:** It provides most frequently used data structures and operations, templates, Macros, Locks and Semaphores, Logging, Binary Codec, Lists and Strings etc.

These modules make the implementation of any new module much easier, thus helping to reduce the time to develop a new feature or functionality.

s1ap is an already implemented module that provides all the s1ap procedures. Since, the s1ap procedure handling support already exists, it will be reused.

The main aim of this project is to load balance the signaling messages across several MMEs with an affinity to MME on a transaction level through the usage of the ENB-UE-S1AP-ID. Hence, a new module is introduced in the wharf, that addresses this requirement of balancing the MME signaling traffic. The primary focus of this project is implementing the s1ap_loadbalancer that transparently redirects the MME signaling using the support provided by s1ap to handle the s1ap procedures. Other existing modules that perform the duties of sctp, ip and mac layers are re used by the S1-MME LB. Hence, it will not influence the operations in the lower layers.

5.3 S1-MME Load Balancer components and Operations

Figure 37 lists the sub components of the S1-MME LB functionality. S1-MME is implemented as a module called s1ap_loadbalancer.

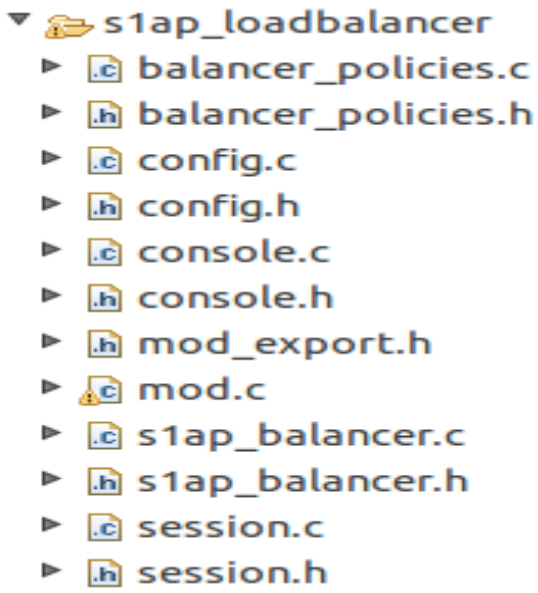


Figure 5.2: Basic components of s1ap-loadbalancer module

The basic components of s1ap_loadbalancer module are:

1. S1AP Load Balancing
2. Session Management
3. Balancer Policing
4. Configuration Management
5. Console Management
6. Module Management

The above stated internal modules (sub modules) perform a specific task, as explained in the sub sections below.

5.3.1 S1AP Load Balancing

This module performs the registration of all the s1ap procedures supported. When a supported s1ap procedure arrives at the S1-MME LB, it calls the remaining internal modules to find out where to forward this message to. More importantly, this module communicates with the session management module and Balancer policing module. It contacts the session management module to find out if there already exists a session that gives the information about where to redirect this message to. It interacts with Balancer policing module, when there are no existing sessions that provides MME redirection information. The details of these internal submodules are discussed in further sections.

5.3.2 Session Management

This module maintains all the sessions required so that the S1-MME LB can redirect the messages to the MME, maintaining affinity at transaction level. This means that S1-MME LB will send the messages to the same MME from a given UE session uniquely identified by a eNB-UE-S1AP-ID. This unique identifier is generated by the eNodeB.

The sessions maintained for mapping between the s1ap ids, i.e, eNB-UE-S1AP-ID and MME-UE-S1AP-ID and the corresponding eNodeB and MME can be seen in the mapping structure s1ap_loadbalancer_session_t in the listing 5.1 below:

Listing 5.1: Main Sesssion for transaction mapping

```
{
typedef struct _s1ap_balancer_session_t {
    uint32_t s1ap_balancer_index;
    str enb_mme_name;
    struct {
        ip_address ipv4;
        ip_address ipv4_target;
        enb_ue_s1ap_id_t  enb_ue_s1ap_id;
    }enb;
    struct{
        ip_address ipv4;
        mme_ue_s1ap_id_t  mme_ue_s1ap_id;
    }mme;
    struct _s1ap_balancer_session_t *next,*prev;
} s1ap_loadbalancer_session_t;
```

Since each eNodeB generates a unique eNB-UE-S1AP-ID on its own, it is possible that two eNodeBs can generate the same value, and therefore this can lead the an ambiguity in identifying a session with it. It is the duty of the S1-MME LB to treat same eNB-UE-S1AP-ID from different eNodeBs uniquely. Hence every eNB-UE-S1AP-ID is replaced with eNB-UE-S1AP-ID" whose value is stored in the variable s1ap_balancer_index, while sending the s1ap message from eNodeB to MME and the vice versa in the opposite direction, as session in the above session mapping structure s1ap_loadbalancer_session_t. While it is possible that two MMEs also can generate the same MME-UE-S1AP-ID, it is ensured that the state sharing mechanism between MME ensures that each MME generates unique set of MME-UE-S1AP-IDs so that there will not be any ambiguity between them. This state sharing work is taken as a separate work independent of the scope of this thesis work.

Also for S1 setup procedures, there is no S1AP-ID in either S1AP setup request/response messages in order to use figure out which MME is assigned to which eNodeB. Hence a session mapping is maintained between enb_name to the ip address of eNodeB and MME as explained in the listing 5.2. MME has to ensure that it declares its name, same as that of enb_name. A single field called enb_mme_name is used in the session map, as both eNodeB and MME uses the same name during their communication with each other, so that the S1-MME LB knows to which eNodeB the S1 setup reply messaged has to be forward to.

Listing 5.2: eNodeB to MME Session mapping

```
typedef struct _s1ap_enb_mme_mapping_session_t {
    uint32_t enb_mme_ses_map_index;
    uint32_t global_enbid;
    str enb_mme_name;
    struct {
        ip_address ipv4;
    }enb;
    struct{
        ip_address ipv4;
    }mme;
    struct _s1ap_enb_mme_mapping_session_t *next,*prev;
} s1ap_loadbalancer_enb_mme_mapping_session_t;
```

5.3.3 Balancer Policing

This module takes care of assigning an MME to a new eNodeB taken into service. For the scope of this project, this part of the work is kept simple by following a simple round-robin like approach, maintaining affinity to MME at a transaction level.

Usually the S1-MME LB is fed with the list of available MMEs available in the core network that can address the requests coming from the subscribers. When a new eNodeB sends any requests to the S1-MME LB, assuming it to be MME, the S1AP load balancing sub module communicates with the Balancer policing sub module, which will inturn assign a new MME. It now redirects the message to MME, after making appropriate modifications to fields such as source ip address and the eNB-UE-S1AP-ID.

Selection of MME is usually in a round-robin fashion as explained in listing 5.3. It keeps assigning an MME in a sequential format from the list of available MMEs, while it ensures that it skips the current MME in the list if it is not responding at the time of assignment and continues with the next MME in the list till it finds a working MME. If it arrives at the end of the list of MMEs, it starts again assigning MMEs from the beginning of the list as shown in the short pseudocode. The short pseudocode focuses on explaining the parts of code that assigns an MME from list by iterating the through the list of available MMEs from the MME list.

Listing 5.3: S1-MME round-robin like balancing

```
int s1ap_loadbalancer_round_robin_policy(mme_addr, num_of_tries)
{
    /*It might so happen that the previously assigned MME
     might not be reachable and hence there might arise
     a situation where new MME has to be assigned, until
     unless the list is exhausted (reaches max size of
     list) */

        /*Check if the S1-MME LB has already tried assigning
        all the MMEs to an eNodeB in the list.*/
    if(num_of_tries == max_size_of_mme_list){
        return 0;
    }

//Assign an MME from the list of MMEs
    *mme_addr = mme_list_t->head->mme_addr.ipv4;
    //Track the no of eNBs using a particular MME
    mme_list_t->head->users +=1;
    //To keep track of the mme list
    mme_list_t->count += 1;

    /*
      Point to the beginning of the list, if all ip
      addresses in the list are already assigned
      (round robin)  else Just increment the pointer
       to the next location in the list
```

```
     */
    if(mme_list_t->count == mme_list_t->size){
        mme_list_t->head = mme_head_list_t->head;
        mme_list_t->count = 0;
    }
    else{
        mme_list_t->head = mme_list_t->head->next;
    }
    return 1;
}
```

5.3.4 Configuration Management

Extensible Markup Language (XML) is used to set the required configuration parameters for the S1-MME LB. The information in the XML config file as shown in figure 5.3 contains the basic details that are needed to bring the S1-MME LB up and running.

The details in the configuration file includes the ip address of the S1-MME LB, the port on which eNodeB and MME communicates with it. Other details specific to Wharf platform include the memory, number of workers and queues that needs to be allocated by the platform while bringing up the element.

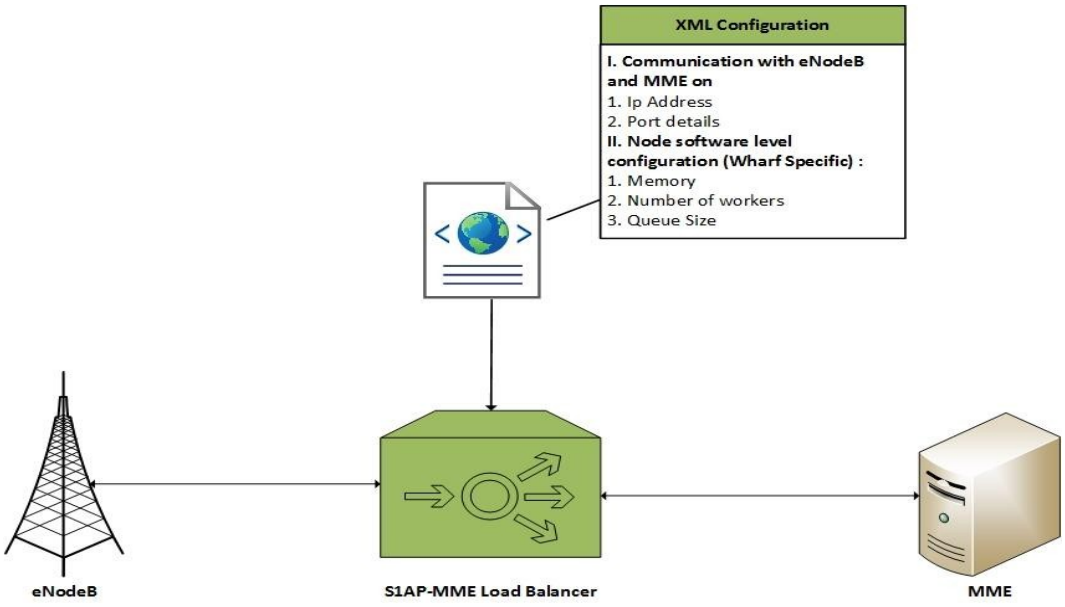


Figure 5.3: S1-MME Load Balancer Configuration

An extract of the S1-MME LB configuration can be seen from the listing 5.4 below:

Listing 5.4: Extract of S1-MME config file

```
<Module binaryFile="modules/s1ap_loadbalancer/s1ap_loadbalancer.so">

                <![CDATA[
                <!--
                 S1AP BALANCER Parameters
    -->
                <WharfBalancer
                        s1ap_balancer_ipv4="192.168.4.70"
                        default_mme_ipv4="192.168.4.80"
                        hash_size="32">
                </WharfBalancer>
                ]]>

</Module>
```

The field s1ap_balancer_ipv4 is an IPv4 address to identify the S1 interface of the S1AP LB that connects to both the eNodeB and MME, while the field default_mme_ipv4 denotes the IPv4 address of the default MME, to which the LB can connect to in case the LB is not fed with any list of MMEs available. hash_size denotes hash size for the hash table named binding cache. Its default value is set to 32. All the newly added sessions identified by s1ap id like eNB-UE-S1AP-ID will be mapped to this hashtable of size 32. If two eNB-UE-S1AP-IDs collide to the same key value, they will follow a separate chaining like technique to resolve the collision.

5.3.5 Console Management

In order for the users to interact with the S1-MME LB, a user interface is provided in the form of a console as shown in figure 5.4.

The console command supports several commands to interact with the S1-MME LB that helps to configure and retrieve other parameters of the node. As, an example, it allows to add new MME elements ip addresses to the balancer, thus allowing it to balance the load across multiple machines. It includes commands that can allow to debug the running node condition like memory allocation information. It also allows to change the amount of debug information printed on the console of the running node. A snapshot of console can be seen in figure 5.4

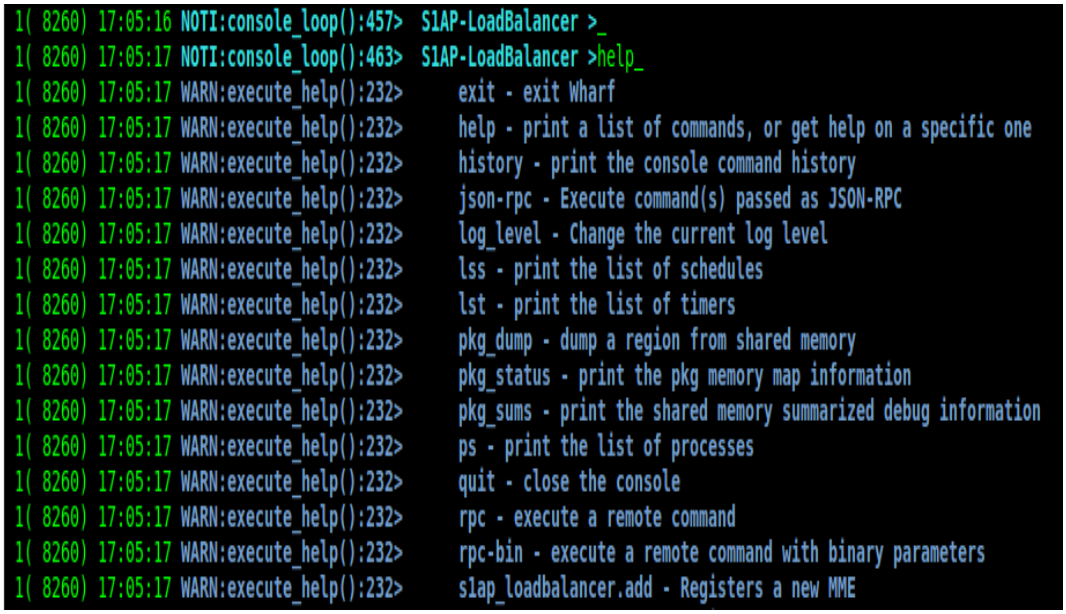


Figure 5.4: Console support for S1-MME Load Balancer

5.3.6 Module Management

Module management deals with initialization and destruction of any external modules needed to support the s1ap_loadbalancer functionality. The external modules needed by the s1ap_loadbalancer are config, console and s1ap. All these modules are already implemented. These modules are integrated with s1ap_loadbalancer module by using module management.

The following are the basic wharf functions that are implemented for bringing the node up and running:

1. **s1ap_loadbalancer_init:** This function will be called at wharf start-up before any forking. Configuration of the S1-MME LB is passed as a parameter to this function.

2. **s1ap_loadbalancer_child_init:** This function is called on every process; this does not only happen at start-up, but also on any process forking. The type of process to be initialized is passed as a parameter to this function.

3. **s1ap_loadbalancer_get_bind:** This function retrieves the bindings of the modules.

4. **s1ap_loadbalancer_destory:** This function will be called on the exit of every process. The type of process to be terminated is passed as an argument to it.

5.4 Virtualization

Virtualization allows to create a virtual version of any resource such as server, network or a storage device. S1-MME LB is implemented as a software function that runs on a virtual platform. It is deployed in the path between eNodeBs and MMEs. The functionality of the impemented system is integrated into EPC environment, where the presence of different network elements like eNodeB, MME, SGW and PGW are needed. As today's service providers are moving towards cloud based infrastructure, it is good to implement the S1-MME LB as a software function so that it can be run on any linux machine irrespective of the underlying hardware. Virtual functions can be scaled out/in easily in the cloud. These EPC elements can be run in different linux machines, but it is difficult for user to handle independent machines especially considering that the configuration of these elements needs to be changed on a need basis. Hence, VMWare Workstation is used to run all these nodes on virtual machines, thus making it easier for the user to manage them. The screenshot of the S1-MME LB running as a virtual instance on VMWare Workstation can be seen in figure 5.5

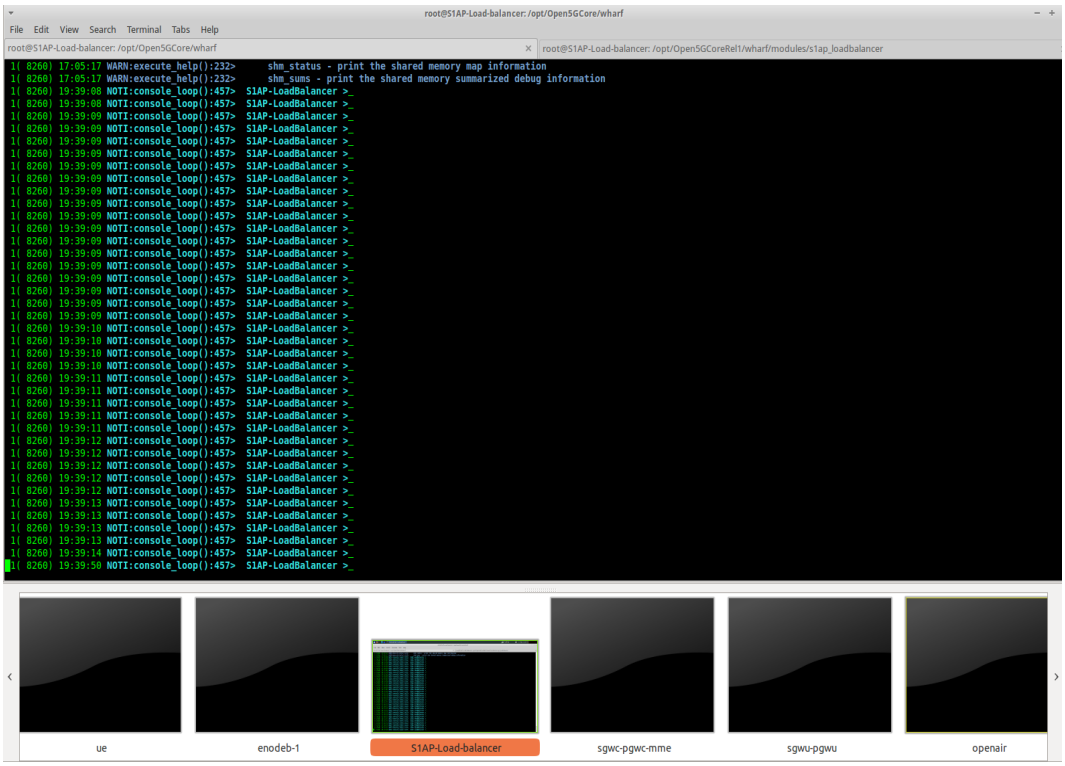


Figure 5.5: S1-MME Load Balancer running VM WorkStation

5.5 Conclusion

In this chapter, the implementation of S1-MME LB was presented. This section started with a brief description of the development environment, followed by project structure, that explains the important modules involved as part of the implementation and the module on which the implementation was focused on. The internal components of the S1-MME LB and their operations were discussed. The details on how each of these modules and sub-modules interconnect and communicate with each other to achieve the required functionality were explained. Session management that is used to decide where a S1AP message has to be forwarded when it arrives at the S1-MME LB was explained with the help of some mapping structures as described in the listings 5.1 and 5.2. Finally, configuration management, console support and why S1-MME LB was implemented as a software function were discussed.

6 Evaluation

This chapter describes how S1-MME LB was evaluated. In the first section of this chapter, the test setup on which the implemented system was evaluated is described. In addition to that, the test scenarios that were performed while evaluation are explained. In the next section, the tests that were conducted to verify the compliance of the S1-MME LB to 3GPP standards are discussed. At the end, the performance tests conducted are explained and their results are presented and analysed to evaluate the performance of the system. A benchmarking tool, an emulated UE, a eNodeB that complies with 3GPP standards and the EPC network which are part of the Fraunhofer FOKUS Open5GCore[Fra] test bed are used for evaluating the system.

6.1 Test Environment

This section describes the test environment used to perform the evaluation of S1-MME LB. It describes the set of network elements used and the configuration parameters set for each of these elements for evaluating the system implemented.

The evaluation of the created S1-MME LB was performed on a virtualized EPC network architecture running on a commodity computer with the following specifications:

- Operating system: Ubuntu 14.04.3 LTS
- Processor: Intel(R) Xeon(R) CPU E5-1620 (3.60GHz x 8)
- RAM memory: 16 GB

Since, the S1-MME LB is located on the S1 interface path between the eNodeB and MME, the evaluation of the developed software requires an EPC system network. For easy evaluation, the virtual software elements provided by Open5GCore testbed that run on VMWare Workstation are used for system evaluation. The used virtual EPC system consists of several core network elements such as MME, SGW, PGW and the EPC-enablers that provides internet Domain naming system (DNS) functionality, HSS, Internet breakout, along with other features like network selection and charging by Policy control rules function (PCRF). The control plane and the data plane of the EPS are separated in the Open5GCore test bed[Vla13]. Hence the control plane processing of the core network goes into the MME-SGWC-PGWC controller while the data path functionality goes into the switch SGWU-PGWU.

EPC network elements were run as virtual machines on the VMWare workstation using Ubuntu operating system. The specifications pertaining to memory and processor settings of each of these virtual machines running the access network and core network functionality is given below:

- **Benchmarking tool:**
  - Memory - 2 GB
  - Processors -2
- **UE**
  - Memory - 2 GB
  - Processors -2
- **eNodeB**
  - Memory - 2 GB
  - Processors -2
- **S1AP Load Balancer**
  - Memory - 2 GB
  - Processors -2
- **Controller (MME-SGWC-PGWC):**
  - Memory - 2 GB
  - Processors -2
- **Switch (Data Path unit SGWU-PGWU)**
  - Memory - 2 GB
  - Processors -1
- **EPC-Enablers**
  - Memory - 2 GB
  - Processors -1

It can be noted that since the evaluation of the system was focused on measuring the delay in the path that involves control plane traffic signalling, the switch and the epc enablers were configured with only 1 processor each while the other elements in the network were running on 2 processors each.

A benchmarking tool that emulates the access network was used for evaluating the performance of S1-MME LB when it is handling the attach and detach requests. Apart from this an emulated UE and a eNodeB that complies with 3GPP standard[Koc13] was

used to test S1 setup procedure, basic functionality of S1AP procedures such as Attach, Detach and intra-lte handovers and 3GPP standard compliance of the S1-MME LB. The testbed environment used for evaluation is as shown in the figure 6.1.

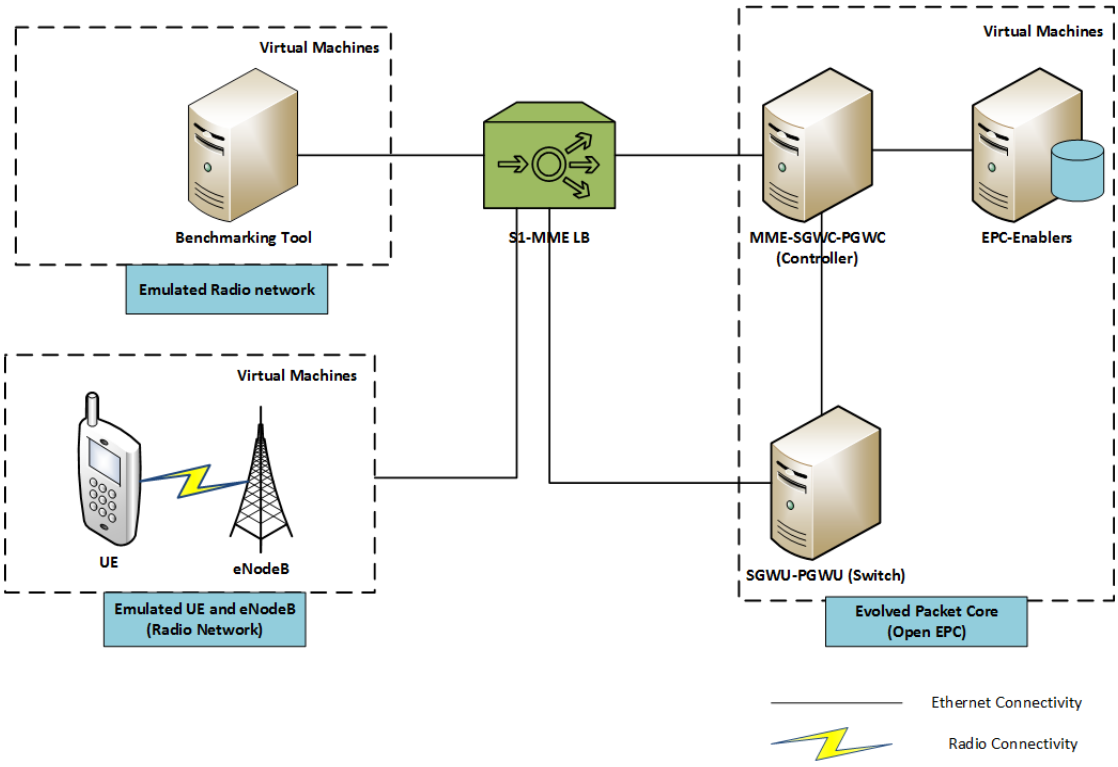


Figure 6.1: S1-MME Testbed environment

The network configuration diagram as shown in 6.2 explains the network level configurations of all the elements required for performing the evaluation of the system. The interface net_c represents the access network communication between UE and eNodeB while the interface net_d represents the communication between on the S1-MME and S1-U interfaces of the 3GPP standard. The ip addresses and port numbers used on each of the interfaces are described.

The benchmarking tool (BT) emulated 5 eNodeBs whose ip addresses are in the range between 192.168.4.100 to 192.168.4.104. For performing handovers 2 eNodeBs were used with the source eNodeB having the ip address 192.168.4.90 and the target eNodeB with the ip addresss 192.168.4.91 communication using the S1 interface on the port 36412. The LB is configured to use ip address 192.168.4.70 and port number 36412 for communication on the S1-MME interface. The controller MME-SGWC-PGWC was configured to use ip address 192.168.4.80 and port number 36412 for communication on the S1-MME interface. The data path element SGWU-PGWU switch was configured to use ip

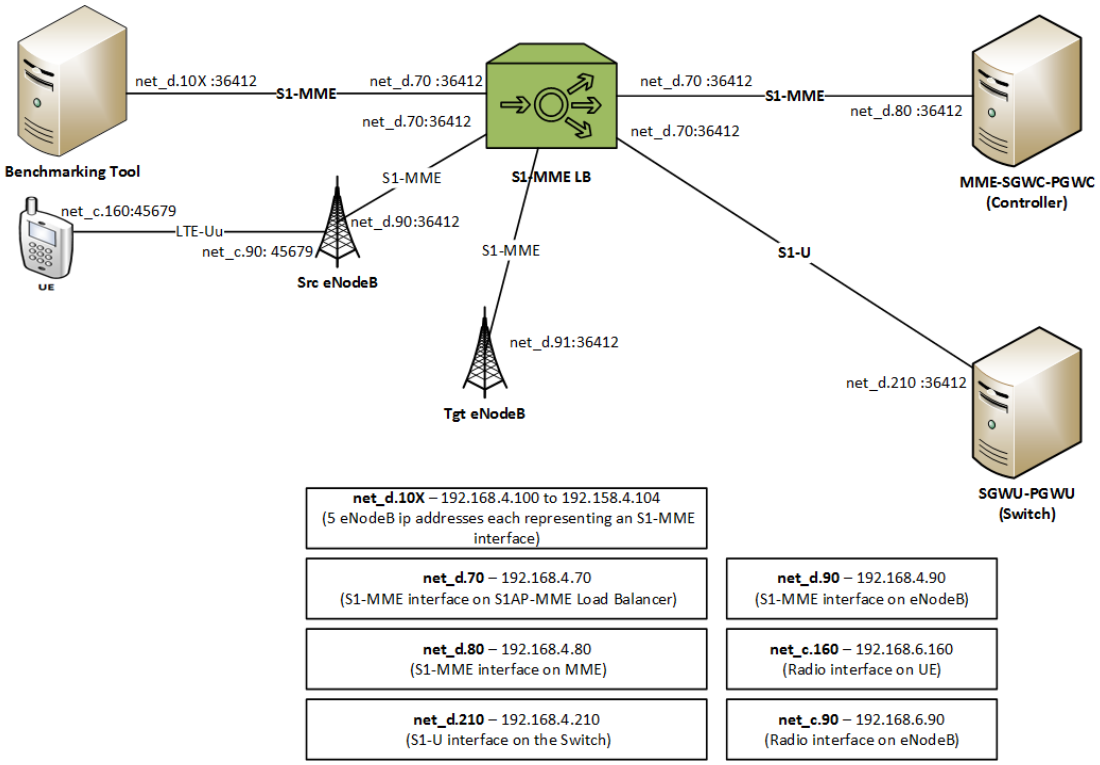


Figure 6.2: S1-MME Testbed Network Configuration

address 192.168.4.210 and port number 36412 for the communication on the S1-U interface. It should also be noted that SACK_TIMEOUT was also set to be 1ms instead of the default timeout value of 200ms on the nodes eNodeB/BT, S1-MME LB, MME and HSS to ensure that there is no delay in attach and detach operations whose messages are exchanged between these nodes on the SCTP sockets.

6.2 Test Scenarios

This section discusses the list of procedures verified using the implemented system. It lists all the scenarios tested along with the command line operations used to test them.

6.2.1 Test Cases

The implemented system was evaluated by verifying its functionality for the following 3GPP S1AP procedures:

- **Attachment Procedure:** An attach procedure as defined in 3GPP standard was performed to verify if the S1-MME LB is able to transparently forward the

messages involved in S1AP attach procedure back and forth between a eNodeB and a MME. A successful attach procedure indicates the ability of the S1-MME LB in handling all the standard signalling messages associated with attach operation.

- **Detachment Procedure:** A detach procedure as defined in 3GPP standard was conducted to evaluate the ability of the S1-MME LB to handle the S1AP detach procedure. The ability of the S1-MME to handle the messages involved in S1AP detach procedure is verified with this test.
- **Handover Procedure:** A S1 intra lte-handover as defined in 3GPP standard was executed to see if the S1-MME LB is properly able to handle the messages involved in S1AP handover procedure between the source eNodeB, target eNodeB and the MME.
- **Performance Tests:** In addition to individual attachment and detachment tests, the performance of the S1-MME LB in terms of delay latencies was evaluated by executing multiple attach and detach operations using the benchmarking tool.

6.2.2 Test Execution Procedures

This section describes the way the test cases were executed from the command line of S1-MME LB system. The following is the list of command line operations along with user input parameters that were used to evaluate the functionality of S1-MME LB:

- **ue.attach *IMSI APN*:** On executing this command, the UE is attached to the core network on successful completion of the NAS attachment procedures and on creation of EPS bearer. IMSI is the parameter used by the UE to identify itself uniquely while registering with the network, while the parameter APN represents the default PDN network the UE needs to establish the connection with.
- **ue.detach *IMSI*:** This command detaches the UE from the core network after the completion of NAS detach procedures. The user to be detached from the network is uniquely identified by the parameter IMSI.
- **enb.intra_lte_ho *ID TAC Cell-ID*:** Triggers an S1 Handover procedure to another eNodeB by sending a Handover Required message. ID is a UE identifier, TAC represents the tracking area code of the eNodeB, where Cell-ID represents the cell identifier of the target eNodeB to which the handover is needed.
- **s1ap_loadbalancer.add *IP_Address*:** As the S1-MME LB needs to balance the incoming S1-MME traffic across different MMEs, it is required to feed the LB with the ip addresses of the MMEs to which it can load balance the incoming traffic. These inputs are provided using the above stated command line operation.

For verifying the performance of the S1-MME LB, a benchmarking tool was used that emulates the entire access network operations. These tests help to analyse the mean delay latencies taken by the system for attach and detach operations in the presence of LB. This analysis will help to evaluate the performance of the implemented software in terms of delay latencies.

The following are the list of commands executed using the command line provided by the benchmarking tool:

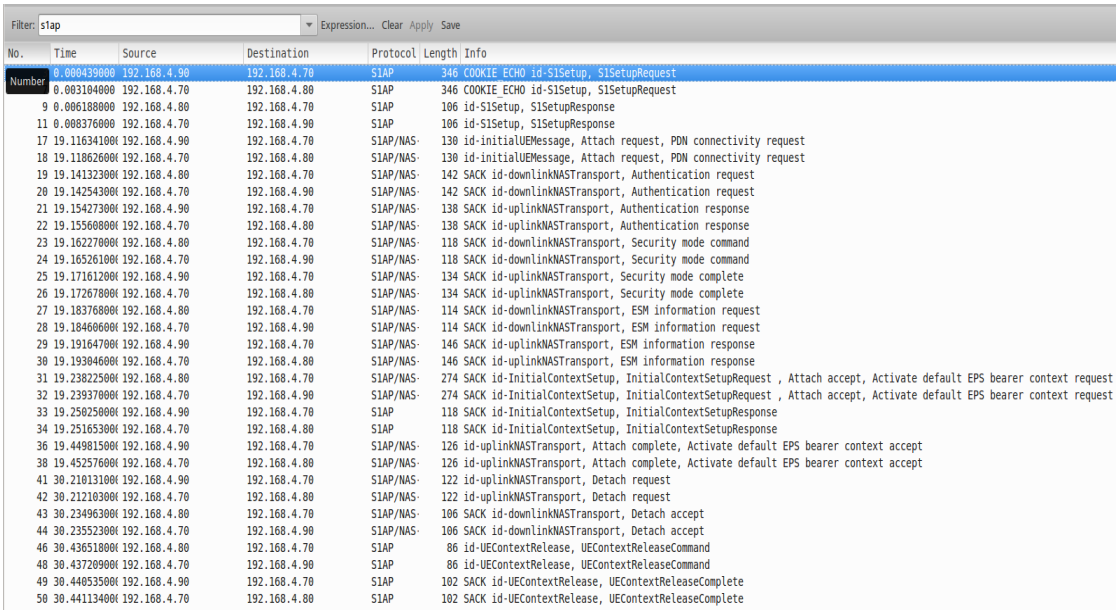
- **btr.attach *No_of_Attach_Operations*:** On executing this command, the benchmarking tool does multiple attach operations to the core network. The number of attach operations are specified as arguments with the command line operation.
- **btr.detach *No_of_Detach_Operations*:** On executing this command, the benchmarking tool de registers multiple users that are previously registered with the core network. The number of detach operations depends on the arguments provided with the command.

6.3 Compliance with 3GPP Standards

To validate the compliance of S1-MME LB with 3GPP standards, the wireshark[Wir] traces of the packets incoming to the S1-MME LB from a standard compliant eNodeB and the traces of the messages forwarded by the S1-MME LB to the MME are compared.

6.3.1 S1AP Attach and Detach Procedures

Figure 6.3 shows the wireshark traces of the traffic captured on S1 interface as seen by the S1-MME LB during the attach and detach operations. On careful study of the messages in the trace, it indicates that S1-MME LB was able to comprehend and handle all the S1AP messages associated with Attach/Detach procedure such as PDN Connectivity, Authentication verification, Security exchange, ESM information exchange, Context setup, Context release etc, thus resulting in successful completion of attach and detach operations.


Filter: s1ap Expression... Clear Apply Save

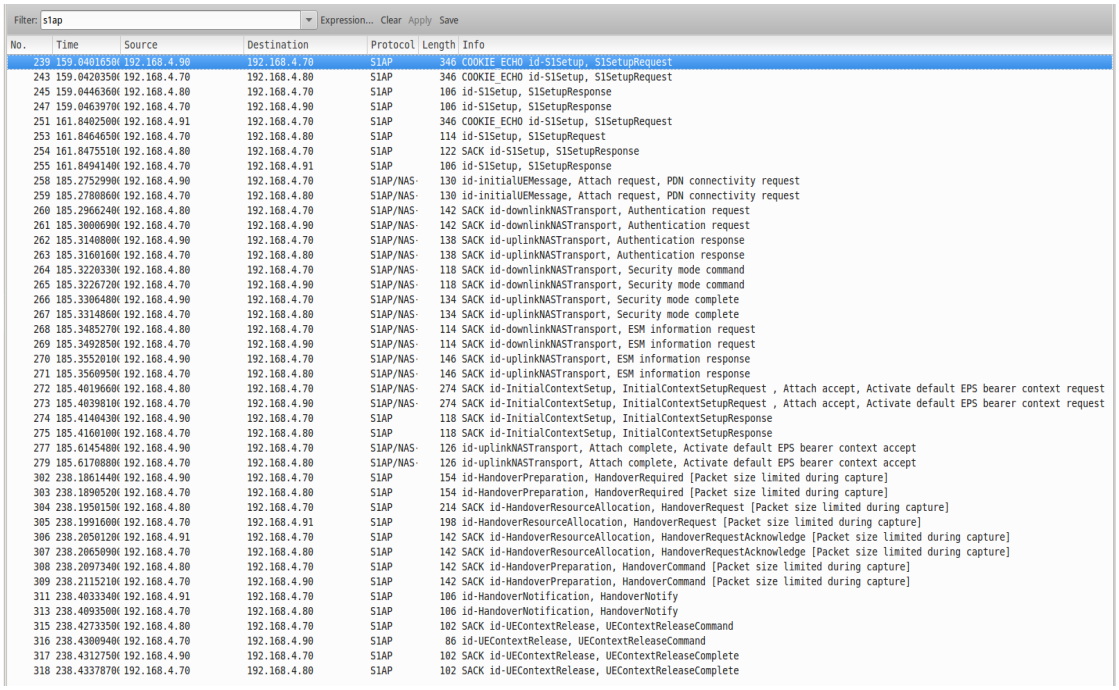
No.	Time	Source	Destination	Protocol	Length	Info
Number	0.000439000	192.168.4.90	192.168.4.70	S1AP	346	COOKIE_ECHO id-S1Setup, S1SetupRequest
	0.003104000	192.168.4.70	192.168.4.80	S1AP	346	COOKIE_ECHO id-S1Setup, S1SetupRequest
9	0.006188000	192.168.4.80	192.168.4.70	S1AP	106	id-S1Setup, S1SetupResponse
11	0.008376000	192.168.4.70	192.168.4.90	S1AP	106	id-S1Setup, S1SetupResponse
17	19.11634100(	192.168.4.90	192.168.4.70	S1AP/NAS·	130	id-initialUEMessage, Attach request, PDN connectivity request
18	19.11862600(	192.168.4.70	192.168.4.80	S1AP/NAS·	130	id-initialUEMessage, Attach request, PDN connectivity request
19	19.14132300(	192.168.4.80	192.168.4.70	S1AP/NAS·	142	SACK id-downlinkNASTransport, Authentication request
20	19.14254300(	192.168.4.70	192.168.4.90	S1AP/NAS·	142	SACK id-downlinkNASTransport, Authentication request
21	19.15427300(	192.168.4.90	192.168.4.70	S1AP/NAS·	138	SACK id-uplinkNASTransport, Authentication response
22	19.15560800(	192.168.4.70	192.168.4.80	S1AP/NAS·	138	SACK id-uplinkNASTransport, Authentication response
23	19.16227000(	192.168.4.80	192.168.4.70	S1AP/NAS·	118	SACK id-downlinkNASTransport, Security mode command
24	19.16526100(	192.168.4.70	192.168.4.90	S1AP/NAS·	118	SACK id-downlinkNASTransport, Security mode command
25	19.17161200(	192.168.4.90	192.168.4.70	S1AP/NAS·	134	SACK id-uplinkNASTransport, Security mode complete
26	19.17267800(	192.168.4.70	192.168.4.80	S1AP/NAS·	134	SACK id-uplinkNASTransport, Security mode complete
27	19.18376800(	192.168.4.80	192.168.4.70	S1AP/NAS·	114	SACK id-downlinkNASTransport, ESM information request
28	19.18460600(	192.168.4.70	192.168.4.90	S1AP/NAS·	114	SACK id-downlinkNASTransport, ESM information request
29	19.19164700(	192.168.4.90	192.168.4.70	S1AP/NAS·	146	SACK id-uplinkNASTransport, ESM information response
30	19.19304600(	192.168.4.70	192.168.4.80	S1AP/NAS·	146	SACK id-uplinkNASTransport, ESM information response
31	19.23822500(	192.168.4.80	192.168.4.70	S1AP/NAS·	274	SACK id-InitialContextSetup, InitialContextSetupRequest , Attach accept, Activate default EPS bearer context request
32	19.23937000(	192.168.4.70	192.168.4.90	S1AP/NAS·	274	SACK id-InitialContextSetup, InitialContextSetupRequest , Attach accept, Activate default EPS bearer context request
33	19.25025000(	192.168.4.90	192.168.4.70	S1AP	118	SACK id-InitialContextSetup, InitialContextSetupResponse
34	19.25165300(	192.168.4.70	192.168.4.80	S1AP	118	SACK id-InitialContextSetup, InitialContextSetupResponse
36	19.44981500(	192.168.4.90	192.168.4.70	S1AP/NAS·	126	id-uplinkNASTransport, Attach complete, Activate default EPS bearer context accept
38	19.45257600(	192.168.4.70	192.168.4.80	S1AP/NAS·	126	id-uplinkNASTransport, Attach complete, Activate default EPS bearer context accept
41	30.21013100(	192.168.4.90	192.168.4.70	S1AP/NAS·	122	id-uplinkNASTransport, Detach request
42	30.21210300(	192.168.4.70	192.168.4.80	S1AP/NAS·	122	id-uplinkNASTransport, Detach request
43	30.23496300(	192.168.4.80	192.168.4.70	S1AP/NAS·	106	SACK id-downlinkNASTransport, Detach accept
44	30.23552300(	192.168.4.70	192.168.4.90	S1AP/NAS·	106	SACK id-downlinkNASTransport, Detach accept
46	30.43651800(	192.168.4.80	192.168.4.70	S1AP	86	id-UEContextRelease, UEContextReleaseCommand
48	30.43720900(	192.168.4.70	192.168.4.90	S1AP	86	id-UEContextRelease, UEContextReleaseCommand
49	30.44053500(	192.168.4.90	192.168.4.70	S1AP	102	SACK id-UEContextRelease, UEContextReleaseComplete
50	30.44113400(	192.168.4.70	192.168.4.80	S1AP	102	SACK id-UEContextRelease, UEContextReleaseComplete

Figure 6.3: Traffic capture of Attach and Detach operations on S1-MME interface

6.3.2 S1AP Handover Procedure:

Figure 6.4 shows the wireshark traces of all the signalling messages involved during the S1 Handover procedure of the user from source eNodeB to the target eNodeB, as seen on the S1-MME LB. It can be observed from the logs that the S1-MME LB is successful in handling the handover from one eNodeB to the other by handling all the messages associated with handover such as Handover Required, Handover Request,

HandoverRequest Acknowledge, Handover Command, Handover Notify, eNodeB status transfer, MME status transfer and finally context release after the handover is successful.



Filter: s1ap — Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
239	159.0401650(	192.168.4.90	192.168.4.70	S1AP	346	COOKIE_ECHO id-S1Setup, S1SetupRequest
243	159.0420350(	192.168.4.70	192.168.4.80	S1AP	346	COOKIE_ECHO id-S1Setup, S1SetupRequest
245	159.0446360(	192.168.4.80	192.168.4.70	S1AP	106	id-S1Setup, S1SetupResponse
247	159.0463970(	192.168.4.70	192.168.4.90	S1AP	106	id-S1Setup, S1SetupResponse
251	161.8402500(	192.168.4.91	192.168.4.70	S1AP	346	COOKIE_ECHO id-S1Setup, S1SetupRequest
253	161.8464650(	192.168.4.70	192.168.4.80	S1AP	114	id-S1Setup, S1SetupRequest
254	161.8475510(	192.168.4.80	192.168.4.70	S1AP	122	SACK id-S1Setup, S1SetupResponse
255	161.8494140(	192.168.4.70	192.168.4.91	S1AP	106	id-S1Setup, S1SetupResponse
258	185.2752990(	192.168.4.90	192.168.4.70	S1AP/NAS-	130	id-initialUEMessage, Attach request, PDN connectivity request
259	185.2780860(	192.168.4.70	192.168.4.80	S1AP/NAS-	130	id-initialUEMessage, Attach request, PDN connectivity request
260	185.2966240(	192.168.4.80	192.168.4.70	S1AP/NAS-	142	SACK id-downlinkNASTransport, Authentication request
261	185.3000690(	192.168.4.70	192.168.4.90	S1AP/NAS-	142	SACK id-downlinkNASTransport, Authentication request
262	185.3140800(	192.168.4.90	192.168.4.70	S1AP/NAS-	138	SACK id-uplinkNASTransport, Authentication response
263	185.3160160(	192.168.4.70	192.168.4.80	S1AP/NAS-	138	SACK id-uplinkNASTransport, Authentication response
264	185.3220330(	192.168.4.80	192.168.4.70	S1AP/NAS-	118	SACK id-downlinkNASTransport, Security mode command
265	185.3226720(	192.168.4.70	192.168.4.90	S1AP/NAS-	118	SACK id-downlinkNASTransport, Security mode command
266	185.3306480(	192.168.4.90	192.168.4.70	S1AP/NAS-	134	SACK id-uplinkNASTransport, Security mode complete
267	185.3314860(	192.168.4.70	192.168.4.80	S1AP/NAS-	134	SACK id-uplinkNASTransport, Security mode complete
268	185.3485270(	192.168.4.80	192.168.4.70	S1AP/NAS-	114	SACK id-downlinkNASTransport, ESM information request
269	185.3492850(	192.168.4.70	192.168.4.90	S1AP/NAS-	114	SACK id-downlinkNASTransport, ESM information request
270	185.3552010(	192.168.4.90	192.168.4.70	S1AP/NAS-	146	SACK id-uplinkNASTransport, ESM information response
271	185.3560950(	192.168.4.70	192.168.4.80	S1AP/NAS-	146	SACK id-uplinkNASTransport, ESM information response
272	185.4019660(	192.168.4.80	192.168.4.70	S1AP/NAS-	274	SACK id-InitialContextSetup, InitialContextSetupRequest , Attach accept, Activate default EPS bearer context request
273	185.4039810(	192.168.4.70	192.168.4.90	S1AP/NAS-	274	SACK id-InitialContextSetup, InitialContextSetupRequest , Attach accept, Activate default EPS bearer context request
274	185.4140430(	192.168.4.90	192.168.4.70	S1AP	118	SACK id-InitialContextSetup, InitialContextSetupResponse
275	185.4160100(	192.168.4.70	192.168.4.80	S1AP	118	SACK id-InitialContextSetup, InitialContextSetupResponse
277	185.6145480(	192.168.4.90	192.168.4.70	S1AP/NAS-	126	id-uplinkNASTransport, Attach complete, Activate default EPS bearer context accept
279	185.6170880(	192.168.4.70	192.168.4.80	S1AP/NAS-	126	id-uplinkNASTransport, Attach complete, Activate default EPS bearer context accept
302	238.1861440(	192.168.4.90	192.168.4.70	S1AP	154	id-HandoverPreparation, HandoverRequired [Packet size limited during capture]
303	238.1890520(	192.168.4.70	192.168.4.80	S1AP	154	id-HandoverPreparation, HandoverRequired [Packet size limited during capture]
304	238.1950150(	192.168.4.80	192.168.4.70	S1AP	214	SACK id-HandoverResourceAllocation, HandoverRequest [Packet size limited during capture]
305	238.1991600(	192.168.4.70	192.168.4.91	S1AP	198	id-HandoverResourceAllocation, HandoverRequest [Packet size limited during capture]
306	238.2050120(	192.168.4.91	192.168.4.70	S1AP	142	SACK id-HandoverResourceAllocation, HandoverRequestAcknowledge [Packet size limited during capture]
307	238.2065090(	192.168.4.70	192.168.4.80	S1AP	142	SACK id-HandoverResourceAllocation, HandoverRequestAcknowledge [Packet size limited during capture]
308	238.2097340(	192.168.4.80	192.168.4.70	S1AP	142	SACK id-HandoverPreparation, HandoverCommand [Packet size limited during capture]
309	238.2115210(	192.168.4.70	192.168.4.90	S1AP	142	SACK id-HandoverPreparation, HandoverCommand [Packet size limited during capture]
311	238.4033340(	192.168.4.91	192.168.4.70	S1AP	106	id-HandoverNotification, HandoverNotify
313	238.4093500(	192.168.4.70	192.168.4.80	S1AP	106	id-HandoverNotification, HandoverNotify
315	238.4273350(	192.168.4.80	192.168.4.70	S1AP	102	SACK id-UEContextRelease, UEContextReleaseCommand
316	238.4300940(	192.168.4.70	192.168.4.90	S1AP	86	id-UEContextRelease, UEContextReleaseCommand
317	238.4312750(	192.168.4.90	192.168.4.70	S1AP	102	SACK id-UEContextRelease, UEContextReleaseComplete
318	238.4337870(	192.168.4.70	192.168.4.80	S1AP	102	SACK id-UEContextRelease, UEContextReleaseComplete

Figure 6.4: Traffic capture of Handover operations on S1-MME interface

6.3.3 Observations from the wireshark traces:

On studying the wireshark trace of S1AP Attach/Detach and handover procedures in 6.3 and 6.4, S1-MME LB comprehends all the signaling messages sent by a 3GPP standard compliant eNodeB emulator, and forwards the same to the MME, after necessary modifications such as changing the source address and the identifier transaction identifier eNB-UE-S1AP-ID. It can be noted from the above traffic capture that all the signaling messages are repeated twice, one that is sent by the eNodeB (identified by ip address 192.168.4.90 for source eNodeB and 192.168.4.91 for target eNodeB), the other is the same message redirected by the S1-MME LB (identified by ip address 192.168.4.70) towards the MME (identified by ip address 192.168.4.80), after deciding to which MME it has to forward the message. It can be noted that the LB acts as a proxy node which transparently adds itself as representative of the correspondent parties in a back-to-back fashion. By doing this, the eNBs are completely unaware of the existence of a LB, while the MME has some minimal information, thus making the S1-MME LB transparent to the access network. From the logs it evident that the S1-MME LB was able to handle all the S1AP messages coming from a 3GPP standard compliant eNodeB emulator and MME. Thus, it can be concluded that the S1-MME LB adheres to the 3GPP standards.

6.4 Performance Measurements:

This section analyses the delay measurements of attach and detach operations in the presenece of S1-MME LB in the 3GPP EPC architecture. These measurements helps to evaluate the performance of the S1-MME LB in terms of the the amount of delay overhead it has introduced in the attach and detach operations. The delay measurements are performed using a benchmarking tool. The benchmarking tool generates several of the attach and detach operations in multiples of 10 starting from 10 and ending at 50 Attach/Detach operations per second. Note that the BT sends traffic continuously for a period of 2 seconds. This is to ensure that the system performance is evaluated in a stable state where it continuously processses the sequence of Attach/Detach operations for the above stated time period. This results in analysing the mean delays in a stable environment.

6.5 Delay analysis for Attach operations:

In the graph as shown in figure 6.5, x-axis represents the number of S1AP attach operations performed per second, while the y-axis represents the average delay in milliseconds it took for each operation to complete.

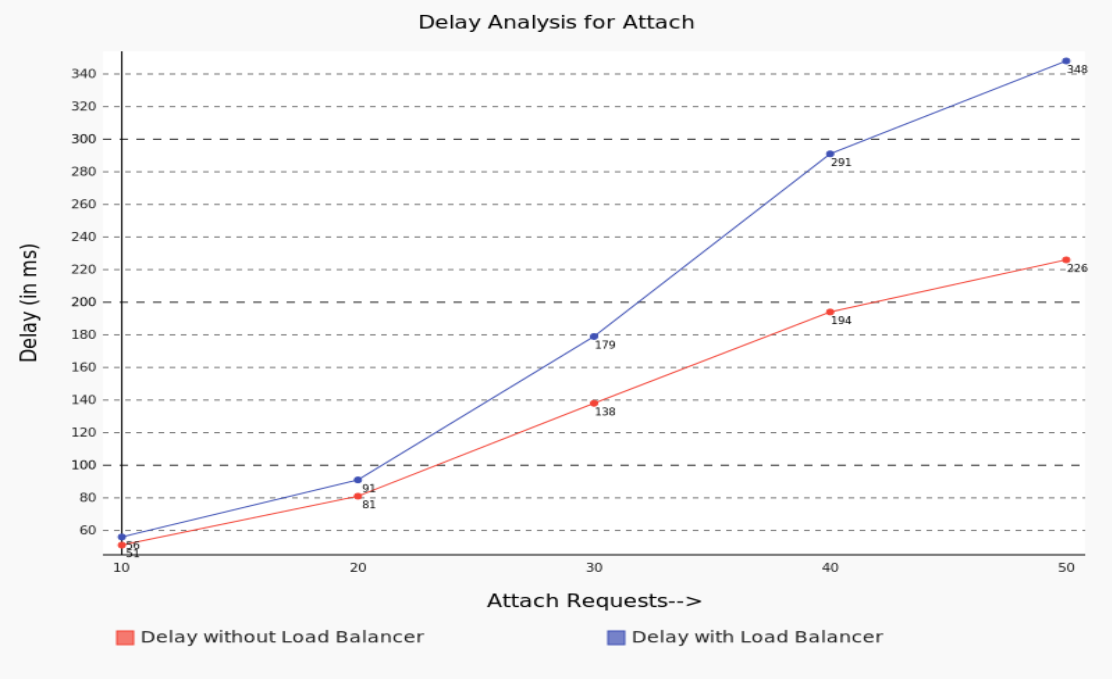


Figure 6.5: Delay analysis for Attach operations

The line graph in red indicates the delay for attach operations in the absence of S1-MME LB on the S1-MME communication path between eNodeB and MME. The average

delay for 10 operations is 51 ms, while the delay for 20 operations is 81 ms. The trend from the graph shows that the delay has grown exponentially as the number of operations are increased in steps of 10 from 10 to 50 operations per second. A similar trend can be observed for the the delay measurements for attach operations in the presence of S1-MME LB as seen in line graph in blue. It can be observed that the overhead introduced by the S1-MME LB is only 9.8% and 12.34% for 10 and 20 operations per respectively, but it has increased to 29.7%, 50%, 53% for 30, 40 and 50 operations per seconds respectively. The extra overhead in delay for 30, 40, 50 operations/sec is due to the fact that the performance of the S1-MME LB is limited by the processing speed of MME and HSS. As the attach operations involve the UE registration with MME and in turn the MME verifying the subscriber information with HSS, the processing and queuing delays at these nodes will add up to the delay overhead at the S1-MME LB, thus limiting its ability by the rate of processing at MME and HSS.

6.5.1 Delay analysis for Detach operations:

In the graph as shown in figure 6.6, x-axis represents the number of S1AP detach operations performed per second, while the y-axis represents the average delay in milliseconds it took for each of operations as mentioned on x-axis to complete.

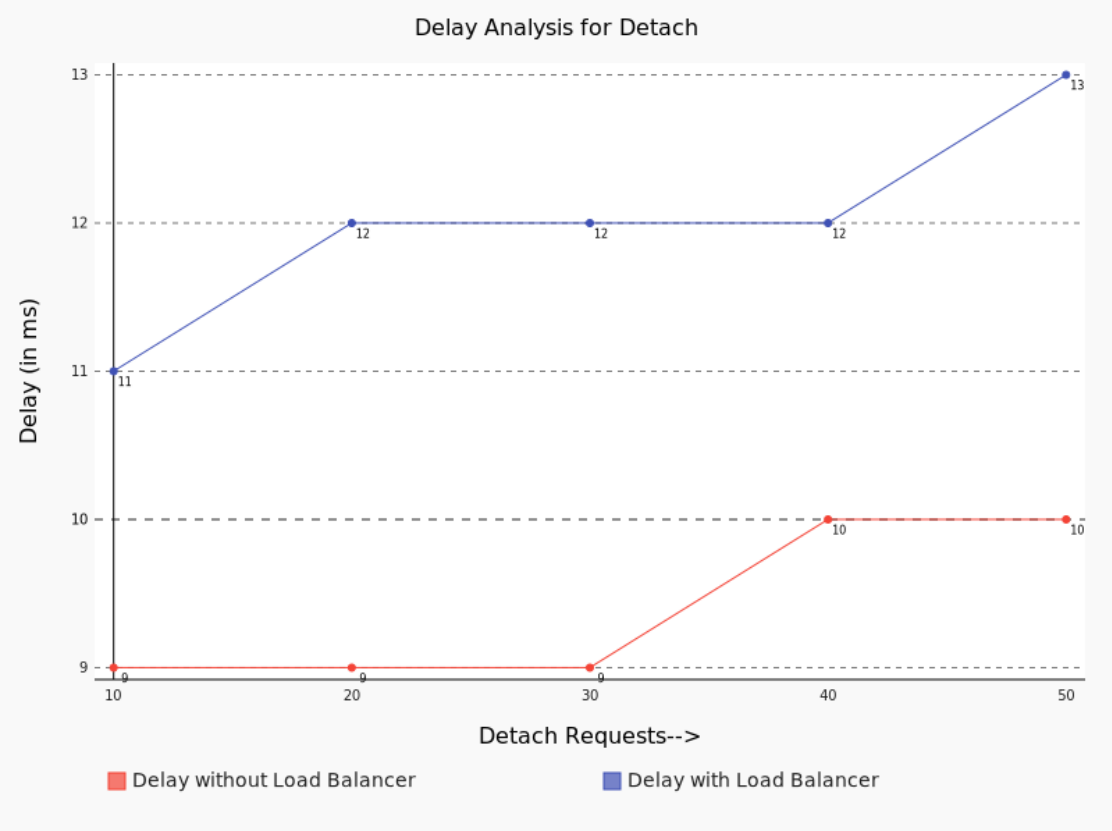


Figure 6.6: Delay analysis for Detach operations

The line graph in red indicates the delay for detach operations in the absence of S1-MME LB. The average delay for 10, 20 and 30 detach operations/sec is 9 ms, while for 40 and 50 detach operations per second, it is 10 ms. Since the detach operations are simple and involve less round trip time operations between eNodeB and MME, the delay values are less compared to the attach operations. The blue line graph as shown in figure 6.6 represents the delay measurements in the presence of S1-MME LB. Both the measurements of detach show a similar trend of linearly growing delays for a constant period and then exponentially growing after reaching a certain threshold of detach operations. For example in the presence of S1-MME LB as in the blue line graph, the delay for detach operations is 11ms for 10 operations/sec, but has exponentially grown to 12ms when the operations are increased from 10 to 20 operations/sec, after which the delay is constant till the operations reached a threshold of 40 operations/sec

after which has exponentially grown to 13ms for 50 detach operations/sec. This is due to the fact that the detach operations involve less signaling messages and thus less processing times at each of the nodes eNodeB, S1-MME LB and MME. Furthermore, these messages messages won't queue up at the each of these nodes till a large number number of detach operations accumulate, thus resulting in less mean delay for completion of the detach operations.

6.6 Conclusion:

Based on the results for the scenarios tested, S1-MME LB was successful in running as a software function in a virtual environment and able to support S1 setup, S1AP procedures such as Attach, Detach and intra-LTE handover conforming to 3GPP standards. As a conclusion on performance, the implemented system performed as expected not introducing much delay latency overhead during the attachment procedures as long as the number of attach operations per second are minimal i.e, for 10-20 operations per second, but there is a significant increase in the delay overhead when the number of users are increased i.e., above 30 operations per second, due to fact that the s1ap messages and other control messages are queued up at MME and HSS thus adding additional delay overhead to the actual processing time of the messages at the S1-MME LB. Coming to S1AP Detach, the S1-MME LB has introduced a delay overhead between 20 to 33.33% for operations in range between 10 to 50 detaches/sec. However, it is permissible to have an delay overhead of greater than 10% in case of detach operations, as this will not hamper the quality of experience perceived by the user.

7 Conclusion

7.1 Summary

With every passing day mobile networks are moving to an all ip based model, where all the services are run on ip traffic, but with a different QoS class, depending on the priority of the service. The trend shows that the number of end-users adopting smartphones increased rapidly who are running several bandwidth-hungry applications like real-time video chat, online-gaming, and multimedia applications that require scalable networks.

As the end-users requirements keep changing rapidly, it is really important that the core networks react flexibly to the needs of the users connected to the access network. It is good to have a mechanism where the core network elements like MME can either be scaled out or scaled in on-demand. An existing approach like S1-Flex can provide scalability, but the downside with this approach is that it is not flexible as it is not transparent to the eNodeB. This makes any change in core network needs to be announced to eNodeB, that involves several round-trip time operations between eNodeB and MME resulting in the user experiencing a bad quality of experience. Another approach is to spread the control plane traffic coming from eNodeB across multiple MMEs by using a random load balancer, but the problem with this approach is that it cannot provide an affinity at transaction level. Hence, the S1-MME load balancing approach is proposed that gives the required elasticity in core networks while being transparent to eNodeB.

Since the existing state of art is not sufficient to handle the dynamically changing needs of the mobile network architecture, a new element called S1-MME load balancer (LB) was introduced to give elasticity to the core network. It was located between the eNodeB and the MME. At the S1-MME LB, S1AP messages are redirected based on eNB-UE-S1AP-ID value as well as the identity of the eNodeBs i.e, eNodeB ip address. When a s1ap message arrives from an eNodeB, it parses the eNB-UE-S1AP-ID and looks in its session management information to find out if the message bearing this identifier from that specific eNodeB is already assigned an MME. If so, it will forward the message to the same MME, thus maintaining affinity to MME at a transaction level. In case no mapping information exists, it will communicate with a internal module called 'balancer policing' requesting it to assign an MME to which this message can be forwarded. The 'balancer policing' module runs a round-robin like algorithm to decide the destination MME. Now, the S1-MME LB forwards the message the chosen MME and MME will respond to S1-MME LB with the corresponding response messages. This makes S1-MME LB to act as a proxy node which transparently adds itself as representative of the

correspondent parties in a back-to-back fashion.This makes eNodeB completely unaware of the existence of the load balancer. This allows the network managers to make any changes in the core network without exposing the details to access network, thus providing the needed flexibility to scale the MMEs in the core network on-demand.

S1-MME LB is implemented as a software function. This implementation is evaluated using the VMWare system virtualization software as it is easier to create and maintain the testbed on a single machine than running them on separate machines. This reduces the costs greatly, without compromising on any functionality of the network elements.

The aim of the thesis was to implement a load balancer on the S1-MME interface that distributes the incoming traffic from eNodeB onto multiple MMEs, while maintaining affinity at an transaction level, while being transparent to eNodeB and MME. It was important that it is as closely compliant as possible to the 3GPP Standards. It was also expected that the load balancer doesn't introduce no more delay than 10% in the communication between eNodeB and MME during the attach operations.

As discussed in section 6.4 of chapter 6, the results showed that the it was a lightweight software component which parses very few parameters of the s1ap messages. This helped to keep the delay overhead of the S1-MME LB under the delay constraints as mentioned in the previous paragraph, as long as it not limited by speed of MME. There were situations where the LB sees more than 10% overhead in delay due to the fact that it is limited by the speed of processing at MME and HSS where the messages related to Attach/Detach operations gets queued up for processing at these elements.

The results also showed that the load balancer complied with the 3GPP standards and successfully handled the S1AP procedures Setup, Attach, Detach and Handover without need for introducing any new variables in the existing S1AP message structure, except that it needs a few tweaks in MME design by using some optional fields of the message as discussed in the section 4.3.2 of Chapter 4.

7.2 Dissemination

The S1-MME load balancer implemented as part of the thesis work will be integrated as part of the Open5GCore test bed platform as a state-of-the-art EPS architecture. This work is expected to be integrated to Mobile Cloud Networking project. It will be delivered to all the partners of the Fraunhofer FOKUS who licensed or to those who decided to license our future releases.

7.3 Problems encountered

The most important problem encountered during the thesis was the need to keep the implementation of S1-MME load balancer as closely compliant as possible to the standard. Since the load balancer is introduced as a new element between eNodeB and MME, it needed some work arounds to maintain the session mappings between eNodeB and MME. Hence, a slight deviation from the 3GPP standard was introduced such as making the MME to use enodeb name as mme name for handling S1 setup request/response message and using pointer variable *MME-UE-S1AP-ID-2 for storing eNB-UE-S1AP-ID that contains the required session information for handling the Handover procedure. The other important problem is to see that the load balancer doesn't introduce much delay in the path between eNodeB and MME. Several session mapping techniques were analysed and finally a special hash based technique was used to ensure that the delay measurements in attach requests are under 10%. There is still some scope for improving the implementation to reduce the delay measurements especially when scaling the attach requests to higher numbers.

7.4 Outlook

- Though there is already a good implementation of the S1-MME load balancer, there are still some areas that need further improvements, to enhance the functionality of the system. The following are some of the future enhancements possible to the existing work:
- Currently the S1-MME LB is balancing the incoming S1 traffic based on the eNB-UE-S1AP-ID, but this can be extended in future to support balancing based on IMSI information that allows the load balancer to do core network separation based on the subscriber type.
- Also S1-MME LB can be improved further by taking into consideration hybrid balancing policies like load on MME, delay to reach an MME and a mix of several other parameters.
- In future versions, in order to reduce the overhead introduced in S1-MME traffic path due to the introduction of S1-MME LB between eNodeB and MME, the S1-MME traffic can be processed at the kernel level at the S1-MME LB for faster packet processing.
- Also, an interface to support communication with the network orchestrator can be added to the load balancer. This allows to give external inputs to the load balancer based on the changes in the core network.

List of Acronyms

2G	Second Generation of Mobile Communications technology
3G	Third Generation of Mobile Communications technology
3GPP	3rd Generation Partnership Project
4G	Fourth Generation of Mobile Communications technology
AMBR	Aggregated Maximum Bit Rate
APN	Access Point Name
BSC	Base Station Controller
BTS	Base Transceiver Station
C-RNTI	Cell Radio Network Temporary Identifier
CDT	Eclipse C/C++ Development Tool
CGI	Cell Group Identifier
COTS	Commercial off the Shelf
CS	Circuit Switched
CSFB	Circuit Switch Fallback
DHCP	Dynamic Host Configuration Protocol
DL	Downlink
DNS	Domain Name System
DRB	Data Radio Bearer
eNB	Evolved Node B
eNB-ID	Evolved Node B Identifier
e-RAB-ID	EUTRAN Radio Access Bearer Id
EBI	EPS Bearer ID
ECM	EPS Connection Management
EMM	EPS Mobility Management
EPC	Evolved Packet Core
EPS	Evolved Packet System
ERAB	EUTRAN Radio Access Bearer (E-RAB)
ETSI	European Telecommunications Standard Institute
EUTRAN	Evolved Universal Terrestrial Radio Access Network
F-TEID	Fully Qualified Tunnel End Point Identifier
FOKUS	Fraunhofer Institut fuer offene Kommunikationssysteme
GBR	Guaranteed Bit Rate
GCC	GNU Compiler Collection
GPRS	General Packet Radio Service
GTP	GPRS Tunneling Protocol
GTP-U	GPRS Tunneling Protocol User Plane
GUI	Graphical User Interface

GUMMEI	Global Unique MME Identifier
GUTI	Globally Unique Temporary Identity
HSS	Home Subscriber Server
IDE	Integrated Development Environment
IPv4	Internet Protocol version 4
IPv4	Internet Protocol version 6
IETF	Internet Engineering Task Force
KSI	Key Set Identifer
LAN	Local Area Network
LTE	Long Term Evolution
M2M	Machine-to-machine
MAC	Medium Access Control
MCC	Mobile Country Code
MME	Mobility Management Entity
MMEI	MME Identifier
MNC	Mobile Network Code
MSC	Mobile Switching Station
NAS	Non-Access Stratum
NFV	Network Function Virtualziation
NGN	Next Generation Network
PDCP	Packet Data Control Protocol
PDN	Packet Data Network
PDU	Protocol Data Unit
PGW/P-GW	Packet Data Network Gateway
PLMN	Public Land Mobile Network
PS	Packet Switched
PUCCH	Physical Uplink Control Channel
QCI	QoS Class Identifier
QoS	Quality of Service
QoE	Quality of Experience
RACH	Random Access Channel
RAN	Radio Access Network
RLC	Radio Link Control
RNTI	Radio Network Temporary Identifier
RRC	Radio Resource Control
S1-MME LB	S1-MME Load Balancer
S1AP	S1 Application Part
SDN	Software Defined Networking
SGW/S-GW	Serving Gateway
SGWU-PGWU	User plane Switch unit of SGW and PGW (Fraunhofer FOKUS Inhouse element)
SN	Sequence Number
SRB	Signaling Radio Bearer
TA	Tracking Area

TAC	Tracking Area Code
TEID	Tunnel Endpoint Identifier
TMSI	Temporary Mobile Subscriber Identity
TS	Technical Specification
UE	User Equipment
UL	Uplink
USIM	Universal Subscriber Identity Module
UTRAN	Universal Terrestrial Radio Access Network
VCS	Version Control System
XML	Extensible Markup Language

Bibliography

[3GP12] 3GPP Tech: *3GPP TS 23.041 version 11.4.0 Release 11, ETSI TS 123 041 V11.4.0: Digital cellular telecommunications system (Phase 2+);Universal Mobile Telecommunications System (UMTS); Technical realization of Cell Broadcast Service (CBS); Stage 3*, 2012. Technical Specification.

[3GP13a] 3GPP Tech: *3GPP: General Packet Radio System (GPRS) Tunneling Protocol User Plane (GTPv1-U). Ts 29.281, 3rd Generation Partnership Project (3GPP)*, 2013. Technical Specification.

[3GP13b] 3GPP Tech: *3GPP Tech. Spec. 23.401 version 11.4.0: LTE; General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E- UTRAN) access*, 2013. Technical Specification.

[3GP13c] 3GPP Tech: *3GPP Tech. Spec. 24.301 version 11.5.0: Universal Mobile Telecommunications System (UMTS); LTE; Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS); Stage 3*, 2013. Technical Specification.

[3GP13d] 3GPP Tech: *3GPP Tech. Spec. 36.306 version 11.4.0 Release 11, LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) radio access capabilities*, 2013. Technical Specification.

[3GP13e] 3GPP Tech: *3GPP Tech. Spec. 36.413 version 11.3.0: LTE; Evolved Universal Terrestrial Radio Access Network (E-UTRAN); S1 Application Protocol (S1AP)*, 2013. Technical Specification.

[3GP13f] 3GPP Tech: *Evolved General Packet Radio Service (GPRS) Tunnelling Protocol for Control Plane (GTPv2-C). Ts 29.274, 3rd Generation Partnership Project (3GPP)*, 2013. Technical Specification.

[CIS14] CISCO: *Cisco Visual Networking Index Forecast and Methodology*, 2014. `http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html`.

[ETS] ETSI: *Network Functions Virtualisation (NFV)*. `http://www.etsi.org/technologies-clusters/technologies/nfv`.

[Eur15] European Commision: *Advanced 5G Network Infrastructure for the Future Internet Public Private Partnership in Horizon 2020*

- Creating a Smart Ubiquitous Network for the Future Internet, Ref. Ares(2014)327845 - 10/02/2014, 2015. http://cordis.europa.eu/docs/projects/cnect/5/317105/080/deliverables/001-D14Annex1Advanced5GNetworkInfrastructurePPPinH2020FinalNovember2013pdf.pdf, accessed on 9th Nov, 2015.

[F5 16] F5 Solutions: *Load Balancer*, 2016. https://f5.com/glossary/load-balancer/, accessed on 1st Nov, 2015.

[Fir] Firmin, Frederic: *3GPP MCC The Evolved Packet Core.* Radisys. http://www.3gpp.org/technologies/keywords-acronyms/100-the-evolved-packet-core, Accessed on 2/2/2016.

[Fra] Fraunhofer FOKUS: *Open5GCore - The Next Mobile Core Network Testbed Platform.* http://www.open5gcore.org/index.html.

[Fuj14] Fujitsu Network communication Inc: *The benefits of Cloud-RAN architecture in Mobile Network expansion*, 2014. http://www.fujitsu.com/us/Images/CloudRANwp.pdf.

[GSM13] GSMA: *LTE and EPC Roaming Guidelines, version 10.0*, 2013. http://www.gsma.com/newsroom/wp-content/uploads/2013/07/IR.88-v10.0.pdf, Accessed on 2/2/2016.

[Jun15] Juniper Research: *Network Optimisation Critical to Redress Balance*, 2015. http://www.juniperresearch.com/press/press-releases/mobile-operator-costs-pass-$800bn-annually-as-marg, accessed on 1st Nov, 2015.

[Koc13] Kocur, Jakub: *MSc-thesis on Design and Implementation of an eNodeB Emulator for the 3GPP Release 11 Evolved Packet System.* TU Berlin in cooperation with Warsaw University of Technology, 2013.

[KS11] K Suzuki, K Kunitomo, T Morita: *Technology Supporting Core Network (EPC) Accommodating LTE, a report by NTT Docomo.* NTT Docommo, 2011. https://www.nttdocomo.co.jp/english/binary/pdf/corporate/technology/rd/technical_journal/bn/vol13_1/vol13_1_033en.pdf,NTT Docommo Technical Journal Vol 13 No 1.

[Mul09] Mulligan, Magnus Olsson Shabnam Sultana Stefan Rommer Lars Frid Catherine: *Architecture overview in SAE and the Evolved Packet Core Driving the Mobile Broadband Revolution, Pages 23-29.* Academic Press, 2009. ISBN: 9780080888705.

[Ope12] Open Networking Foundation: *Open Networking Foundation: Software-Defined Networking: The New Norm for Networks. White paper*, 2012. http://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf.

[RK09] Ralf Kreher, Karsten Gaenger: *LTE Signaling, Troubleshooting, and Optimization,E- UTRAN/EPC Signaling, Pages 131-186.* Wiley-Blackwell, 2009. ISBN: 9780470689004.

[STK11] Suyash Tripathi, Vinay Kulkarni and Alok Kumar: *E-UTRAN LTE E-UTRAN and its Access Side Protocols.* Radisys, 2011. `http://go.radisys.com/rs/radisys/images/paper-lte-eutran.pdf`.

[Vla13] Vlad, Valentin: *Master thesis on Design and implementation of an openflow protocol stack into openepc.* TU Berlin, 2013.

[WIK16] WIKI: *Load balancing (computing)*, 2016. `https://en.wikipedia.org/wiki/Load_balancing_%28computing%29`, accessed on 3rd Feb, 2016.

[Wir] The Wireshark Foundation: *Wireshark, the worlds foremost network protocol analyzer.* `http://www.wireshark.org/`.