# Scalability Analysis of LTE-EPC in an NFV Environment


**Faiber Botina Anacona**

**Kelly Tatiana Tobar Ortega**

*Universidad del Cauca*

*Facultad de Ingeniería Electrónica y Telecomunicaciones*

*Departamento de Telemática*

*Popayán*

*2018*

# Scalability Analysis of LTE-EPC in an NFV Environment

Faiber Botina Anacona
Kelly Tatiana Tobar Ortega

Trabajo de Grado presentado a la Facultad de Ingeniería
Electrónica y Telecomunicaciones de la
Universidad del Cauca para obtener el título de:
Ingeniero en Electrónica y Telecomunicaciones

Advisor: MSc. Carlos Hernán Tobar Arteaga
Co-Advisor: PhD. Oscar Mauricio Caicedo Rendón

*Universidad del Cauca*
*Facultad de Ingeniería Electrónica y Telecomunicaciones*
*Departamento de Telemática*
*Popayán*
*2018*

# Acknowledgements

First and foremost, we would like to thank God for his help during this work as a part of his generous blessing throughout our life. We express our deepest gratitude to our families, parents, and siblings for their infinite love and support throughout the years. They have dedicated their lives to our education, and we would like to dedicate this work to them. We are also grateful to all our friends for their enthusiastic encouragement during our training process as people and professionals. We would like to thank our academic advisor, Professor MSc. Carlos Hernán Tobar Arteaga, for his support directing our research. We would like to thank our co-advisor, Professor PhD. Oscar Mauricio Caicedo Rendón for his invaluable help and support during our research.

# Abstract

The scaling is a pivotal task to address workload variation that affects the performance of mobile networks. In the literature, several approaches propose scaling methods to the mobile Evolved Packet Core (EPC). However, so far, none of these approaches have exploited the benefits of elastic scaling to achieve an adaptive EPC in front of workload variations. In this work, we demonstrate that Network Functions Virtualization (NFV) allows incorporating elastic scaling to EPC aiming to address workload variations, and thus, to improve the network performance and the resources utilization. In particular, we present an elastic scaling mechanism in an NFV-based LTE-EPC, hereinafter, just called virtualized EPC (vEPC), a deployment of a vEPC that supports elastic scaling capability and a performance evaluation of each vEPC entity regarding throughput and latency. The evaluation results reveal a significant increase ($\sim$300%) in throughput and an important decrease ($\sim$70%) of latency when the vEPC uses our elastic scaling mechanism. These results corroborate the importance of including elastic scaling capability to improve the vEPC performance.

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| **4G** | Fourth Generation |
| **BSS** | Business Support Systems |
| **B-vEPC** | Bombay-virtualized Evolved Packet Core |
| **CAPEX** | Capital Expenditure |
| **CPU** | Central Processing Unit |
| **DS** | Data Store |
| **eNodeB** | evolved Node B |
| **EPC** | Evolved Packet Core |
| **HSS** | Home Subscriber Server |
| **IP** | Internet Protocol |
| **LB** | Load Balancer |
| **LTE** | Long Term Evolution |
| **NF** | Network Function |
| **NFV** | Network Functions Virtualization |
| **NFVI** | Network Functions Virtualization Infrastructure |
| **MANO** | Management and Orchestration |

| | |
|---|---|
| **MME** | Mobility Management Entity |
| **OPEX** | Operational Expenditure |
| **OSS** | Operations Support System |
| **PGW** | Packet Data Network Gateway |
| **QoS** | Quality of Service |
| **RAN** | Radio Access Network |
| **RAM** | Random Access Memory |
| **RMA** | Reliability, Maintainability and Availability |
| **SDN** | Software Defined Networking |
| **UE** | User Equipment |
| **vEPC** | virtualized Evolved Packet Core |
| **VNF** | Virtualized Network Function |
| **VM** | Virtual Machine |
| **vMME** | virtualized Mobility Management Entity |

# Chapter 1

# Introduction

## 1.1 Problem Statement

Mobile cellular networks have been rapidly deployed globally and, with the introduction of Long Term Evolution (LTE), service providers are looking to offer high data rates, multiple services and higher Quality of Service (QoS) [1]. In LTE networks, access and core refer to the Evolved Packet System. The core corresponds to the Evolved Packet Core (EPC) that is responsible for controlling the signaling and data traffic of mobile network; according to Cisco's Visual Networking Index projections, for 2021, there will be around 5500 millions of mobile phones in the world, and LTE traffic will grow 11 times between 2016 and 2021 [2].

EPC has not evolved as fast as the growing demand for speed, number of accesses and new applications [3]. Thus, the core of LTE networks faces significant challenges, including increasing capacity, supporting various types of traffic and accelerating the Time-to-Market for current and new applications. These challenges can be addressed with new paradigms such as Network Functions Virtualization (NFV) [4]. In NFV, the network functions are implemented in software and then run as virtualized instances, allowing the deployment of network functions in commodity hardware [5]. NFV can be a solution to the challenges of

LTE-EPC because it offers mobile operators the ability to virtualize network entities and manage their virtual and physical infrastructure efficiently [6].

A fundamental benefit of NFV is to provide the scalability that is the ability of a network to be expanded/contracted without requiring significant changes in its architecture, thus, achieving high network performance [7]. There are different scaling methods (*i.e.,* horizontal, vertical, and elastic). Horizontal scaling increases the number of instances of the network elements. Vertical scaling increases the capacity of the instances. Elastic scaling scales in both dimensions when the network load increases (*e.g.,* traffic and number of users) [8]. By using NFV, virtualized network capabilities can be dynamically scaled on demand to meet a given network performance requirement [9].

By analyzing network performance, it is possible to observe the behavior of a network service, identify faults and determine the metrics to improve QoS. Therefore, mobile operators are studying the possibility of scaling their networks through NFV as an alternative to improve network performance [10]. Analyzing the network performance of an LTE-EPC with elastic scaling capability in an NFV environment is important because it allows mobile operators to establish which network core entities require to be scaled, determine a trade-off between vertical and horizontal scaling, and thus make better use of available resources, and identify, for example, where are network bottlenecks [5].

In the literature, to provide EPC scalability, diverse methods have been used. Some authors have horizontally scaled the EPC entities [11, 12, 13, 14, 15, 16, 17]. In turn, vertical scaling has been less incorporated in EPC [18]. Nevertheless, the above research is only focused on scaling EPC in one dimension (*i.e.,* horizontal or vertical) and does not perform a performance assessment of a mobile network core with elastic scalability capability in an NFV environment. Thus, the benefits of scaling vertically and horizontally are not determined, which could be, for example, to simplify management with the centralization of the workload in entities with higher capacity and to allow high network availability with the distribution of the workload in multiple entities.

In summary, several works have proposed to apply scaling methods to LTE-EPC

in NFV environments. However, to the best of our knowledge, none of the proposals analyze the network performance of an LTE-EPC with elastic scalability capability in an NVF environment. Considering the above-mentioned, we propose the following research question:

**What is the elastic scalability behavior of an LTE-EPC in an NFV environment?**

To answer this research question, we present the following objectives.

## 1.2 Objectives

### 1.2.1 General

- Analyze the horizontal and vertical scalability of an LTE-EPC in an NFV environment.

### 1.2.2 Specifics

- Adapt an LTE-EPC emulator in an NFV environment.
- Incorporate elastic scalability capability into an NFV-based LTE-EPC.
- Evaluate at emulation level the performance of an NFV-based LTE-EPC regarding throughput and latency.

## 1.3 Research Contributions

The key contributions provided in this work are:

- An elastic scaling mechanism in a vEPC that determines the scaling method to support the workload variation in vEPC. This mechanism is formed by three modules (*i.e.,* Data Collection, Scaling Decision, and Scaling Execution) and an algorithm that defines its operation.
- A deployment of a vEPC that supports elastic scaling capability. To perform this deployment, we deployed an open source vEPC from the Indian Institute of Technology Bombay.
- A performance evaluation of each vEPC entity when it supports elastic scaling capability. This performance evaluation was performed regarding throughput, latency, CPU usage, and RAM usage.
- The Grupo de Ingeniería Telemática (GIT) and the training research group ComsoCauca of the University of Cauca have obtained knowledge in mobile networks in an NFV environment. In particular, this undergraduate work supports the doctoral work of our advisor Carlos Hernán Tobar Arteaga and was used as a part of the content of the subject Recent Topics in Networking. Also, our deployment of the open source vEPC from the Indian Institute of Technology Bombay was used as a test scenario by the undergraduate work Automatic IP Traffic Classification in an NFV-based environment.

## 1.4 Publications

The work presented in this monograph was reported to the scientific community by paper submission to a renowned journal.

- **Kelly Tatiana Tobar Ortega, Faiber Botina Anacona,** Carlos Hernan Tobar Arteaga, Oscar Mauricio Caicedo Rendon. **Elastic Scaling in the Virtualized Evolved Packet Core.** IEEE Transactions on Network and Service Management.
  - Status: Submitted
  - Classification: A1 (COLCIENCIAS) and Q1 (JCR)

## 1.5 Document Structure

This document has been divided into chapters described below.

- Chapter 1 presents the **Introduction** that contains the Problem Statement, Objectives, Research Contributions, Publications and the structure of this document.
- Chapter 2 presents the **Background** about the relevant topics concerning our research. These topics include NFV, EPC, network performance and scalability.
- Chapter 3 presents the **Related Work** that describes the research work closer to our proposal.
- Chapter 4 presents the **Elastic Scaling Mechanism**. This chapter exposes the motivating scenario and introduces our elastic scaling mechanism formed by three modules (*i.e.,* Data Collection, Scaling Decision, and Scaling Execution) and an algorithm that defines its operation.
- Chapter 5 presents the **Evaluation and Analysis** of the vEPC performance with elastic scaling in an NFV environment regarding throughput and latency.
- Chapter 6 presents **Conclusions** and **Future work**. This chapter provides the main conclusions of our work and important implications for future work.

# Chapter 2

# Background

In this chapter, we present the background related to our approach. First, we describe NFV and its architecture. Second, we introduce EPC and its components. Third, we present important concepts about network performance. Fourth, we introduce relevant concepts associated with elastic scalability.

## 2.1 Network Functions Virtualization

NFV is a concept that transforms the way network operators define the architecture and operation of their network infrastructure since it uses virtualization technologies to deploy network functions on hardware commodity (*e.g.,* servers, switches, and storage) [4]. NFV brings significant benefits such as: to incorporate scalability capacity into the network, reduce Capital Expenditure (CAPEX) and Operational Expenditure (OPEX), improve the management of the network, and reduce time-to-market to deploy new network applications [19].

NFV consists of three main elements (see Figure 2.1): Network Function Virtualization Infrastructure (NFVI), Virtualized Network Functions (VNFs) and NFV Management and Orchestration (MANO). NFVI combines hardware resources and software that make up the environment where VNFs are implemented [20].

A Network Function (NF) is a functional block within a network infrastructure that has defined external interfaces and functional behavior, therefore, a VNF is an NF that is deployed on virtual resources such as virtual machines (VMs) or containers technology [4]. Virtual resources are an abstraction of the physical resources of the network, storage, and computing which is achieved through a virtualization layer [21].

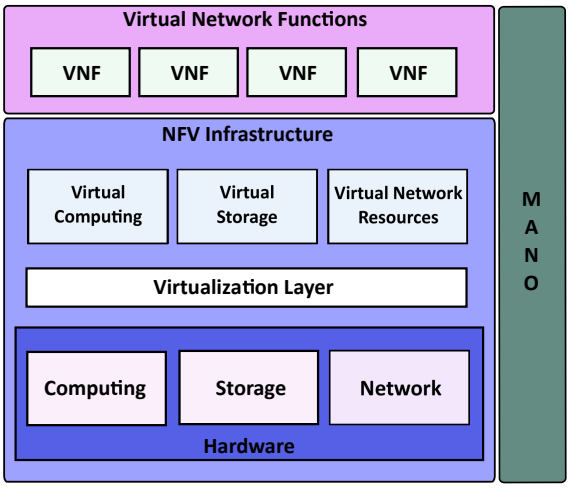



Figure 2.1: NFV architecture
Source: [5]

NFV MANO provides the necessary functionality for the configuration and management of the VNFs and the orchestration of the physical resources and software that support the virtualization of the infrastructure. It also defines the interfaces for communication between the different components, as well as coordination with traditional network management systems such as Operations Support System (OSS) [22] and Business Support Systems (BSS) [23] to enable the management of VNFs [24].

## 2.2 Evolved Packet Core

The current standard for 4G (*Fourth Generation*) networks is LTE [25]. EPC is LTE core defined by 3rd Generation Partnership Project for the provision of 4G

services with the feature of interoperating with 2G and 3G services [26]. The main functions of EPC are [27]:

- Manage the network congestion to provide QoS to applications such as voice and video in real time.
- Authenticate and authorize user traffic to facilitate mobility management in different access contexts.
- Manage terminal mobility between base stations connected to the EPC and thus, to guarantee users constant network connectivity.
- Add traffic from different access networks to a single Internet gateway to maintain a traffic density between the radio access network and the network core.

EPC is composed of four primary entities (see Figure 2.2) [25]: Mobility Management Entity (MME), Home Subscriber Server (HSS), Serving Gateway (SGW) and Packet Data Network Gateway (PGW).

- MME is the control entity in charge of managing the authentication and configuration of the User Equipment (UE) session. This entity handles the signaling related to mobility and security. MME is responsible for the tracking and the paging of UE in idle-mode.
- HSS is the repository that contains the information related to the end-users (*e.g.,* authentication keys and UE capabilities). This entity provides support in mobility management, user authentication, and access authorization.
- SGW and PGW compose the EPC data plane that is responsible for routing the packets. SGW supports IP data traffic between the Radio Access Network (RAN) and PGW. Furthermore, SGW configures the uplink and downlink tunnels for data transfer. PGW sends EPC data traffic to external IP networks. It is noteworthy that SGW and PGW participate in the control operations as well as MME.

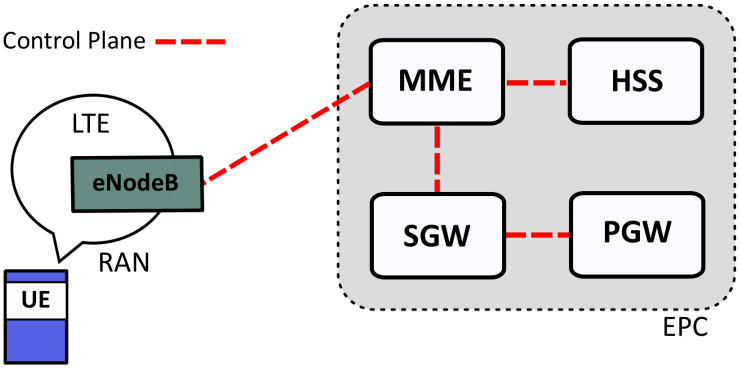



Figure 2.2: Evolved Packet Core
Source: [28]

The EPC control plane operates as follows (see Figure 2.3). When a UE requires to connect to an LTE network makes an attach request and when it needs to disconnect makes a detach request [29]. To perform a UE attach process, a radio connection must first be established between UE and eNodeB, then UE sends an attach request to MME via eNodeB; the attach request includes the International Mobile Subscriber Identity that identifies UE. The attach process involves sub-process such as user authentication, security and session setup, as follows: MME performs UE authentication by using HSS. HSS updates the UE data and sends a response to MME. Then the security setup includes key encryption to ensure communication between UE and MME. After the successful security setup, in session setup, a default “bearer” is created for UE through the packet core. During this process, an IP address is assigned to UE from PGW, and Tunnel Endpoint Identifier values for this “bearer” are exchanged among eNodeB, MME, SGW, and PGW. At the end of this process, a tunnel is established for data traffic between UE and PGW via SGW. Finally, when a UE sends a detach request, the entire UE state is cleared from all EPC entities. Then the MME sends the detach response to the UE via eNodeB.

There can be situations in which a UE movement could trigger a change in network IP address. In such situations, UE mobility process are initiated to handover the UE to a new eNodeB according to the current location of the UE. In a handover process, the source eNodeB after getting measurement reports from the UE de-

cides to handover to a target eNodeB and makes handover request to the source MME. After receiving the handover request, the source MME and the target MME communicate and decide to handover the UE connection to the new eNodeB and the SGW transfers its tunnel to the target eNodeB. After the successful handover, resources allocated for the UE are removed at the source side.

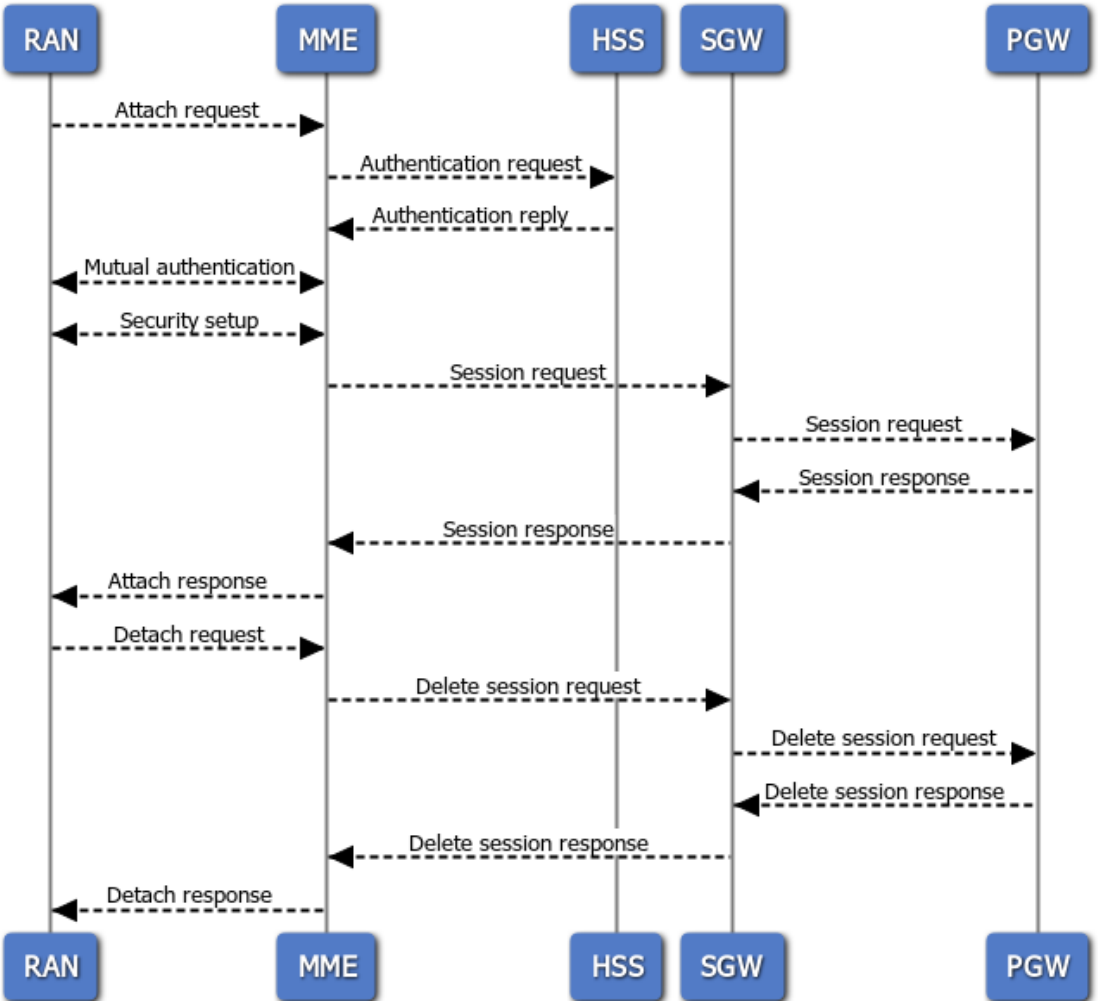



Figure 2.3: EPC control plane operation
Source: [30]

## 2.3 Network Performance

Performance is the set of capacity, latency and Reliability, Maintainability, and Availability (RMA) levels in a network [31]. Capacity includes bandwidth and throughput metrics, latency includes Round Trip Time (RTT) and delay variation, and RMA includes features such as reliability, maintainability, and availability; the metrics analyzed depend on the type of network [32]. The main metrics for evaluating the performance of a network are [33]:

- **Capacity** is the measure of the network ability to transfer information (*e.g.,* voice, data and video).
- **Latency** is the measure of the time difference in the transmission of information (*e.g.,* bit, byte, frame, and packet) from source to destination.
- **Reliability** is the frequency of failure of the network and its components.
- **Maintainability** is the measure of the time to restore the system to full operational status after it has experienced a fault.
- **Availability** is the relationship between the frequency of critical failures and the time to restore service.

It is to highlight that in the EPC control plane context, throughput refers to the number of registrations (*i.e.,* attach and detach processes) successfully completed by the EPC per second, and latency refers to the time that a UE takes to perform the attach and detach processes [34]. Furthermore, in the NFV context is important to include in the performance evaluation metrics such as CPU and RAM [35].

- **CPU** is the percentage of use of each processing core assigned to a virtual instance.
- **RAM** is the measure of the consumption of Megabytes assigned to a virtual instance.

To improve the behavior of network performance metrics, scaling methods can be employed to increase the network ability to adapt to variations in the network workload [36]. Scaling methods allow to a network to support more users, offer new services, and provide high QoS to meet a given network performance requirement [37].

## 2.4 Scalability

Scalability is the network ability to continue to function with acceptable performance when the workload has been significantly increasing [38]. If the target of a network is to increase the number of users that support while maintaining its current performance, it has to evaluate the possible options: by using a distribution of hardware and software or more powerful hardware.

In the NFV context, scalability is employed in VNFs that can be dynamically scaled according to network performance requirements [39]. The main scaling methods are [40]:

**Vertical scaling (*up*/*down*)** is the ability to scale by expanding (*up*)/ decreasing (*down*) a resource that is assigned to a VNF (*e.g.,* memory, CPU capacity, and storage). This scaling method does not imply any significant modification at the structural level, which makes it a good option because it makes easy its management. However, this scaling method has a limiting aspect, by increasing the power based on hardware capacity, there will come a time when there will be a hardware limitation [41].

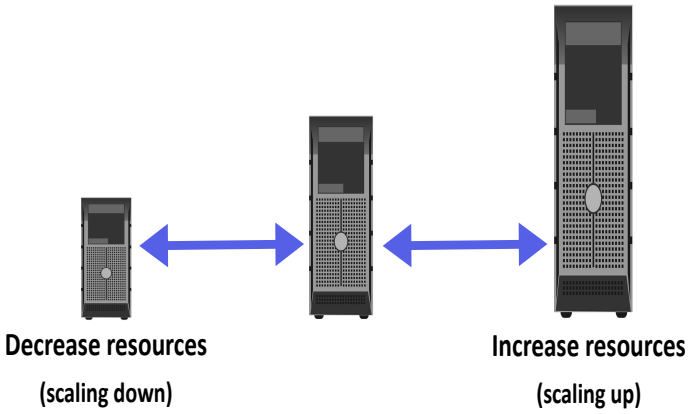



Figure 2.4: Vertical scaling
Adapted from source: [42]

**Horizontal scaling (*out/in*)** is the ability to scale by adding (*out*)/removing (*in*) instances (*e.g.,* VMs, processing and storage equipment). This scaling method is about improving network performance from an overall architectural improvement perspective by distributing the workload between multiple instances, as opposed to increasing the power of a single instance. As the main limitation, this scaling method involves a major modification in the design of the network architecture [43].

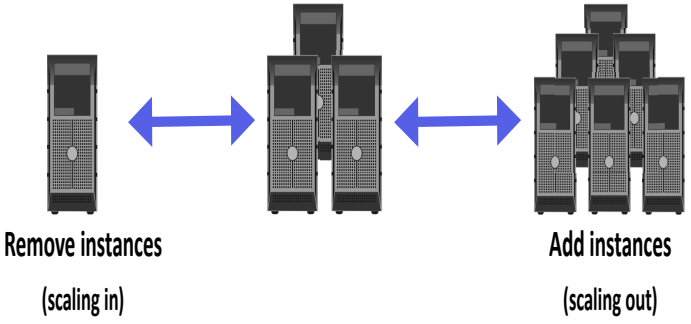



Figure 2.5: Horizontal scaling
Adapted from source: [42]

**Elastic scaling** is the ability to scale in both dimensions (*i.e.,* horizontal and vertical) [44]. This scaling method combines the benefits of vertical and horizontal

scaling to adapt to workload variations and make good use of available resources. The network needs to be able to increase or decrease hardware resources of the instances or including/deleting the number of virtual instances [45].

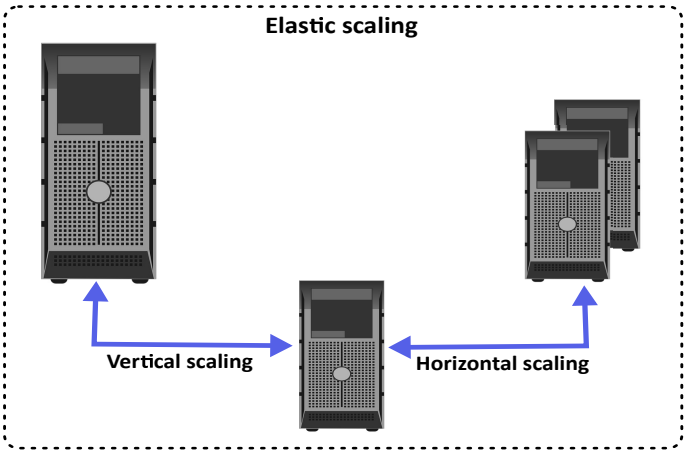



Figure 2.6: Elastic scaling
Adapted from source: [42]

# Chapter 3

# Related Work

This chapter describes the related work closer to our proposal. First, we present the related work about the use of horizontal scaling in EPC in an NFV environment. Second, we introduce the related work about the use of vertical scaling in EPC in an NFV environment. Third, we expose some final remarks. It is important to highlight that to the best of our knowledge, up to now, elastic scaling has not been deployed on a mobile network core.

## 3.1 Horizontal Scaling

Some investigations have proposed an MME architecture to support horizontal scaling based on three components: a front-end, a set of workers, and a database. Front-end behaves like a proxy that maintains interfaces to other EPC entities. One or more workers nodes that process the control traffic. The database stores the worker state, which makes the workers behave as stateless ones. Thus, the front-end and database are transparent to other entities (*e.g.,* SGW and PGW) and the workers scale horizontally to face the network workload [11, 12]. Following the same architecture, MME, SGW, and PGW can be deployed as clusters of replicas that share the incoming workload. Each EPC entity is composed of replicas (*i.e.,* workers), a load balancer (*i.e.,* front-end) that distributes the in-

coming workload among the replicas by using a round-robin policy and a shared database that stores the replicas state with several state synchronization options (*i.e.,* no sync, session sync and always sync). The performance evaluation of this architecture presents that always sync option imposes a performance penalty of 71% regarding latency and 75% regarding throughput compared to no sync option [16].

In previous work of our advisors [13], they propose an adaptive mechanism that can perform horizontal scaling of vEPC by considering only MME. They consider a virtualized MME (vMME) that can scale by increasing or decreasing the number of instances of its service logic, and a network manager responsible for selecting the number of instances to perform the scaling. The mechanism combines Q-Learning and system models based on Gaussian Processes. These models allow estimating the performance of a network service, and a Q-Learning agent improves its scaling policy by using them. By simulations, they evaluate their mechanism to manage variations of Mean Response Time in a mobile network core, corroborating that their mechanism is more accurate than approaches based on static threshold rules and Q-Learning without the use of models for policy improvement.

SCALE is a framework for performing horizontal scaling of MME. The MME functionality is re-designed into two parts: a load balancer and an MME processing cluster. The authors by using consistent hashing take advantage of the access patterns of available devices registered in MME to intelligently reduce memory usage. Thus, the re-use of sessions created by the entity allows MME to allocate fewer resources to processing link creation requests for data and control traffic. The evaluation results reveal that SCALE reduces the processing delay of control plane messages from 1 second to 250 ms [14].

Cloud Native Solution for Mobility Management Entity (CNS-MME) is a proposal for performing horizontal scaling of MME based on a micro services architecture. This architecture deploys a CNS-MME as a virtualized micro services cluster where a load balancer separates the control processes (*i.e.,* attach and detach) and delivers them separately to groups of VNFs intended for each process. The

other entities (*i.e.,* HSS, SGW, and PGW) are also deployed as VNFs in containers technology. CNS-MME is highly available and supports automatic scaling to horizontally scale the micro-service required for load balancing. The authors determine that the CNS-MME performance is higher (approx. 7%) than a monolithic MME architecture, and also reduces processing resource consumption ( approx. 26%) [15].

SGW and PGW can be scaled horizontally in a virtualized environment by considering two approaches. The first one, it is the deployment of SGW and PGW within a single virtual machine that handles data and control traffic (combined model). The second one, it is the deployment of SGW and PGW that relies on separating the processing of the control and user plane on different VMs (decomposed model). The authors evaluate the benefits of the dynamic adaptation of SGW and PGW resources to face the traffic demand for the two approaches. This evaluation reveals that decoupling data and control plane in SGW and PGW provides better adaptability to traffic fluctuation [17].

## 3.2 Vertical Scaling

A mechanism to scale a cloud-based 5G mobile system vertically can trigger scaling by using threshold values. The authors by using a decision-making module decide when triggering a vertical scaling based on Mean Opinion Score and the resources usage regarding CPU and RAM usage. This decision-making module indicates when to extend or reduce the physical resource allocated per instance while preventing a service disruption [18].

## 3.3 Final Remarks

In the NFV research, several works have proposed the use of scaling methods in EPC. Most of the works have used horizontal scaling since it provides high availability and performance because of the distribution of the workload in multiple

instances of the EPC entities. In turn, vertical scaling represents a good cost-benefit ratio for mobile operators because the increase in resources per EPC entity makes easier the network management. To the best of our knowledge, up to now, elastic scaling has not been deployed on a mobile network core. The research about the use of horizontal or vertical scaling in vEPC is presented below.

| Investigations | Environment | Entity | Scaling Method | | |
|---|---|---|---|---|---|
| | | | Horizontal Scaling | Vertical Scaling | Elastic Scaling |
| [11] | EPC | MME | ✓ | | |
| [12] | EPC | MME | ✓ | | |
| [13] | EPC | MME | ✓ | | |
| [14] | EPC | MME | ✓ | | |
| [15] | EPC | MME | ✓ | | |
| [16] | EPC | MME, SGW, PGW | ✓ | | |
| [17] | EPC | SGW, PGW | ✓ | | |
| [18] | EPC | MME | | ✓ | |
| Our proposal | EPC | MME, SGW, PGW | ✓ | ✓ | ✓ |

Table 3.1: Scaling proposals in vEPC with scaling methods

Table 3.1 introduces the most important related work to our approach. This table reveals four facts:

- Most of the proposals focus on scaling the MME.
- Most of the proposals have performed horizontal scaling.
- The work that applies vertical scaling only focus on MME.
- To the best of our knowledge, none of the works have incorporated elastic scaling in EPC.

Unlike the above works, we propose an elastic scaling mechanism that allows EPC entities to deal with workload variations. Also, the performance analysis of

individual EPC entities to determine which entity requires to be scaled to improve the EPC performance.

# Chapter 4

# Elastic Scaling Mechanism

In this chapter, first, we expose a motivating scenario. Second, we present the elastic scaling mechanism formed by three modules (*i.e.,* Data Collection, Scaling Decision, and Scaling Execution) and an algorithm that defines its operation.

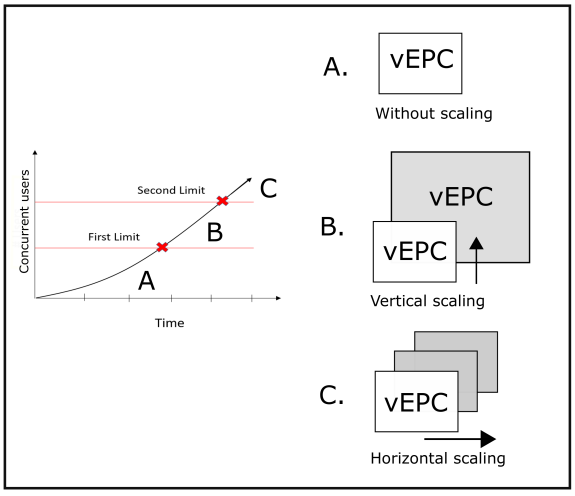
## 4.1 A Motivating Scenario

Before introducing our elastic scaling mechanism, let us consider a vEPC scenario. Figure 4.1 presents vEPC with specific resources in memory, number of cores, and storage. These resources allow vEPC to handle the control traffic. When the number of concurrent users of vEPC increases, the set of static resources assigned to vEPC could not support the workload. As a consequence, the end-users QoS will degrade. When the number of concurrent users decreases, vEPC is usually over-provisioned and, thus, it wastes resources.

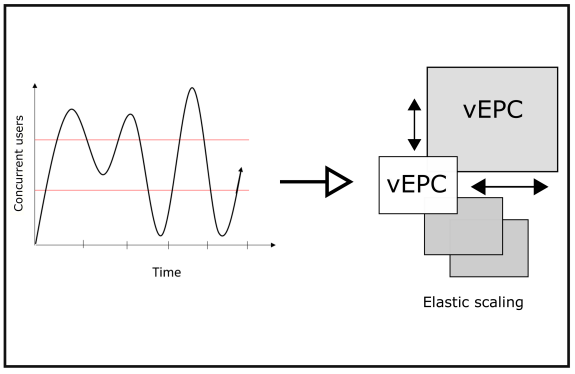
Figure 4.1a illustrates a particular case where the number of concurrent users increases and therefore, the amount of control traffic supported by vEPC. To address the above scenario, network operators can perform different actions. The first one, it is to support the workload when the number of concurrent users is below the first limit (Region A) by a vEPC without scaling. The second action, it is to

apply vertical scaling to vEPC when the number of concurrent users exceeds the first limit and is below the second limit (Region B). Third, it is to incorporate horizontal scaling to vEPC when the number of concurrent users passes the second limit (Region C).

Figure 4.1b presents a daily case where the workload of vEPC varies during the day. To face the workload variation is necessary to apply elastic scaling to vEPC and thus, take advantage of both vertical and horizontal scaling. In this paper, we propose an elastic scaling mechanism that determines the scaling method to support the workload variation in vEPC and avoid the resources wasting.




(a) Vertical and horizontal scaling




(b) Elastic scaling

Figure 4.1: Motivating scenario

## 4.2 General Operation of the Mechanism

Figure 4.2 presents the high-level operation of our mechanism. First, we define regions of performance behavior (*i.e.,* regions I, II, and III) to trigger our elastic scaling mechanism. These regions are initially established by making ad-hoc measurements and performing a performance evaluation from the variation of the number of concurrent users to determine threshold values of the number of concurrent users (*i.e.,* $C_{users1}$ and $C_{users2}$) and the number of registrations per second (*i.e.,* $Th_1$ and $Th_2$) supported by vEPC with static resources allocation. Second, when the regions are defined the mechanism operates as follows. If the vEPC performance is in the region I, the mechanism does not perform any action because vEPC with static resources allocation can support the workload. If the vEPC performance passes from Region I to Region II, the vertical scaling is activated. Thus, it is necessary to increase the resources per VNF to face the workload. If the performance passes from Region II to Region III, the horizontal scaling starts, and, it is required to create one or more VNFs to support the workload. If the performance passes from Region III to Region II, it is necessary to delete one or more VNFs to handle the workload. If the vEPC performance passes from Region II to Region I, the vertical scaling is activated, and, it is required to decrease the resources per VNF to support the workload. Finally, the performance at the output of the mechanism is feedbacked for continuous evaluation and, thus, tunning the decision thresholds when needed. Thus, vEPC can handle the workload variations by using our elastic scaling mechanism.

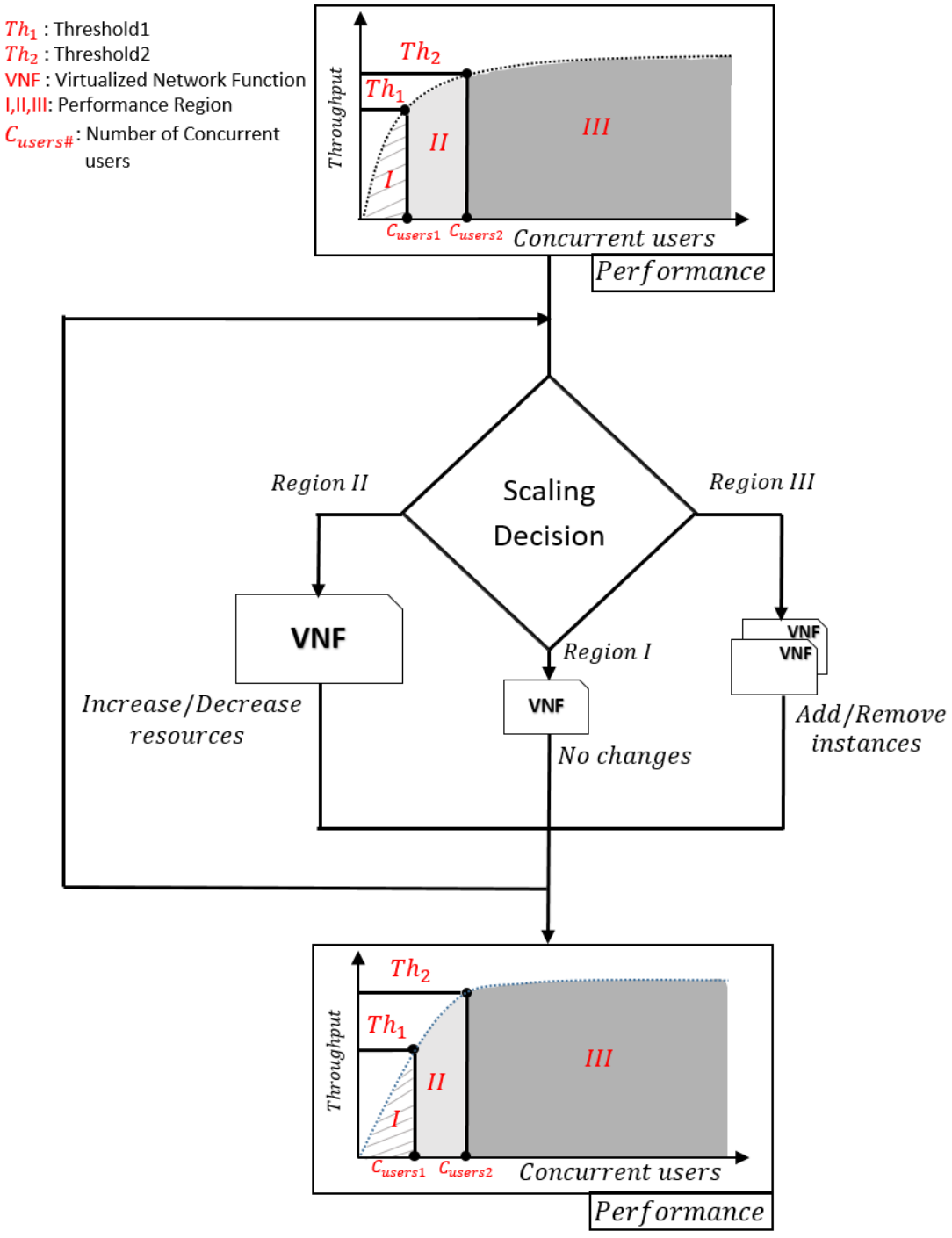



Figure 4.2: Elastic scaling mechanism

## 4.3 Elastic Scaling Algorithm

By using pseudo-codes, we detail the processes to perform the elastic scaling in vEPC (see Algorithm 1). Our algorithm has as input data a number of concurrent users (*n*), vEPC performance metrics, vEPC performance thresholds, and a configuration of vEPC entities. The algorithm results are the performance metrics with the adjustments provided by the elastic scaling and a scalability configuration for vEPC.

For each discrete time (*t*) defined by the network administrator, throughput values are monitored for a specific number of concurrent users (*n*). From these values, the network administrator obtains an initial behavior of the vEPC performance and defines the throughput thresholds (*i.e.,* $Th_1$ and $Th_2$) and the number of concurrent users that generate them (*i.e.,* $C_{users1}$ and $C_{users2}$). These thresholds define the regions of performance behavior where a scaling method must be incorporated.

Our algorithm operates as follows. If throughput value is lower than the first throughput threshold ($Th_1$) and the number of concurrent users is lower than the first number of concurrent users threshold ($C_{users1}$), it indicates that the vEPC performance is in Region I and the vEPC with fixed resource allocation must address the workload.

If throughput value is between the two throughput thresholds ($Th_1$ and $Th_2$) and the number of concurrent users is between the two thresholds of concurrent users ($C_{users1}$ and $C_{users2}$), it indicates that the vEPC performance is in region II and vertical scaling must be applied. To perform the vertical scaling, the network administrator determines one vertical scaling configuration and assigns the resources per vEPC entity required to support the workload.

Finally, if throughput value exceeds the second throughput threshold ($Th_2$) and the number of concurrent users is above the second threshold of concurrent users ($C_{users2}$), it indicates that the vEPC performance is in region III and horizontal scaling must be applied. To perform the horizontal scaling, the network administrator

determines one horizontal scaling configuration and assigns the number of vEPC entity instances required to support the workload.

**Data:** Number of concurrent users (*n*), performance metrics (*e.g.,* throughput, latency, CPU usage), performance thresholds (*e.g.,* $Th_1$, $Th_2$, $C_{users1}$, $C_{users2}$), an vEPC scaling configuration
**Result:** Performance metrics (*i.e.,* throughput, latency, CPU usage) and a new vEPC scaling configuration
**for** *each* $t$ **do**
  **if** *throughput* $< Th_1$ **and** *n* $< C_{users1}$ **then**
    Region I;
    Activate vEPC configuration with static resources allocation $\rightarrow$ MME(), SGW(), PGW();
  **else if** $Th_1 \leq$ *throughput* $< Th_2$ **and** $C_{users1} \leq$ *n* $< C_{users2}$ **then**
    Region II;
    Activate vEPC configuration with vertical scaling $\rightarrow$ MME(# resources), SGW(# resources), PGW(# resources);
  **else if** *throughput* $\geq Th_2$ **and** *n* $\geq C_{users2}$ **then**
    Region III;
    Activate vEPC configuration with horizontal scaling $\rightarrow$ MME(# instances), SGW(# instances), PGW(# instances);
**end**

**Algorithm 1:** Elastic scaling

## 4.4 Modules of the Scaling Mechanism

Figure 4.3 illustrates our mechanism formed by three modules called Data Collection, Scaling Decision and Scaling Execution. The Data Collection module is in charge of monitoring and collecting data from vEPC. This module takes measurements of the vEPC performance (*i.e.,* throughput, latency, CPU usage, and RAM usage) when the number of concurrent users varies. Once the performance data are captured, this module stores the data and plots the performance metrics vs. the number of concurrent users to graph the performance behavior of the vEPC and thus, define the maximum workload than vEPC can support. From the performance metrics, the network administrator can determine when vEPC

is becoming saturated, and this allows our mechanism to define the thresholds values to obtain a vEPC performance target. Finally, the Data Collection module provides to Scaling Decision module the information about vEPC performance.

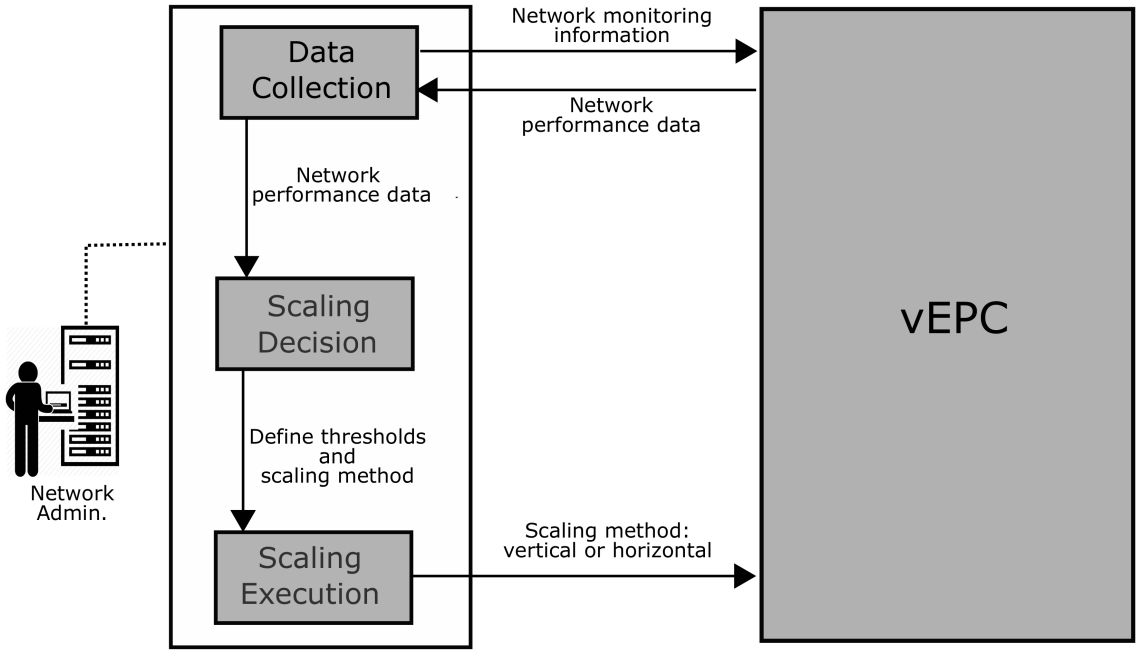



Figure 4.3: Modules of the scaling mechanism

The Scaling Decision module is responsible for selecting the scaling method to deal with the workload variation. In this module our mechanism determines, from the performance data, the thresholds values that define the regions of performance behavior where a scaling method must be incorporated. Thus, this module establishes a maximum number of concurrent users that can be supported by vEPC without scaling, with vertical scaling, and with horizontal scaling. Finally, the Scaling Decision module provides to the Scaling Execution module the scaling method to be applied.

The Scaling Execution module is in charge of applying the scaling method to vEPC. To perform the vertical scaling, our mechanism increases or decreases the resources allocated to vEPC entities. To perform horizontal scaling, our mechanism adds or removes the instances of the vEPC entities. Furthermore, if the

Scaling Decision module does not indicate the need for scaling, the workload must be supported by EPC without scaling. It is important to highlight, that the Data Collection module is monitoring the vEPC performance continuously and, if the workload varies, the Scaling Decision module takes actions again to ensure high performance and proper use of vEPC resources.

# Chapter 5

# Evaluation and Analysis

This chapter presents the evaluation results of our mechanism. First, we expose the test environment. Second, we present the evaluation and performance analysis of the vEPC including baseline, vertical, horizontal, and elastic scaling.

## 5.1 Test Environment

To evaluate our mechanism, we deployed an open source vEPC from the Indian Institute of Technology Bombay (IIT Bombay) [34]. The IIT Bombay vEPC, hereinafter, just called B-vEPC simulates the behavior of a typical EPC that handles control and data traffic. B-vEPC has two versions: version 1.0 that is a monolithic implementation of vEPC, and version 2.0 that is a distributed implementation of vEPC. In our evaluation, version 1.0 was used to define the vEPC baseline and vertical scalability. Version 2.0 was used to analyze the behavior of vEPC with horizontal scalability.

Since we were focused on assessing the elastic scalability from the B-vEPC when it handles control traffic, the performance evaluation was performed regarding throughput, latency, CPU usage, and RAM usage. Throughput refers to the number of registrations (*i.e.,* attach and detach processes) successfully completed by

the B-vEPC per second [46]. Latency is the time that a UE takes to perform the attach and detach processes [47]. Regarding CPU and RAM usage is considered in a saturation level when it reaches values higher than 90% in any entity [48]. To perform the evaluation, we varied the number of concurrent users using B-vEPC to determine its performance behavior. The number of users for each evaluation was 10, 25, 50, 100, and 200. B-vEPC supports a maximum of 200 concurrent users. This number of concurrent users is not a limiting factor because, for example, B-vEPC can generate 16440 registrations during 120 seconds, it means 137 registrations per second. In all evaluation cases, we took the average values for 30 measurements with a 95% confidence level and performed all the tests during 120 seconds. It is important to highlight that to evaluate our mechanism we used the data center provided by the project Telco 2.0 of the University of Cauca. In this way, each B-vEPC entity ran on an Ubuntu 14.04 VM hosted by a Blade Server with two processors Intel Xeon E5-2600 v3. We used as virtualization platform VMware vSphere 6 to run each VM.

## 5.2 Performance with Static Resources Allocation - Baseline

Figure 5.1 illustrates B-vEPC to obtain the baseline, where each entity run on Ubuntu 14.04 VM as a VNF based on the client/server paradigm. Each entity acts as a client by sending requests (*e.g.,* user attach request) to the next entity that processes the request and sends the responses. RAN is a module that combines UE and eNodeB functionalities and generates control traffic to B-vEPC core. RAN does not implement the radio processes that take place between UE and eNodeB; it only focuses on the traffic that B-vEPC core handles. RAN generates multiple threads related to attach and detach processes and also handles communication with MME. B-vEPC responds to UE attach and detach processes for control traffic. Attach is the process used to connect the UE to the B-vEPC core and includes the authentication, security and session setup. Detach is the process used to disconnect UE from the B-vEPC core. It is important to highlight

that MME, HSS, SGW, and PGW are involved in UE attach and detach processes. Table 5.1 presents the resources assigned to B-vEPC for the baseline [34].

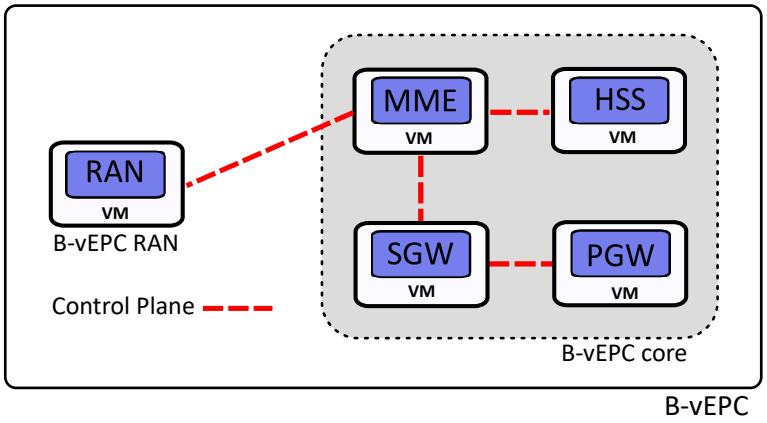



Figure 5.1: B-vEPC for the baseline

| Entity | Hardware Resources | | |
|---|---|---|---|
| | RAM GB | CPU Cores | Storage GB |
| RAN | 4 | 4 | 10 |
| MME | 1 | 1 | 10 |
| HSS | 2 | 1 | 10 |
| SGW | 1 | 1 | 10 |
| PGW | 1 | 1 | 10 |

Table 5.1: Resources assigned to B-vEPC for the baseline

Figure 5.2 depicts the baseline evaluation results of B-vEPC regarding throughput. We perform stress tests by using 200 concurrent users that allow generating 137 registrations per second during 120 seconds that generates a total of 16440 registrations. The evaluation results reveal that the slope of throughput is positive up to 50 concurrent users. From this point, when the number of concurrent users increases, the slope of throughput is less steep. These results mean that the number of attach and detach processes that B-vEPC can complete is constrained after 50 concurrent users.

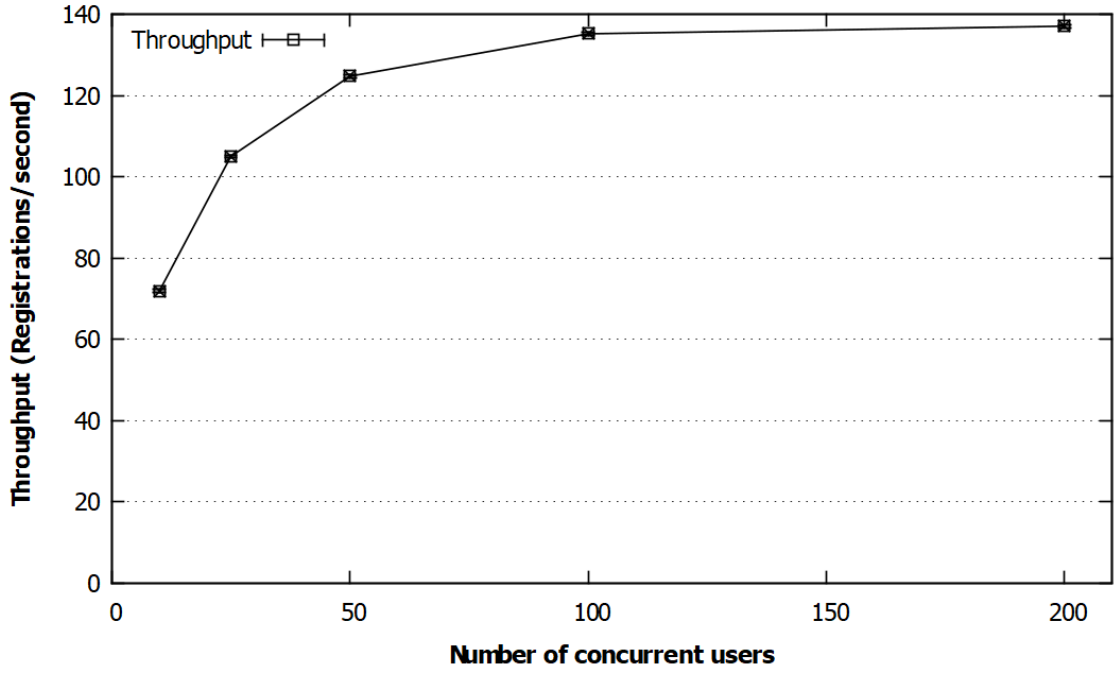



Figure 5.2: Throughput in the baseline evaluation

Figure 5.3 presents the baseline evaluation results of B-vEPC regarding latency. The evaluation results reveal that latency is lower than 100 ms up to 50 concurrent users, however, from this point, the latency increases up to 230 ms. These results are in accordance with the obtained throughput, and it means that after 50 concurrent users, the time to perform the attach and detach processes increases and then, a fewer number of attach and detach processes are successfully completed by B-vEPC.

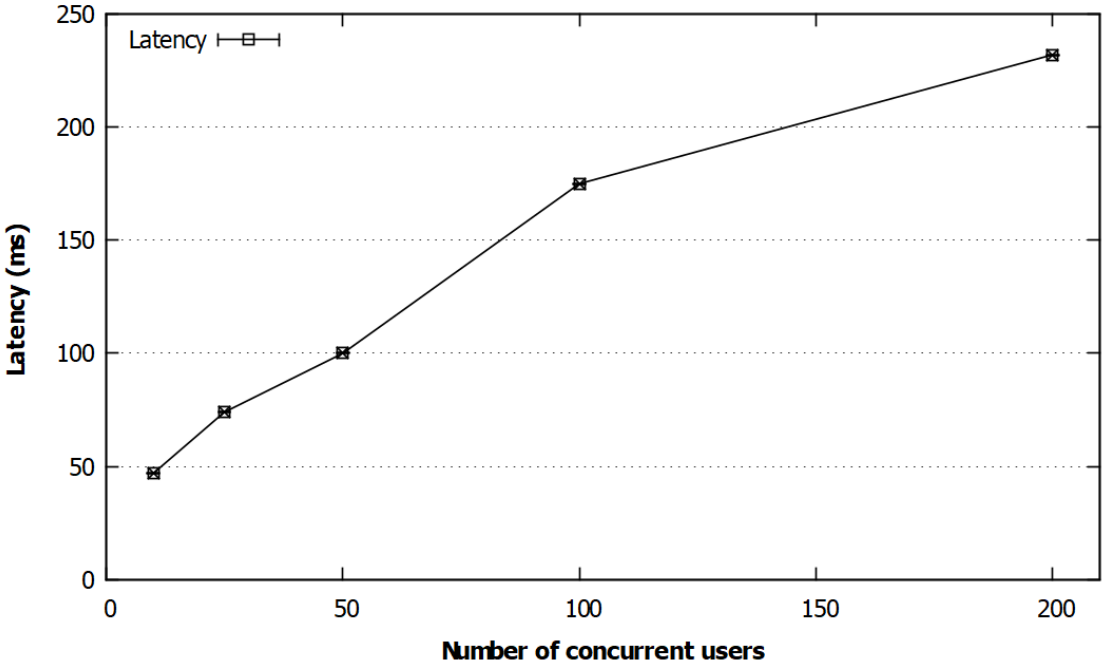



Figure 5.3: Latency in the baseline evaluation

Figure 5.4 depicts the B-vEPC baseline evaluation results regarding CPU usage. These results reveal that the use of CPU in MME and SGW is near to 90% for 50 concurrent users and near to 96% for 200 concurrent users. Nevertheless, the use of CPU in PGW increases as maximum 59%. This CPU behavior indicates that MME and SGW are at saturation levels and, so, the number of control processes that can support is limited.

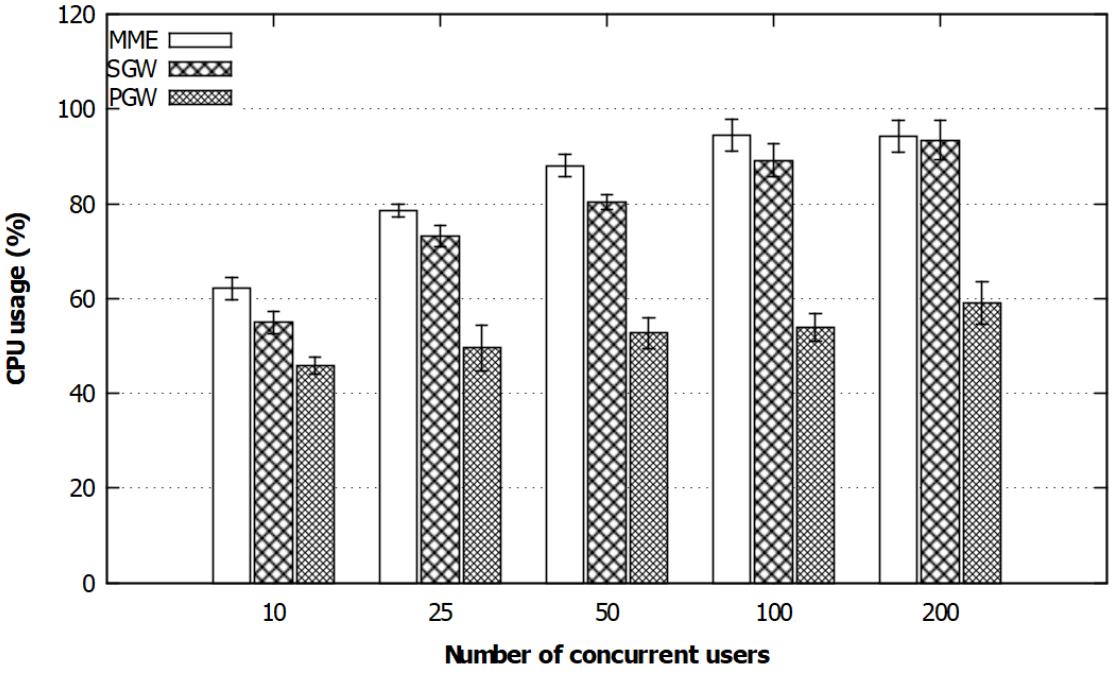



Figure 5.4: CPU usage in the baseline evaluation

Figure 5.5 illustrates the B-vEPC baseline evaluation results regarding RAM usage. These results reveal that the maximum use of RAM is 620 MB. This RAM behavior indicates that RAM is not saturated for any workload variation and, therefore, it is expected that RAM presents a low influence on the performance behavior of the baseline measured for B-vEPC.

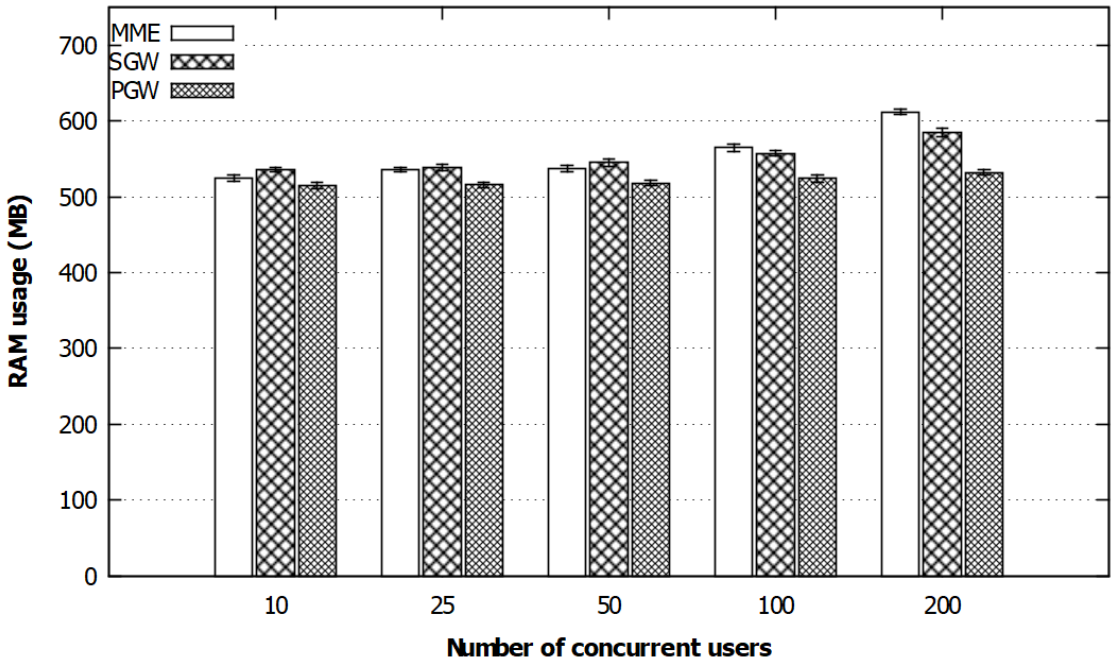



Figure 5.5: RAM usage in the baseline evaluation

To sum up, the baseline evaluation of B-vEPC reveals that:

- MME and SGW play an important role in control plane processes.
- In particular, we identified bottlenecks in MME and SGW that limit B-vEPC performance.
- The RAM has a low influence on the B-vEPC performance.

## 5.3 Performance with Vertical Scaling

To scale B-vEPC vertically, we varied the processing capacity of the EPC entities involved in control plane processes. In particular, we increased the number of processing cores per entity of B-vEPC to analyze the performance behavior regarding throughput, latency, and CPU usage. Note that we did not analyze RAM

because our previous baseline analysis revealed the low incidence of RAM in the B-vEPC performance.

In our vertical scaling evaluation, three cores were deployed per entity giving a total of 27 alternatives. All possible configurations were analyzed, however, for the sake of brevity only the most important results are discussed. Table 5.2 presents the most significant configurations performed per entity. Vx denotes Configuration number 1, 2, 3 or 4 for vertical scaling. The configurations were performed to determine the entities that require vertical scaling to meet the workload based on its use of CPU.

| Entity | Config V1 | | Config V2 | | Config V3 | | Config V4 | |
|---|---|---|---|---|---|---|---|---|
| | RAM GB | CPU Cores | RAM GB | CPU Cores | RAM GB | CPU Cores | RAM GB | CPU Cores |
| MME | 1 | 2 | 1 | 1 | 1 | 2 | 1 | 3 |
| SGW | 1 | 1 | 1 | 2 | 1 | 2 | 1 | 3 |
| PGW | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Table 5.2: B-vEPC configurations for vertical scaling

Figure 5.6 illustrates the evaluation results of the B-vEPC performance for vertical scaling regarding throughput. We perform stress tests by using 200 concurrent users that allow generating 392 registrations per second during 120 seconds that generates a total of 47040 registrations. The vertical scaling evaluation results reveal that:

- Scaling both MME and SGW is better (approx. 63%) than scaling just MME or SGW.
- With two CPU cores in MME and SGW (Config V3), throughput is higher (approx. 80%) than baseline.
- With three CPU cores in MME and SGW (Config V4), the throughput is greater (approx. 89%) than the baseline.

- Increasing more than three CPU cores in MME and SGW does not lead to a better performance regarding throughput because the resources are over-provisioned without obtaining a significant increase in throughput.

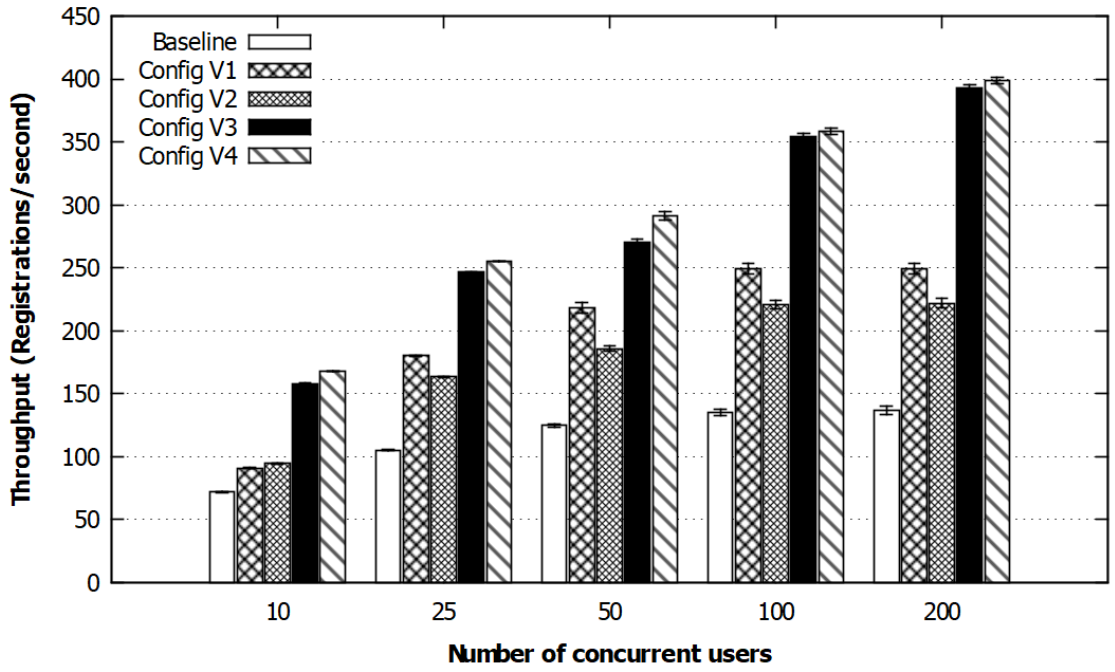



Figure 5.6: Throughput with vertical scaling

Figure 5.7 depicts the evaluation results of the B-vEPC performance for vertical scaling regarding latency. These latency results reveal that scaling both MME and SGW is lower (approx. 54%) than scaling just MME or SGW, and lesser (approx. 70%) than the baseline. Furthermore, with three CPU cores in MME and SGW (Config V4), the latency is lower (approx. 5%) than with two CPU cores in MME and SGW (Config V3).

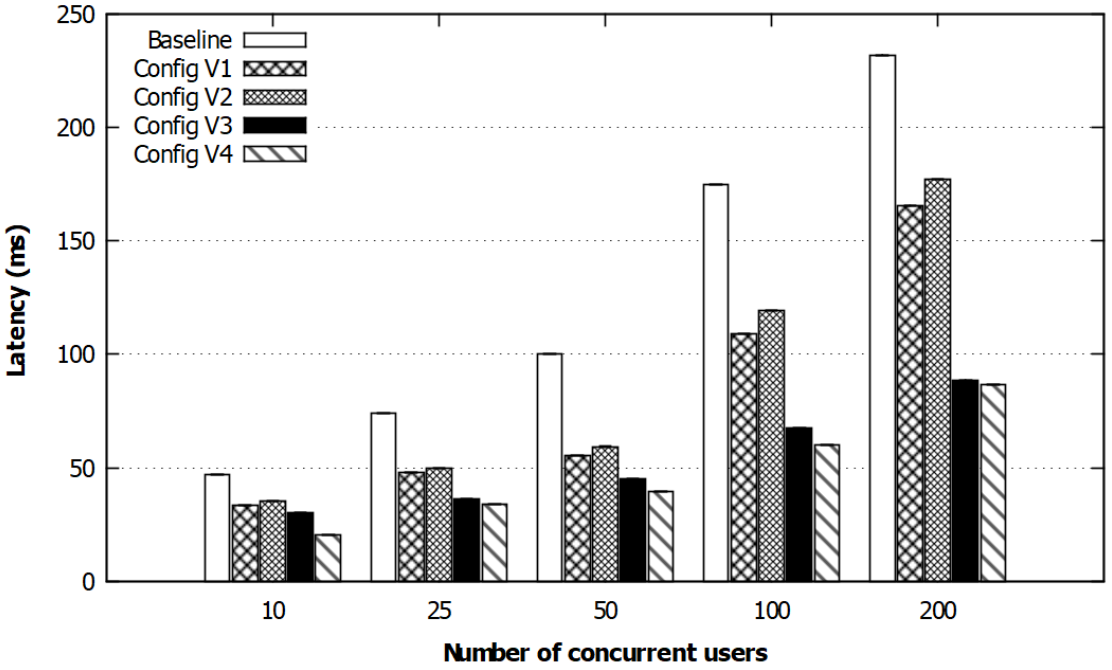



Figure 5.7: Latency with vertical scaling

Figure 5.8 presents the B-vEPC performance evaluation results regarding the use of CPU in MME. According to these results, the MME from the baseline quickly becomes saturated and reaches levels of 97% of CPU usage for 200 concurrent users. This saturation indicates that MME requires horizontal scaling to handle control processes. When MME is vertically scaled, the maximum use of CPU in MME is 65% for 200 concurrent users. SGW is also vertically scaled (Config V2) to test its relevance in the control plane processes. The evaluation results reveal that scaling only SGW leads to increase the use of CPU in MME because SGW generates an increasing workload to MME. When we performed a uniform vertical scaling of MME and SGW, the use of CPU in MME does not exceed the CPU saturation level.

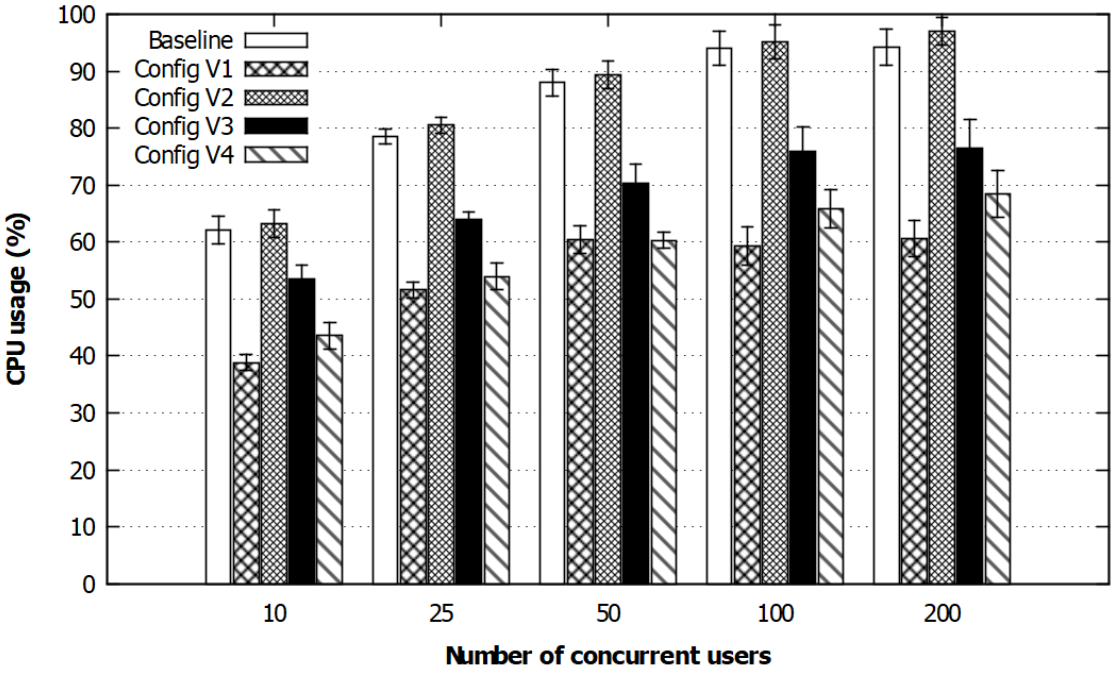



Figure 5.8: CPU usage in MME with vertical scaling

Figure 5.9 illustrates the B-vEPC performance evaluation results regarding the use of CPU in SGW. According to these results, the SGW from the baseline quickly becomes saturated. When MME is scaled (Config V1), it quickly leads to SGW to CPU saturation because MME sends too many control requests to SGW. Furthermore, when we performed a uniform vertical scaling of MME and SGW, the use of CPU in MME does not exceed the CPU saturation level. This CPU behavior corroborates the relevance of MME and SGW in control processes and, thus, the importance of scaling both MME and SGW.

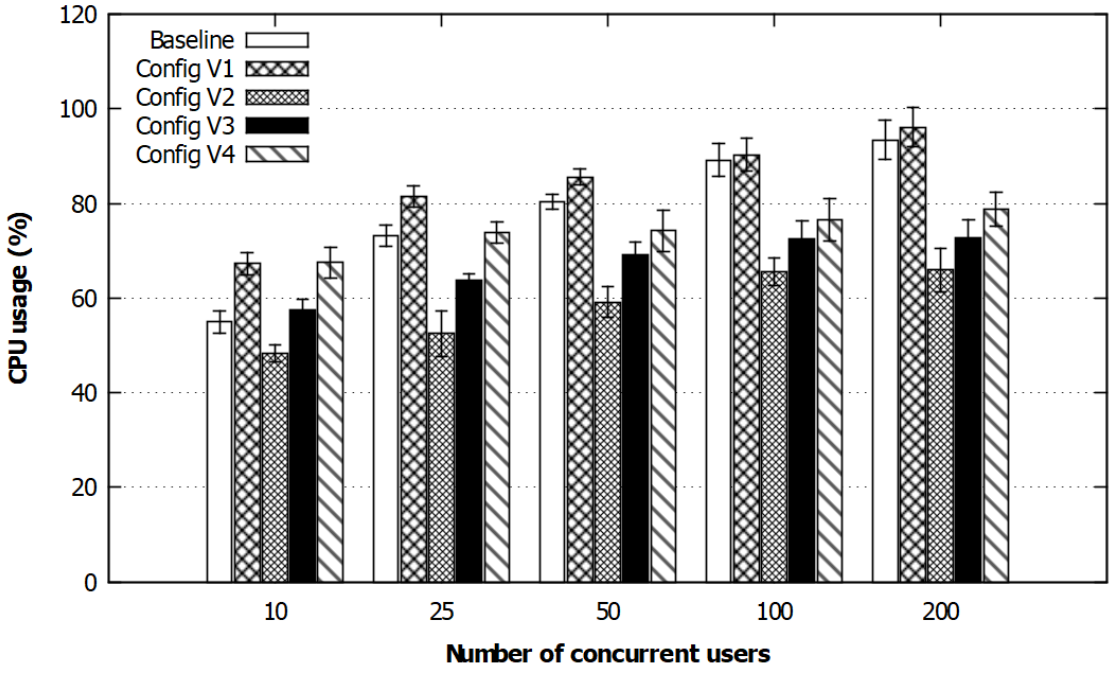



Figure 5.9: CPU usage in SGW with vertical scaling

Figure 5.10 depicts the B-vEPC performance evaluation results regarding the use of CPU in PGW. These results reveal that although PGW handles the control requests received from SGW, its use of CPU does not exceed the CPU saturation level in any of the vertical scaling configurations. This saturation behavior indicates that PGW does not require vertical scaling for handling the workload since PGW does not affect the B-vEPC performance.

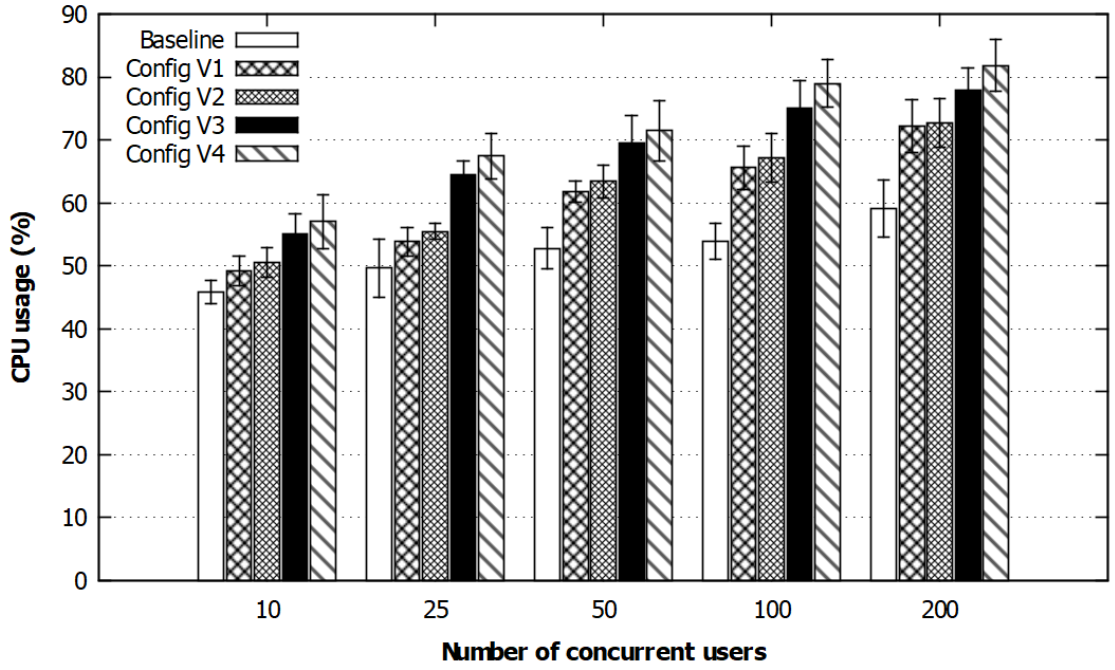



Figure 5.10: CPU usage in PGW with vertical scaling

To sum up, the B-vEPC performance evaluation for vertical scaling provides the following results.

- The number of processing cores is the most relevant resource to handle the control processes.
- PGW does not require vertical scaling for 200 concurrent or less because it does not exceed the CPU saturation level in any configuration.
- To obtain the highest performance value of B-vEPC with vertical scaling, it is necessary to scale both MME and SGW and assign resources uniformly.
- Scaling vertically with three CPU cores in MME and SGW (Config V4) does not represent a significant improvement in performance regarding throughput and latency compared with scaling with two CPU cores in MME and SGW (Config V3). Considering the above results, to improve the performance of B-vEPC another scaling method should be used.

## 5.4 Performance with Horizontal Scaling

Figure 5.11 presents the B-vEPC defined to perform horizontal scaling. This B-vEPC consists of a set of clusters per vEPC entity (*i.e.,* MME, SGW, and PGW). Each cluster is composed of a load balancer, workers, and a data store. The load balancer uses the round robin algorithm [49] to distribute the control traffic among the workers. The workers are exact replicas of the B-vEPC entities used in the baseline evaluation that are connected to the load balancer and the data store. The data store is in charge of storing the workers states, and its primary function is to guarantee the entity is fault tolerance. For instance, if a worker fails, the data store assigns the workload to another worker that is in operation.

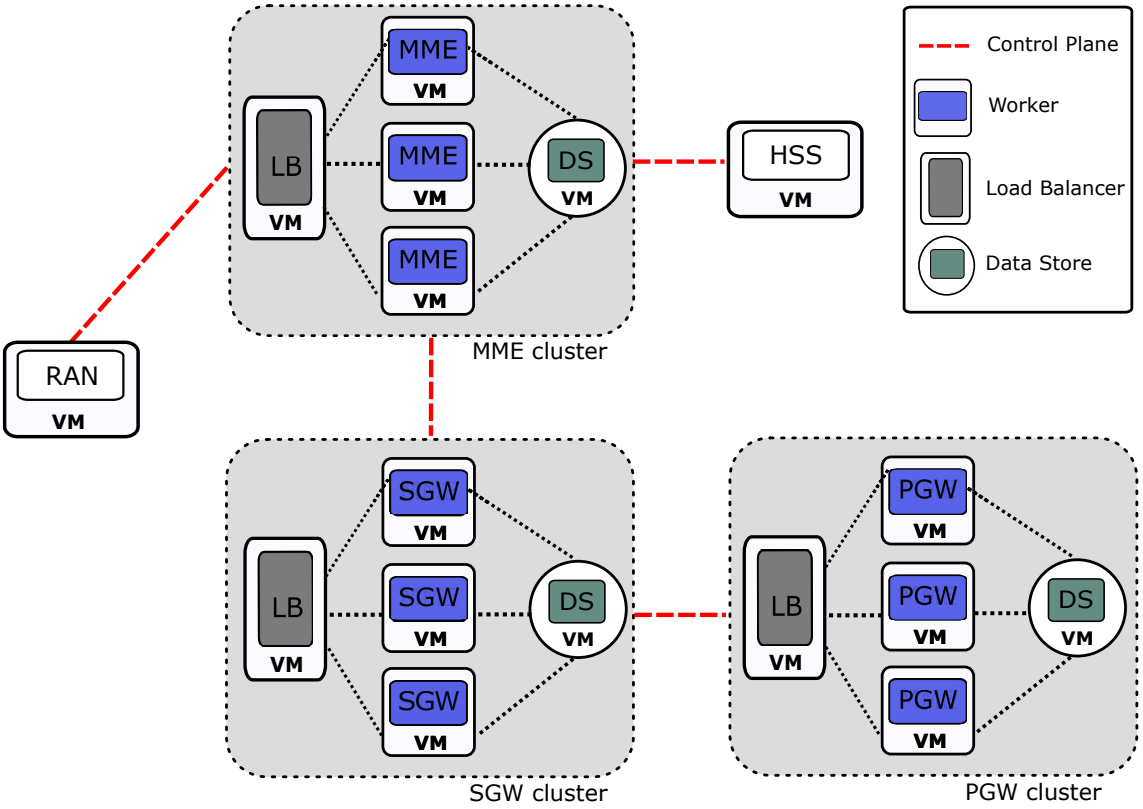



Figure 5.11: B-vEPC for horizontal scaling

In our horizontal scaling evaluation, three workers were deployed per cluster giving a total of 27 alternatives. All possible configurations were analyzed, however, for the sake of brevity only the most important results are discussed. Table 5.3 presents the most significant configurations performed per cluster. Hx denotes Configuration number 1, 2, 3, 4, 5 or 6 for horizontal scaling. Table 5.4 presents the resources assigned to B-vEPC for horizontal scaling [16].

| Cluster | Config H1 | Config H2 | Config H3 | Config H4 | Config H5 | Config H6 |
|---|---|---|---|---|---|---|
| | Workers | Workers | Workers | Workers | Workers | Workers |
| MME | 2 | 2 | 3 | 3 | 3 | 3 |
| SGW | 1 | 2 | 1 | 2 | 3 | 3 |
| PGW | 1 | 1 | 1 | 1 | 1 | 2 |

Table 5.3: B-vEPC configurations for horizontal scaling

| Entity | Hardware Resources | | |
|---|---|---|---|
| | RAM GB | CPU Cores | Storage GB |
| RAN | 4 | 4 | 10 |
| MME | 1 | 1 | 10 |
| HSS | 2 | 1 | 10 |
| SGW | 1 | 1 | 10 |
| PGW | 1 | 1 | 10 |
| LOAD BALANCER | 2 | 1 | 10 |
| DATA STORE | 2 | 2 | 15 |

Table 5.4: Resources assigned to B-vEPC for horizontal scaling

Figure 5.12 illustrates the evaluation results of the B-vEPC performance for horizontal scaling regarding throughput. We perform stress tests by using 200 concurrent users that allow generating 592 registrations per second during 120 seconds that generates a total of 71040 registrations. The horizontal scaling evaluation results reveal that:

- With two workers in MME cluster (Config H1), throughput is better (approx. 16%) than baseline.
- With three workers in MME and one worker in SGW clusters (Config H3), throughput is higher (approx. 36%) than baseline.
- By increasing the number of workers in equal proportion per cluster, throughput increases around 186% with two workers in MME and SGW clusters (Config H2) and 308% with three workers in MME and SGW (Config H5).
- Two workers in PGW cluster (Config H6) does not generate a significant improvement in performance regarding throughput.
- Increasing only the number of workers in the MME cluster becomes a bottleneck between MME and the SGW clusters because SGW cannot respond to the requests from the MME cluster. Thus, to obtain the highest throughput, it is necessary to increase the number of workers in MME and SGW uniformly.

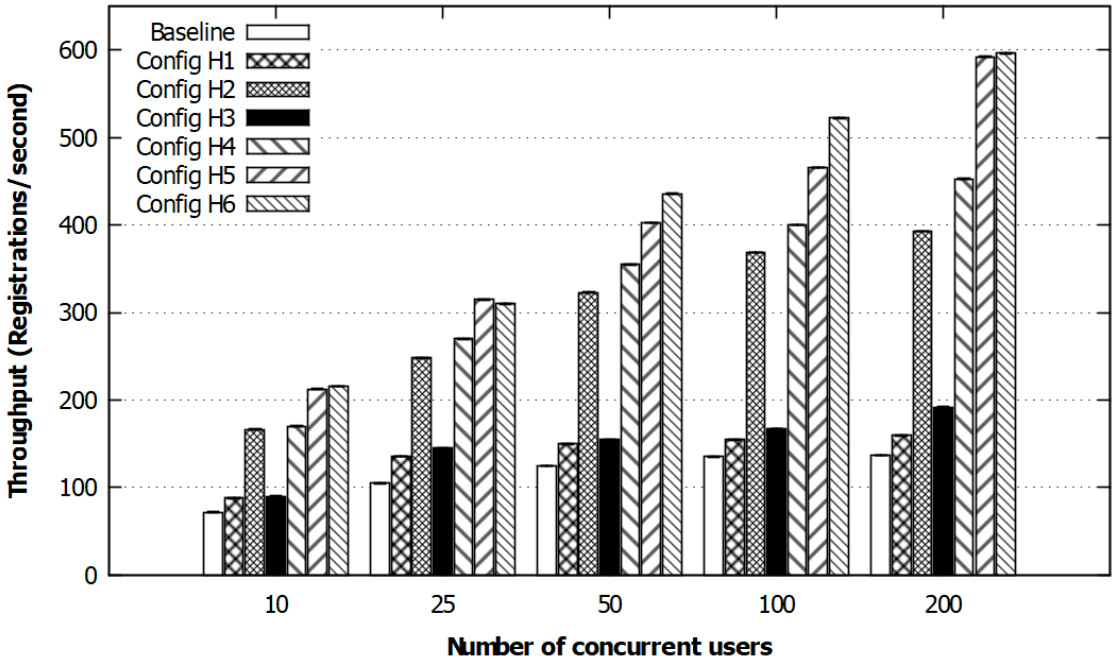



Figure 5.12: Throughput with horizontal scaling

Figure 5.13 depicts the evaluation results of the B-vEPC performance for horizontal scaling regarding latency. These results reveal that the latency increases as long as the number of workers per cluster increases. For instance, with three workers in MME and SGW clusters (Config H5), the latency is higher (approx. 8%) than the baseline. Although throughput increases because the workers distribute the workload to respond to more requests, B-vEPC with horizontal scaling adds latency. This increasing latency means that, for instance, the services with low latency requirement (*e.g.,* augmented reality, high-definition video streaming, and gaming) could face service performance degradations.

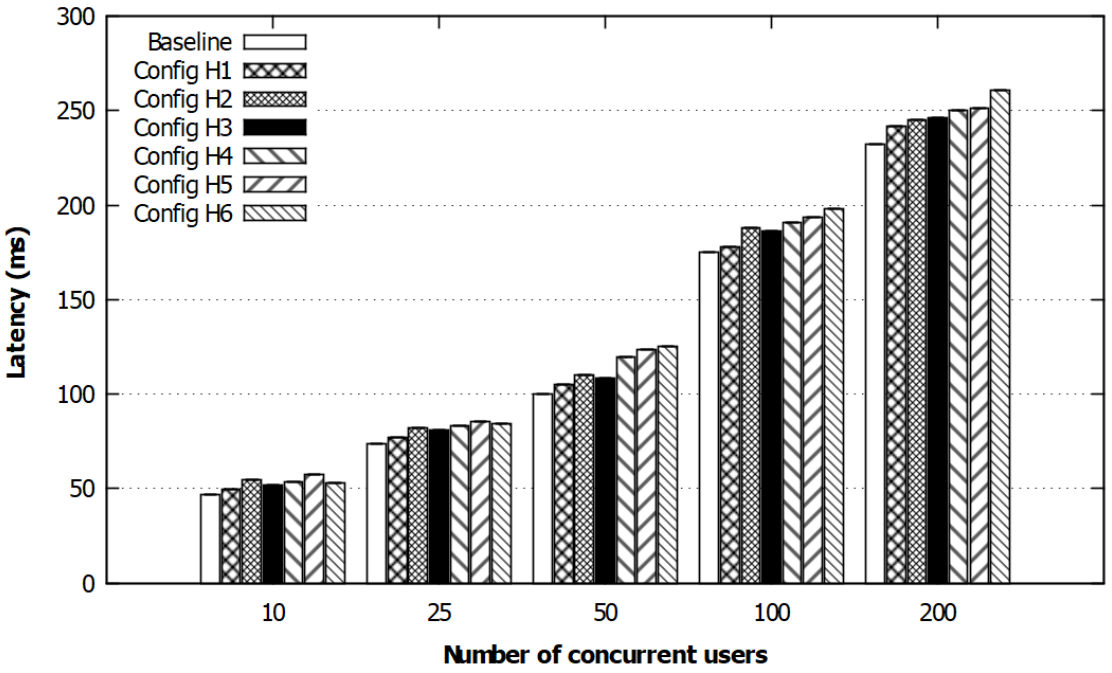



Figure 5.13: Latency with horizontal scaling

To sum up, the B-vEPC performance evaluation for horizontal scaling provides the following results.

- MME and SGW are the clusters that most affect the performance.
- PGW does not require horizontal scaling for 200 concurrent or less because increasing the number of workers in PGW does not achieve a significant

improvement in control plane performance.

- To obtain the highest throughput, it is necessary to increase the number of workers in MME and SGW uniformly.
- Although throughput increases because the workers distribute the workload to respond to more requests, B-vEPC with horizontal scaling adds latency.

## 5.5 Performance with Elastic Scaling

To evaluate our elastic scaling mechanism, we used three configurations, one from the baseline, one from the vertical scaling and another one from the horizontal scaling. We selected from vertical scaling Config V3, where MME and SGW have two processing cores, and PGW has one processing core. From the horizontal scaling, we selected Config H5, where MME and SGW clusters have three workers, and PGW is not scaled. We selected these configurations because they maintain a good balance between the resources consumed and the behavior of the performance metrics. The evaluation of the mechanism has as its starting point the B-vEPC for baseline.

To trigger our elastic scaling mechanism, we established regions that determine when to pass from one scaling method to another one. Figure 5.14 depicts the three regions framed by a number of concurrent users and their throughput for three different B-vEPC configurations. These regions define the behavior of our mechanism. For instance, if the B-vEPC workload is in the region I, the mechanism does not perform any action because B-vEPC for baseline can support the workload. If the B-vEPC workload increases and passes to Region II, the vertical scaling is activated. At last, if B-vEPC workload is in Region III, the horizontal scaling starts. Similarly, if the B-vEPC workload decreases and passes to Region II the mechanism return to vertical scaling to handle the workload. Moreover, if the workload is low and passes to Region I the mechanism uses the B-VEPC for baseline to support the workload.

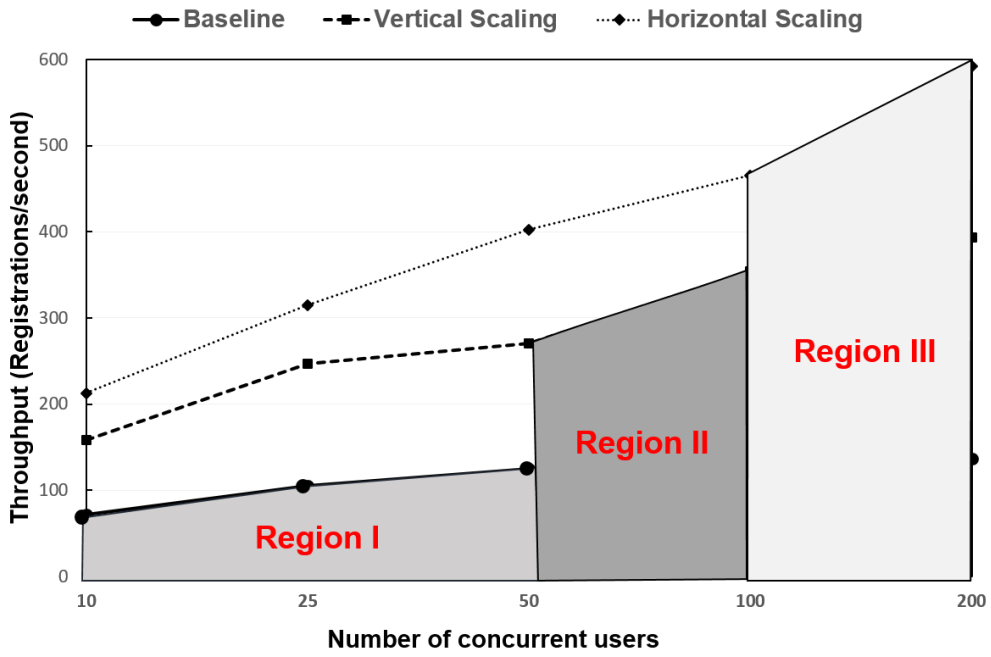



Figure 5.14: Working regions of the elastic scaling mechanism

| Scaling Method | Number of concurrent users | | |
|---|---|---|---|
| | 50 | 100 | 150 |
| Baseline | 90.47% | 98.54% | 99.82% |
| Vertical | 68.85% | 90.30% | 96.82% |
| Horizontal | 68.25% | 78.5% | 87.99% |

Table 5.5: B-vEPC throughput

To define the regions I, II and III, we took as a reference the throughput capacity from of the baseline, the vertical scaling (Config V3) and the horizontal scaling (Config H5). We calculated the percentage of the throughput capacity per configuration concerning its maximum throughput. Table 5.5 presents throughput capacities for 50, 100 and 150 concurrent users for the three B-vEPC configurations.

The B-vEPC for baseline supports the workload of 50 concurrent users (Region I) because for this number of concurrent users the percentage of throughput of

B-vEPC is more than 90% that indicates that from this point B-vEPC is becoming saturated and a fewer number of registrations per second are completed. The B-vEPC with vertical scaling supports the workload of 100 concurrent users (Region II) because for this number of concurrent users the percentage of throughput of B-vEPC is more than 90% that indicates that up to this point B-vEPC presents a good performance behavior regarding throughput.

The B-vEPC with horizontal scaling supports the workload of 150 concurrent users (Region III) because for this number of concurrent users the percentage of throughput of B-vEPC is less than 90% that indicates a good performance behavior in the face a high number of concurrent users. Thus, we determine that the workload of the concurrent users below 50 users defines the region I and can be supported by the B-vEPC for baseline. The workload of concurrent users that goes from 50 to 99 users defines the region II and can be supported by the B-vEPC with vertical scaling. The workload of concurrent users goes from 100 users defines the region III and can be supported by the B-vEPC with horizontal scaling. It is important to note that the mechanism allows the B-vEPC to move from any region to another depending on the workload variation.

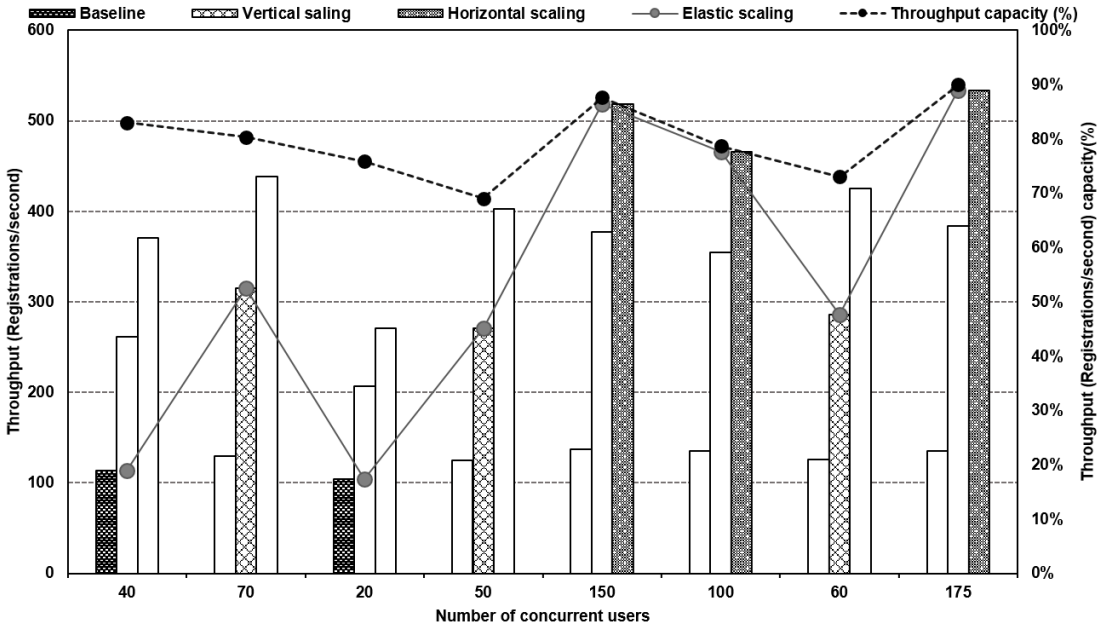



Figure 5.15: Elastic scaling mechanism

To evaluate our elastic scaling mechanism, we propose a daily scenario where the workload of concurrent users varies during the day. This scenario allows to evaluate the response of our mechanism to workload variations and illustrate the adaptive behavior of B-vEPC regarding such variations.

Figure 5.15 presents the daily scenario where a B-vEPC configuration supports each variation in the number of concurrent users. In the first point, 40 concurrent users are supported by B-vEPC with baseline configuration because the number of concurrent users is in Region I. At the next point, for 70 concurrent users the mechanism decides to scale vertically. Then when the number of concurrent users falls to 20, our mechanism decides that the B-vEPC for baseline must attend the workload again. In the next two points, for 50 concurrent users the mechanism scales the B-vEPC vertically and then, for 150 concurrent users, the mechanism scales the B-vEPC horizontally. Finally, in the last three points, from 100 to 60 and then to 175 concurrent users, the mechanism passes from horizontal scaling to vertical scaling the B-vEPC and vice versa. Furthermore, note that by using the elastic scaling, throughput capacity is always below 90% and above 65% that

indicates that it is possible to handle a workload variation and make resources saving.

# Chapter 6

# Conclusions and Future Work

In this chapter, we answer the proposed research question. Then, we provide the main conclusions obtained through the evaluation. Finally, we propose directions for future work.

## 6.1 Conclusions

This work presented the proposed solution to answer the research question: **What is the elastic scalability behavior of an LTE-EPC in an NFV environment?.** For answering this question, we introduced an elastic scaling mechanism in B-vEPC. The elastic scaling mechanism determines which scaling method must be carried out to support the workload variation in B-vEPC. This mechanism is formed by three modules (*i.e.,* Data Collection, Scaling Decision and Scaling Execution) and an algorithm that defines its operation. In particular, we deployed the B-vEPC and presented the evaluation and performance analysis of the B-vEPC including baseline, vertical, horizontal and elastic scaling. It is important to highlight that our proposal covers the performance evaluation of each individual vEPC entity when it supports elastic scaling capability and, this performance evaluation was performed regarding throughput, latency, CPU usage, and RAM usage.

This work represented an academic challenge because the inclusion of elastic scaling in B-vEPC involves first including the evaluation analysis of B-vEPC for baseline, and then applying vertical and horizontal scaling to B-vEPC. Also, the analysis of the elastic scaling per B-vEPC entity is innovative, since it allows to know what specific entity requires to be scaled to meet a workload. Moreover, we highlight the hard work because of deployment and configuration of B-vEPC and its high number of VMs to perform the evaluation of elastic scaling.

The results of our proposed solution revealed that:

- Our elastic scaling mechanism in B-vEPC provides the capability to adapt to variations in the number of concurrent users for the control traffic. Our mechanism determines whether an initial static configuration can handle the workload or whether it becomes necessary to increase the processing resources to the B-vEPC entities or whether it requires generating replicas of entities to distribute the workload.
- The most important control entities are MME and SGW, in both vertical and horizontal scaling. Scaling only MME becomes a bottleneck between MME and the SGW because SGW cannot respond to the requests from the MME. Thus, to obtain the highest throughput, it is necessary to scale both MME and SGW and assign resources uniformly.
- When we scale the B-vEPC vertically, the latency is 70% lower than the baseline, and throughput is almost 180% higher than the baseline.
- When we scale the B-vEPC horizontally, latency increases as well as the number of nodes that a control message needs to across. Furthermore, when we scale the B-vEPC horizontally, we can reach throughput 308% higher than the baseline.
- Our mechanism presents resources saving, and it scales according to the number of concurrent users, which allows it to scale to the right dimension to meet the workload variations. Moreover, it maintains the behavior of B-vEPC in a region without saturating its resources.

## 6.2 Future Work

According to work done for developing this undergraduate work, we expose some interesting ideas to continue it. These ideas are outlined below.

- Deploy the B-vEPC by using containers technology, to apply elastic scaling and evaluate the performance of B-vEPC under this virtualization technology to determine the advantages or disadvantages compared to VMs.
- Perform an evaluation of the B-vEPC based on Software Defined Networking (SDN) and to apply elastic scalability to find out how it can leverage the EPC control plane.
- Evaluate the B-vEPC data plane by using NFV and SDN and to incorporate scaling to perform a complete analysis of the B-vEPC scaling capacity.
- Perform the latency evaluation for horizontal scaling, by using vertical scaling of load balancers as an alternative to decrease latency.

# Bibliography


[1] S. S. Soliman and B. Song, “Fifth generation (5g) cellular and the network for tomorrow: cognitive and cooperative approach for energy savings,” *Network and Computer Applications*, vol. 85, pp. 84–93, May 2017.

[2] Cisco. (2017) Cisco visual networking index: Global mobile data traffic forecast update, 2016–2021.

[3] A. Gonzalez, P. Gronsund, and K. Mahmood, “Service availability in the nfv virtualized evolved packet core,” in *Global Communications Conference*. San Diego: IEEE, February 2015, pp. 1–6.

[4] X. Wang, C. Wu, F. Le, A. Liu, Z. Li, and F. Lau, “Online vnf scaling in datacenters,” in *International Conference on Cloud Computing*. San Francisco, CA, USA: IEEE, January 2016, pp. 140–147.

[5] H. Hawilo, A. Shami, M. Mirahmadi, and R. Asal, “Nfv: state of the art, challenges, and implementation in next generation mobile networks (vepc),” *IEEE Network*, vol. 28, pp. 18–26, December 2014.

[6] Y. Ren, T. Phung-Duc, J. C. Chen, and Z. W. Yu, “Dynamic auto scaling algorithm (dasa) for 5g mobile networks,” in *Global Communications Conference*. Washington, DC, USA: IEEE, December 2016, pp. 1–6.

[7] N. T. Jokin Garay, Jon Matias, “Toward an sdn-enabled nfv architecture,” *IEEE Communications Magazine*, vol. 53, pp. 187–193, April 2015.

[8] Y. Takano, A. Khan, and M. Tamura, “Virtualization-based scaling methods for stateful cellular network nodes using elastic core architecture,” in *International Conference on Cloud Computing Technology and Science*. Singapore, Singapore: IEEE, December 2014, pp. 204–209.

[9] S. William, *Foundation of modern networking SDN, NFV, QoE, IoT and Cloud*, 1st ed. New Jersey, USA: Addison-Wesley Professional, 2016, vol. I.

[10] R. Jain and S. Paul, “Network virtualization and software defined networking for cloud computing: a survey,” *IEEE Communications Magazine*, vol. 51, pp. 24–31, April 2013.

[11] G. Premsankar, K. Ahokas, and S. Luukkainen, “Design and implementation of a distributed mobility management entity on openstack,” in *International Conference on Cloud Computing Technology and Science*. Vancouver, BC, Canada: IEEE, November 2015, pp. 487–490.

[12] J. Prados-Garzon *et al.*, “Modeling and dimensioning of a virtualized mme for 5g mobile networks,” *IEEE Transactions on Vehicular Technology*, vol. 66, pp. 4383 – 4395, May 2017.

[13] C. Tobar, F. Risso, and O. Caicedo, “An adaptive scaling mechanism for managing performance variations in network functions virtualization: A case study in an nfv-based epc,” in *International Conference on Network and Service Management*. Tokyo, Japan: IEEE, January 2017, pp. 1–9.

[14] A. Banerjee, R. Mahindra, K. Sundaresan, S. Kasera, K. V. der Merwe, and S. Rangarajan, “Scaling the lte control-plane for future mobile access,” in *Conference on Emerging Networking Experiments and Technologies*, no. 19. Heidelberg, Germany: ACM, December 2015, pp. 1–13.

[15] P. Amogh, G. Veeramachaneni, A. K. Rangisetti, B. R. Tamma, and A. A. Franklin, “A cloud native solution for dynamic auto scaling of mme in lte,” in *International Symposium on Personal, Indoor, and Mobile Radio Communications*. Montreal, QC, Canada: IEEE, February 2018, pp. 1–7.

[16] P. Satapathy, J. Dave, P. Naik, and M. Vutukuru, "Performance comparison of state synchronization techniques in a distributed lte epc," in *Conference on Network Function Virtualization and Software Defined Networks*. Berlin, Germany: IEEE, November 2017, pp. 1–7.

[17] W. Hahn and B. Gajic, "Gw elasticity in data centers: Options to adapt to changing traffic profiles in control and user plane," in *International Conference on Intelligence in Next Generation Networks*. Paris, France: IEEE, April 2015, pp. 16–22.

[18] S. Dutta, T. Taleb, and A. Ksentini, "Qoe-aware elasticity support in cloud-native 5g systems," in *International Conference on Communications*. Kuala Lumpur, Malaysia: IEEE, July 2016, pp. 1–6.

[19] ETSI-GS-NFV-MAN. (2014) Network functions virtualization (nfv); management and orchestration. European Telecommunications Standards Institute. [Online]. Available: http://www.etsi.org

[20] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. D. Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 236–262, September 2016.

[21] ETSI-GS-NFV. (2013) Network functions virtualization (nfv); architectural framework. European Telecommunications Standards Institute. [Online]. Available: http://www.etsi.org

[22] J. G. Herrera and J. F. Botero, "Resource allocation in nfv: A comprehensive survey," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 518–532, August 2016.

[23] B. Han, V. Gopalakrishnan, and L. Ji, "Network function virtualization: Challenges and opportunities for innovations," *IEEE Communications Magazine*, vol. 53, pp. 90–97, February 2015.

[24] J. Ordonez, P. Ameigeiras, D. Lopez, J. J. Ramos, J. Lorca, and J. Folgueira, "Network slicing for 5g with sdn/nfv: Concepts, architectures and chal-

lenges," *Computer Science: Networking and Internet Architecture*, vol. 1, p. 19, 2017.

[25] E. Dahlman, S. Parkvall, and J. Skold, *4G: LTE/LTE_Advance for mobile Broadband*, 2nd ed. Waltham: Elsevier, 2014, vol. I.

[26] 3GPP. (2011) General packet radio service (gprs) enhancements for evolved universal terrestrial radio access network (e-utran) access. 3rd Generation Partnership Project. [Online]. Available: http://portal.3gpp.org

[27] Y. hwan Kim, H. kyo Lim, K. han Kim, and Y.-H. Han, "A sdn-based distributed mobility management in lte/epc network," *The Journal of Supercomputing*, vol. 73, p. 2919–2933, July 2017.

[28] S. B. Hadj-Said, M. R. Sama, and K. Guillouard, "New control plane in 3gpp lte/epc architecture for on-demand connectivity service," in *International Conference on Cloud Networking*. San Francisco, USA: IEEE, January 2014, pp. 205–209.

[29] T. Taleb, M. Corici, C. Parada, A. Jamakovic, S. Ruffino, G. Karagiannis, and T. Magedanz, "Ease: Epc as a service to ease mobile core network deployment over cloud," *IEEE Network*, vol. 29, no. 2, pp. 78–88, March 2015.

[30] G. Hasegawa and M. Murata, "Joint bearer aggregation and control-data plane separation in lte epc for increasing m2m communication capacity," in *Global Communications Conference*. San Diego, CA, USA: IEEE, December 2015, pp. 1–6.

[31] E. Dahlman, S. Parkvall, and J. Skold, *4G, LTE-Advanced Pro and The Road to 5G*, 3rd ed. Waltham: Elsevier, 2016, vol. I.

[32] F. Mehmeti and T. Spyropoulos, "Performance analysis of mobile data offloading in heterogeneous networks," *IEEE Transactions on Mobile Computing*, vol. 16, no. 2, pp. 482–497, April 2017.

[33] M. N. Sadiku and S. M. Musa, *Performance analysis of computer networks*, 1st ed. Springer, 2013, vol. I.

[34] M. Vutukuru, N. Sadagopan, P. Satapathy, and J. K. Dave, "A virtualized evolved packet core for lte networks," https://github.com/networkedsystemsIITB/NFV_LTE_EPC, 2016.

[35] R. Morabito, "Virtualization on internet of things edge devices with container technologies: A performance evaluation," *IEEE Access*, vol. 5, pp. 8835–8850, May 2017.

[36] L. Cao, P. Sharma, S. Fahmy, and V. Saxena, "Nfv-vital: A framework for characterizing the performance of virtual network functions," in *Conference on Network Function Virtualization and Software Defined Network (NFV-SDN)*. San Francisco, CA, USA: IEEE, November 2015, pp. 93–99.

[37] S. Nadgowda, S. Suneja, and A. Kanso, "Comparing scaling methods for linux containers," in *International Conference on Cloud Engineering*. Vancouver, BC, Canada: IEEE, April 2017, pp. 266–272.

[38] S. Becker, G. Brataas, and S. Lehrig, *Engineering Scalable, Elastic, and Cost-Efficient Cloud Computing Applications*, 1st ed. Cham, Switzerland: Springer International Publishing, 2017, vol. I.

[39] T. Choi, T. Kim, W. Tavernier, A. Korvala, and J. Pajunpaa, "Agile management and interoperability testing of sdn/nfv-enriched 5g core networks," *Communications and Experimental Trials with Heterogeneous and Agile Mobile networks*, vol. 40, p. 72–88, february 2018.

[40] T. Botran, J.-M. Alonso, and J. Lozano, "A review of auto-scaling techniques for elastic applications in cloud environments," *Grid Computing*, vol. 12, p. 559–592, December 2014.

[41] H. Arabnejad, C. Pahl, P. Jamshidi, and G. Estrada, "A comparison of reinforcement learning techniques for fuzzy cloud auto-scaling," in *International Symposium on Cluster, Cloud and Grid Computing*. Piscataway, NJ, USA: ACM, May 2017, pp. 64–73.

[42] S. Farokhi, P. Jamshidi, D. Lucanin, and I. Brandic, "Performance-based vertical memory elasticity," in *International Conference on Autonomic Computing*. Grenoble, France: IEEE, July 2015, pp. 151–157.

[43] A. Naskos, E. Stachtiari, A. Gounaris, P. Katsaros, D. Tsoumakos, I. Konstantinou, and S. Sioutas, "Dependable horizontal scaling based on probabilistic model checking," in *International Symposium on Cluster, Cloud and Grid Computing*, Shenzhen, China, May 2015, pp. 31–40.

[44] Y. Al-Dhuraibi, F. Paraiso, N. Djarallah, and P. Merle, "Elasticity in cloud computing: State of the art and research challenges," *IEEE Transactions on Services Computing*, vol. 11, no. 2, pp. 430–447, March 2018.

[45] S. Jangiti, V. S. S. Sriram, and R. Logesh, "The role of cloud computing infrastructure elasticity in energy efficient management of datacenters," in *International Conference on Power, Control, Signals and Instrumentation Engineering*. Chennai, India: IEEE, September 2017, pp. 758–763.

[46] N. Baldo, M. Miozzo, M. Requena-Esteso, and J. Nin-Guerrero, "An open source product-oriented lte network simulator based on ns-3," in *International conference on Modeling, analysis and simulation of wireless and mobile systems*. Miami, Florida, USA: ACM, November 2011, pp. 293–298.

[47] N. Nikaein and S. Krea, "Latency for real-time machine-to-machine communication in lte-based system architecture," in *European Wireless-Sustainable Wireless Technologies*. Vienna, Austria: IEEE, July 2011, pp. 1–6.

[48] W. Felter, A. Ferreira, R. Rajamony, and J. Rubio, "An updated performance comparison of virtual machines and linux containers," in *International Symposium on Performance Analysis of Systems and Software*. Philadelphia, PA, USA: IEEE, April 2015, pp. 171–172.

[49] M. U. Farooq, A. Shakoor, and A. B. Siddique, "An efficient dynamic round robin algorithm for cpu scheduling," in *International Conference on Communication, Computing and Digital Systems*. Islamabad, Pakistan: IEEE, March 2017, pp. 244–248.