QoS del Servicio de Video Llamada en una Red IMS Virtualizada

Monografía de Trabajo de Grado

María Camila Lara Paz
Heyman Andrés Coral Sarria

Director: Mg. Eduardo Rojas Pineda
Co-Director: Mg. Hebert Jair Gómez Fajardo

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Telemática
Línea de Investigación Servicios Avanzados de Telecomunicaciones
Popayán, Octubre de 2017

María Camila Lara Paz
Heyman Andres Coral Sarria

QoS del Servicio de Video Llamada en una Red IMS Virtualizada

Trabajo de grado presentado en la Facultad de Ingeniería Electrónica y Telecomunicaciones de la Universidad del Cauca para la obtención del Título de

Ingeniero en
Electrónica y Telecomunicaciones

Director:

Eduardo Rojas Pineda
Magister en Sistemas y Redes de Comunicaciones

Co-Director:

Hebert Jair Gómez Fajardo
Magister en Ingeniería en Telemática

Popayán
2017

AGRADECIMIENTOS

Como autores del presente trabajo de grado queremos brindar un agradecimiento muy especial a nuestro director Magister Eduardo Rojas Pineda por el tiempo dedicado, por su guía, sus consejos y su valiosa contribución para culminar con éxito el presente trabajo de grado. De la misma forma, expresar nuestros agradecimientos por el conocimiento compartido a nuestro Co-Director Magister Hebert Jair Gómez Fajardo y a todas las personas que de alguna u otra manera aportaron su conocimiento para el desarrollo de este trabajo de grado.

Quiero agradecer especialmente a mi familia por su amor incondicional, a mi padre Edgar Lara por sacrificarse y haberme formado como la persona que soy en la actualidad, porque en todo momento me ha apoyado y respetado mis decisiones. A mi madre Lupe Paz por sus consejos y respaldo, a mis hermanas Alejandra y Valentina Lara por ser mis amigas y compañeras de vida, por el apoyo incondicional y por creer en mí, muchos de mis logros se los debo a ustedes. A mi compañero de tesis por su amistad, por todas esas noches de desvelo y por su ayuda durante el desarrollo de este trabajo. También quiero agradecer a mis amigos por su colaboración y amistad a lo largo de toda la carrera y su constante interés en la evolución de este proyecto. ¡Muchas Gracias!

Maria Camila Lara Paz

Un especial agradecimiento a mi madre Lilia Sarria Alvares por el apoyo y sacrificios hechos durante toda mi carrera y mi vida, a mi padre José Eduardo Coral que desde muy lejos siempre me ha apoyado y gracias a sus consejos soy la persona que hoy en día soy, a mis hermanos David y Jonathan por su apoyo incondicional durante todo mi estudio, a Lorena Guerrero por brindarme su amor, alegría y comprensión a lo largo de estos años y a mi familia que es la razón de luchar cada día. Por último y no menos importante agradecer a mi compañera de tesis por su ayuda, paciencia y amistad durante el desarrollo de este proyecto.

Heyman Andrés Coral Sarria

RESUMEN ESTRUCTURADO

La mayoría de los sistemas de telecomunicaciones están migrando hacia redes convergentes buscando garantizar niveles adecuados de QoS para los diferentes tipos de servicios disponibles. Un estudio previo sobre QoS y redes IMS indican que no se cuenta con un modelo bien establecido que permita medir la QoS de un servicio de video llamada sobre una red IMS, debido a diferentes factores como la falta de estandarización, la introducción de nuevas tecnologías y la falta de estudios de este servicio sobre la arquitectura de IMS. Razón por la cual, el objetivo del presente trabajo de grado es proponer un modelo para evaluar la QoS del servicio de Video Llamada en una red IMS Virtualizada.

Para cumplir con dicho objetivo, se realiza un estudio de los parámetros de calidad que afectan la QoS de los servicios multimedia, seleccionando los más adecuados que permitan medir la calidad de una video llamada sobre una red IMS. También, se describe la infraestructura general de la red, especificando la topología, herramientas software y dispositivos hardware utilizados para las diferentes pruebas realizadas. Se realiza una caracterización de la red analizando la escalabilidad tanto vertical como horizontal para verificar la eficiencia y el rendimiento que esta tiene al momento de implementar un servicio. Además, se elabora un plan de pruebas con el fin de estructurar los diferentes tipos de escenarios y casos de estudio de una forma eficiente, entendible y que aborde los aspectos, parámetros y características que componen la red analizando los resultados obtenidos en cada uno de los casos de estudio planteados.

A partir de estos resultados se presentan las conclusiones, los aportes del proyecto y posibles trabajos futuros de investigación que pueden desprenderse del trabajo realizado, con el fin de dar continuidad a esta línea de investigación.

STRUCTURED ABSTRACT

Most telecommunication systems are migrating to converged networks looking for the appropriate levels of QoS for the different types of services available. A previous study on QoS and IMS networks indicate that there is no well-established model to measure the QoS of a video call service over a IMS network, due to different factors such as the lack of standardization, the introduction of new technologies and the lack of studies of this service on IMS architecture. For this reason, the objective of the present work of degree is to propose a model to evaluate the QoS of the Service of Video Call in a network IMS Virtualized.

To meet this objective, a study of the quality parameters that affect the QoS of multimedia services is carried out, selecting the most suitable to measure the quality of a video call over an IMS network. Also, the general infrastructure of the network is described, specifying the topology, software tools and hardware devices used for the different tests performed. A network characterization is performed analyzing the vertical and horizontal scalability to verify the efficiency and the performance that it has in the moment of implementing a service. In addition, a test plan is developed in order to structure the different types of scenarios and case studies in an efficient, understandable way and using the parameters, and the characteristics that make up the network analyzing the results obtained in each of the case studies.

From these results we present the conclusions, the contributions of the project and possible future research work that can be derived from the work done, in order to give continuity to this line of research.

TABLA DE CONTENIDO

AGRADECIMIENTOS i
RESUMEN ESTRUCTURADO ii
STRUCTURED ABSTRACT iii
LISTA DE FIGURAS ix
LISTA DE TABLAS xi
1. INTRODUCCIÓN 1
1.1. CONTEXTO DEL PROYECTO 1
1.2. OBJETIVOS 2
1.2.1. Objetivo General 2
1.2.2. Objetivos Específicos 2
1.3. ESTRUCTURA DEL DOCUMENTO 2
2. ESTADO DEL ARTE 5
2.1. VIRTUALIZACIÓN 5
2.1.1. Conceptos Básicos de Virtualización 6
2.1.1.1. Máquina Virtual (VM) 6
2.1.1.2. Hipervisor 6
2.1.2. Tipos de Virtualización 8
2.1.2.1. Virtualización de Servidores. 8
2.1.2.2. Virtualización de Redes 8
2.1.2.3. Virtualización de Escritorios 9
2.1.2.4. Almacenamiento definido por software (SDS) 9
2.1.3. Ventajas de la Virtualización 9
2.2. REDES NGN 9
2.2.1. Arquitectura 10
2.2.1.1. Capa de Acceso 11
2.2.1.2. Transporte 11
2.2.1.3. Capa de Control 11
2.2.1.4. Capa de Servicio 12
2.2.2. Características 12
2.3. IMS 13
2.3.1. Arquitectura IMS 13
2.3.1.1. CSCF 14
2.3.1.2. HSS 15

2.3.1.3. AS 15
2.3.2. Protocolos en IMS 15
2.3.2.1. SIP 16
2.3.2.1.1. Entidades SIP 17
2.3.2.1.2. Mensaje SIP 17
2.3.2.2. SDP 18
2.3.2.3. DIAMETER 19
2.3.2.4. RTP 19
2.3.2.5. RTCP 20
2.4. QoS 20
2.4.1. Modelos de QoS 20
2.5. Video Llamada 21
3. MODELO DE CALIDAD DE SERVICIO (QoS) 23
3.1. PARAMETROS DE CALIDAD 23
3.1.1. Factores de Calidad 23
3.1.1.1. Ancho de banda 23
3.1.1.2. Retardo “extremo a extremo” o latencia 24
3.1.1.3. Variabilidad del retardo o “jitter” 24
3.1.1.4. Pérdida de paquetes 25
3.1.1.5. Velocidad de bits de vídeo 25
3.1.1.6. Pérdida de fotogramas de video 25
3.1.1.7. Tiempo de retardo Round-Trip (RTT) 25
3.1.1.8. Tasa de fotogramas 26
3.1.1.9. Densidad de ráfaga 26
3.1.1.10. Retardo de transferencia de paquetes IP (IPTD) 26
3.1.1.11. Variaciones de retardo de paquetes IP (IPDV) 26
3.1.1.12. Relación de error de paquetes IP (IPER) 27
3.1.1.13. Relación de perdida de paquetes IP (IPLR) 27
3.1.1.14. Tasa de paquetes IP espurios (SPR) 27
3.1.1.15. Relación de paquetes IP reordenados (IPRR) 27
3.1.1.16. Relación de bloques de pérdida severa de paquetes IP (IPSLBR) 27
3.1.1.17. Relación de paquetes IP duplicados (IPDR) 28
3.1.1.18. Relación de paquetes IP replicados (RIPR) 28
3.1.1.19. Evento de pérdida de paquetes consecutivos 28

3.1.1.20. Segundo degradado 28
3.1.1.21. Cuantificación de fluctuación/variación del retardo IP a corto plazo ($IPDVShort\_term$) 28
3.1.1.22. Ventana de descarte 29
3.1.2. Clasificación de los parámetros 29
3.1.2.1. Características físicas de la red 29
3.1.2.2. Retardo 29
3.1.2.3. Variación del retardo 30
3.1.2.4. Pérdida de información 30
3.1.3. Parámetros Seleccionados 30
3.1.3.1. Características físicas de la red 31
3.1.3.1.1. Ancho de banda 31
3.1.3.2. Retardo 33
3.1.3.2.1. Retardo de “Extremo a Extremo” (latencia) 34
3.1.3.3. Variación del retardo 34
3.1.3.3.1. “Jitter” 34
3.1.3.4. Pérdida de información 35
3.1.3.4.1. Pérdida de paquetes 36
3.2. INFRAESTRUCTURA GENERAL DE LA RED 36
3.2.1. Hardware 36
3.2.1.1. Switch HP 5500 36
3.2.1.2. Virtual Connect 37
3.2.1.3. Blade BL460 Gen 9 37
3.2.1.4. MSA 2040 38
3.2.1.5. Host 39
3.2.2. Software 39
3.2.2.1. VMWare ESXi 39
3.2.2.2. Ubuntu 14.04 LTS 40
3.2.2.3. Open IMS Core 40
3.2.2.4. Fokus Monster Client 41
3.2.2.5. Wireshark 41
3.2.2.6. Wondershaper 41
3.2.3. Topología general de la red 42
3.3. CARACTERIZACIÓN DE LA RED 44
3.3.1. Escalabilidad Vertical 44

3.3.2. Escalabilidad Horizontal 46
3.4. CARACTERIZACIÓN DEL SERVICIO DE VIDEO LLAMADA 47
3.4.1. Propiedades de una video llamada 48
3.4.1.1. Propiedades de audio 48
3.4.1.2. Propiedades de video 48
3.4.2. Secuencia de mensajes 49
4. PLAN DE PRUEBAS Y ANALISIS DE RESULTADOS 55
4.1. ESTRUCTURA DE LAS PRUEBAS 55
4.1.1. Escalabilidad Vertical 56
4.1.2. Escalabilidad Horizontal 56
4.1.3. Parámetros de Calidad 56
4.2. DESARROLLO DE LAS PRUEBAS 56
4.2.1. Escalabilidad Vertical 56
4.2.1.1. Componentes de pruebas 57
4.2.1.1.1. Tiempo de concurrencia entre peticiones consecutivas 58
4.2.1.2. Plan de pruebas 58
4.2.1.3. Resultados de las pruebas 59
4.2.1.3.1. Caso de estudio 1 59
4.2.1.3.2. Caso de estudio 2 61
4.2.1.3.3. Caso de estudio 3 63
4.2.1.3.4. Caso de estudio 4 64
4.2.1.4. Análisis de las pruebas 66
4.2.2. Escalabilidad Horizontal 67
4.2.2.1. Componentes de prueba 68
4.2.2.1.1. Balanceador de carga 68
4.2.2.2. Plan de pruebas 69
4.2.2.3. Resultados de las pruebas 69
4.2.2.3.1. Caso de estudio 1 69
4.2.2.3.2. Caso de estudio 2 71
4.2.2.4. Análisis de las pruebas 72
4.2.3. Parámetros de Calidad 74
4.2.3.1. Componentes de prueba 75
4.2.3.2. Plan de pruebas 75
4.2.3.3. Resultados de las pruebas 75
4.2.3.3.1. Retardo 77
4.2.3.3.2. Pérdida de paquetes 78

4.2.3.3.3. “Jitter” 80
4.2.3.4. Análisis de las pruebas 82
5. CONCLUSIONES Y TRABAJOS FUTUROS 85
5.1. CONCLUSIONES 85
5.2. APORTES ADICIONALES 86
5.3. TRABAJOS FUTUROS 87
REFERENCIAS BIBLIOGRAFICAS 89
ANEXO A. INSTALACION Y CONFIGURACION DEL SOFTWARE 95
A.1. OPEN IMS CORE 95
A.2. INSTALACIÓN MONSTER FOKUS 102
A.3. INSTALACION WONDERSHAPER 103
ANEXO B. COMPONENTES DE PRUEBA 105
B.1. TIEMPO DE CONCURRENCIA ENTRE PETICIONES CONSECUTIVAS SEGÚN LA DISTRIBUCION POISSON 105
B.2. CREACION DE USUARIOS EN OPEN IMS CORE 106
B.3. SIMULADOR DE MENSAJES SIP 108
B.3.1. Software y lenguajes de programación 108
B.3.2. Desarrollo del simulador de mensajes SIP 110
B.4. BALANCEADOR DE CARGA 117

LISTA DE FIGURAS

Figura 1. Infraestructura Virtual. Fuente: [8] 5
Figura 2. Hipervisor Tipo 1. Fuente: [10] 7
Figura 3. Hipervisor Tipo 2. Fuente: [10] 7
Figura 4. Hipervisor Monolítico. Fuente: [10] 7
Figura 5. Hipervisor con Microkernel. Fuente: [10] 8
Figura 6. Evolución de la red clásica a la red NGN. Fuente: [13]. 10
Figura 7. Arquitectura Red NGN. Fuente: [13]. 11
Figura 8. Arquitectura IMS. Fuente: [17]. 13
Figura 9. Protocolos usados en IMS. Fuente: [19]. 16
Figura 10. Entidades SIP. Fuente: [20]. 17
Figura 11. Tiempo de transmisión en un sentido. Fuente: [26] 24
Figura 12. Switch HP 5500. Fuente: [38] 36
Figura 13. HP Virtual Connect. Fuente: [39] 37
Figura 14. Servidor Blade BL460c Gen 9. Fuente: [40] 38
Figura 15. MSA 2040. Fuente: [41] 38
Figura 16. Sistema básico de gestión. Fuente: [42] 39
Figura 17. Topología general de la red. Fuente: Propia 42
Figura 18. Diagrama jerárquico de implementación del Core IMS. Fuente: Propia 43
Figura 19. Diagrama jerárquico de implementación del cliente IMS. Fuente: Propia 43
Figura 20. Escalabilidad Vertical. Fuente: Propia 45
Figura 21. Topología Escalamiento Vertical. Fuente: Propia 45
Figura 22. Escalabilidad Horizontal. Fuente: Propia 46
Figura 23. Topología Escalamiento Horizontal. Fuente: Propia 47
Figura 24. Diagrama de secuencia del mensaje Register. Fuente: Propia 50
Figura 25. Diagrama de secuencia del mensaje Subscribe. Fuente: Propia 50
Figura 26. Diagrama de secuencia del mensaje Invite. Fuente: Propia 51
Figura 27. Diagrama de secuencia de mensajes RTP. Fuente: Propia 52
Figura 28. Diagrama de secuencia del mensaje Bye. Fuente: Propia 53
Figura 29. Diagrama de secuencia para anular el registro en el Core IMS. Fuente: Propia 54
Figura 30. Tiempo de establecimiento del Core IMS 1. Fuente: Propia 60
Figura 31. Tiempo de desconexión del Core IMS 1. Fuente: Propia 61
Figura 32. Perdida de paquetes del Core IMS 1. Fuente: Propia 61
Figura 33. Tiempo de establecimiento del Core IMS 2. Fuente: Propia 62
Figura 34. Tiempo de desconexión del Core IMS 2. Fuente: Propia 62
Figura 35. Perdida de paquetes del Core IMS 2. Fuente: Propia 63
Figura 36. Tiempo de establecimiento del Core IMS 3. Fuente: Propia 63
Figura 37. Tiempo de desconexión del Core IMS 3. Fuente: Propia 64
Figura 38. Perdida de paquetes del Core IMS 3. Fuente: Propia 64
Figura 39. Tiempo de establecimiento del Core IMS 4. Fuente: Propia 65
Figura 40. Tiempo de desconexión del Core IMS 4. Fuente: Propia 65
Figura 41. Perdida de paquetes del Core IMS 4. Fuente: Propia 66

Figura 42. Tiempos de respuesta de cada Core IMS. Fuente: Propia..... 67
Figura 43. Perdida de paquetes de cada Core IMS. Fuente: Propia 67
Figura 44. Tiempo de establecimiento de 2 Core IMS. Fuente: Propia 70
Figura 45. Tiempo de desconexión de 2 Core IMS. Fuente: Propia..... 70
Figura 46. Tiempo de establecimiento de 3 Core IMS. Fuente: Propia 71
Figura 47. Tiempo de desconexión de 3 Core IMS. Fuente: Propia..... 72
Figura 48. Tiempos de respuesta según el número de Core IMS. Fuente: Propia 72
Figura 49. Porcentaje de mejora según el número de Core IMS. Fuente: Propia 73
Figura 50. Perdida de paquetes según el número de Core IMS. Fuente: Propia 74
Figura 51. Tiempo de Conexión. Fuente: Propia..... 76
Figura 52. Tiempo de desconexión. Fuente: Propia..... 76
Figura 53. Retardo de transmisión. Fuente: Propia..... 78
Figura 54. Perdida de paquetes de audio. Fuente: Propia 79
Figura 55. Perdida de paquetes de video. Fuente: Propia 79
Figura 56. "Jitter" de audio. Fuente: Propia..... 80
Figura 57. "Jitter" de video. Fuente: Propia..... 81
Figura 58. Jitter de video a partir de 512 Kbps. Fuente: Propia 81
Figura 59. Calidad de video para un ancho de banda de 64 y 256 Kbps. Fuente: Propia 82
Figura 60. Calidad de video para un ancho de banda de 512 y 640 Kbps. Fuente: Propia..... 83
Figura 61. Interfaz web FHoSS. Fuente: Propia..... 102
Figura 62. Configuración del cliente Fokus Monster. Fuente: Propia..... 103
Figura 63. Script para la creación masiva de usuarios. Fuente: Propia 106
Figura 64. Modificación del script de Open IMS Core. Fuente: Propia..... 107
Figura 65. Comando para ejecutar el script de creación de usuarios. Fuente: Propia 107
Figura 66. Lista de usuarios registrados en Open IMS Core. Fuente: Propia 107
Figura 67. Modelo conceptual de la base de datos. Fuente: Propia..... 111
Figura 68. Modelo lógico de la base de datos. Fuente: Propia 112
Figura 69. Modelo físico de la base de datos. Fuente: Propia 113
Figura 70. Comandos para el envío de paquetes. Fuente: Propia 114
Figura 71. Diagrama de flujo del envío de peticiones. Fuente: Propia 114
Figura 72. Comandos para escuchar, procesar y enviar mensajes SIP. Fuente: Propia 115
Figura 73. Diagrama de flujo para escuchar, procesar y enviar mensajes SIP. Fuente: Propia..... 116
Figura 74. Comando para el cálculo de tiempos de respuesta y pérdida de paquetes. Fuente: Propia 117
Figura 75. Diagrama de flujo del balanceador de carga. Fuente: Propia 118

LISTA DE TABLAS

Tabla 1. Clases de peticiones SIP..... 18
Tabla 2. Clases de respuestas SIP. 18
Tabla 3. Ancho de banda según el códec de video. Fuente: [35]..... 32
Tabla 4. Ancho de banda según el códec de audio. Fuente: [36]..... 33
Tabla 5. Características de los hosts 39
Tabla 6. Características físicas de las MV 57
Tabla 7. Anchos de banda según los parámetros de calidad..... 82

1. INTRODUCCIÓN

1.1. CONTEXTO DEL PROYECTO

La demanda de servicios de telecomunicaciones crece de manera acelerada y requiere que las redes que prestan estos servicios sean convergentes y gestionables [1]. La tendencia predominante (en los Telco) es implementar una infraestructura de red NGN virtualizada, basada en IMS, que se integre con las redes ya existentes y que facilite la evolución y adaptación de los servicios para poder atender la diversidad de requerimientos de los usuarios [2]. Las redes virtualizadas pueden facilitar la mejora en rendimiento, portabilidad, flexibilidad y, principalmente, la reducción de costos en todos los aspectos, como por ejemplo la reducción en la energía consumida por los servidores, la disminución en tiempo y esfuerzos para el mantenimiento, administración, soporte y recuperación de los servicios, incrementando la calidad ofrecida de los mismos [3] [4].

La mayoría de los sistemas de telecomunicaciones están migrando hacia redes virtualizadas con el fin de aprovechar todas las ventajas que ofrecen, además de facilitar la convergencia con diferentes servicios de telecomunicaciones tales como sesiones multimedia y VoIP, entre otros. Estas nuevas redes deben garantizar niveles de calidad del servicio (QoS, por sus siglas en inglés Quality Of Service) adecuados para los diferentes tipos de servicios (servicios convergentes) que transporta [5].

Uno de los servicios que será propósito de investigación en la plataforma TELCO 2.0 de la Universidad del Cauca es el de video llamada en ambiente IMS, ya que integra un servicio de telecomunicaciones típico (transmisión de voz) con un servicio multimedia web (video, texto, documentos, etc.), buscando garantizar y mantener la calidad de servicio (QoS) sobre la red virtualizada. La calidad de servicio, según la UIT, se define como: “La totalidad de las características de un servicio de telecomunicaciones que determinan su capacidad para satisfacer las necesidades explícitas e implícitas del usuario del servicio” [6]. Por lo tanto, se hace necesario contar con la metodología adecuada que permita medir la QoS de los servicios sobre el ambiente de red virtualizada, para establecer de qué manera se afectan las distintas características del servicio que determinan su capacidad para satisfacer las necesidades de sus usuarios.

Un primer acercamiento al tema indica que no se cuenta con un modelo bien establecido que permita medir la QoS de un servicio de video llamada sobre una red virtualizada, debido a diferentes factores como la falta de estandarización, la introducción de nuevas tecnologías y la falta de estudios de este servicio sobre la arquitectura de IMS. En consecuencia, el objetivo de este trabajo de grado es realizar un estudio de los diferentes parámetros que

permiten medir la QoS del servicio de video llamada sobre una red IMS y proponer un modelo apropiado para el análisis de la calidad del servicio.

Considerando lo expuesto, surge la siguiente pregunta de investigación: ¿Es posible estructurar un modelo que permita medir la calidad del servicio (QoS) de video llamada implementado sobre una red virtualizada IMS?

1.2. OBJETIVOS

1.2.1. Objetivo General

Proponer un modelo para evaluar la calidad de servicio (QoS) de un servicio de video llamada implementado sobre la red virtualizada IMS.

1.2.2. Objetivos Específicos

- Evaluar y seleccionar los parámetros más apropiados para medir la QoS del servicio de video llamada.
- Proponer un modelo para el análisis de la QoS de un servicio de video llamada sobre una red virtualizada IMS, utilizando los parámetros seleccionados.
- Validar el modelo de QoS de video llamada propuesto sobre la red virtualizada IMS.

1.3. ESTRUCTURA DEL DOCUMENTO

Este documento se encuentra estructurado en cinco capítulos. A continuación se da una breve explicación del contenido de cada uno:

- **Capítulo 1:** Introducción. Se da una descripción del contexto del proyecto, se exponen los objetivos tanto generales como específicos y se detalla la estructura en general del documento.

- **Capítulo 2:** Estado del Arte. Este capítulo presenta una breve definición de los principales términos relacionados con el documento. En este caso, los conceptos a tratar son Virtualización, NGN, IMS y QoS.

- **Capítulo 3:** Modelo de calidad. En este capítulo se analizan, clasifican y seleccionan los parámetros más adecuados para medir la QoS de un servicio de video llamada. Además, se describe la infraestructura general, definiendo las características del hardware, software y topología de la red a utilizar. Por último, se realiza una caracterización de la red, analizando la escalabilidad tanto vertical como horizontal para verificar la eficiencia y el rendimiento que esta tiene al momento de implementar un servicio.

- **Capítulo 4:** Plan de pruebas y análisis de resultados. En este capítulo se describe la metodología de pruebas para cada caso de estudio. En primer lugar se estudia la escalabilidad vertical, especificando las características físicas de las máquinas virtuales a utilizar, los componentes necesarios para el desarrollo de dichas pruebas y el análisis de los resultados obtenidos. De la misma manera se examina la escalabilidad horizontal de la red haciendo uso de la técnica de balanceo de carga y por último, se analizan los parámetros de calidad como retardo, perdida de paquetes y “jitter”, observando su comportamiento al variar el ancho de banda.

- **Capítulo 5:** Conclusiones y trabajos futuros. En este capítulo se presentan las conclusiones y posibles trabajos futuros de investigación que pueden desprenderse del trabajo de grado realizado, con el fin de dar continuidad a esta línea de investigación.

2. ESTADO DEL ARTE

Para contextualizar la investigación se muestra una descripción detallada de los principales términos relacionados con el presente trabajo de grado. Se define el concepto de virtualización, así como las principales ventajas que trae consigo su implementación en las redes. También se aborda el concepto de NGN, la descripción de su arquitectura y sus principales características. De la misma manera se analiza la definición de IMS, la descripción por capas de su arquitectura con sus principales componentes y los protocolos de comunicación utilizados por IMS. Finalmente, se especifica el concepto y los modelos de QoS.

2.1. VIRTUALIZACIÓN

La virtualización consiste en crear una versión virtual de elementos dentro de una red y recursos computacionales, permitiendo que se puedan dividir o agregar según sea la necesidad del servicio que se despliega sobre la red. La virtualización permite que un único recurso físico como un servidor, un switch, un dispositivo de almacenamiento o un sistema operativo sea utilizado como múltiples recursos virtuales, en sentido inverso, agregar múltiples recursos físicos, tales como varios servidores o dispositivos de almacenamiento, para que sean manejados como un único recurso virtual [7].

En un entorno virtual, se pueden ejecutar diferentes sistemas operativos y aplicaciones en un único servidor físico o “host”. Cada sistema operativo se encuentra en un contenedor software completamente independiente, denominado máquina virtual (VM). Estas VM comparten los mismos recursos disponibles del servidor físico como la memoria, CPU, almacenamiento, entre otras [8].

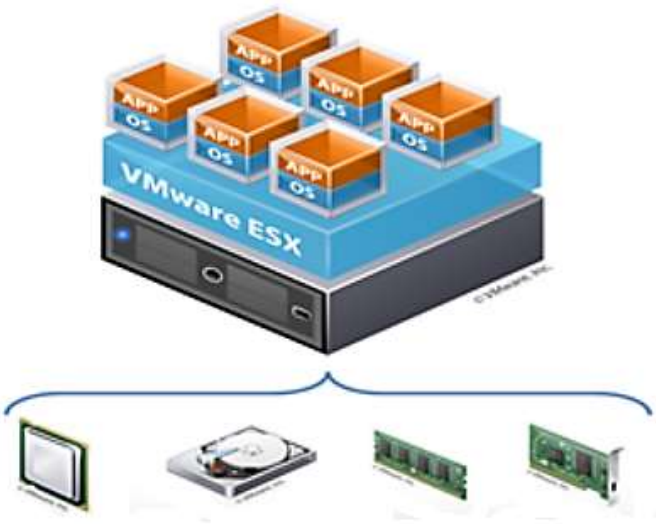

Figura 1. Infraestructura Virtual. Fuente: [8]

2.1.1. Conceptos Básicos de Virtualización

2.1.1.1. Máquina Virtual (VM)

Una máquina virtual es un software que emula un computador ejecutando programas como si fuera uno real. Cada máquina virtual puede tener su propio sistema operativo y aplicaciones utilizando los recursos de hardware del servidor físico como disco duro, memoria RAM, CPU, red, entre otros. Estos recursos son gestionados de manera dinámica por un hipervisor o Virtual Machine Monitor (VMM) [9].

Con el fin de optimizar la utilización de los recursos del hardware, las VMs se pueden eliminar, mover, copiar y reasignar con facilidad entre servidores anfitriones, proporcionando ciertas ventajas descritas a continuación:

- Aislamiento entre máquinas virtuales.
- Independencia del hardware entre máquinas virtuales.
- Ejecuta varios sistemas operativos en un solo host físico.
- Fragmenta los recursos hardware del sistema entre las máquinas virtuales.
- Migra cualquier máquina virtual a un servidor físico.
- Almacena el estado completo de una máquina virtual en archivos.
- Proporciona un aislamiento por fallas y de seguridad a nivel hardware.

2.1.1.2. Hipervisor

Hipervisor o Virtual Machine Monitor (VMM) es una plataforma que permite utilizar múltiples sistemas operativos o máquinas virtuales en un único equipo central. El Hipervisor suministra un entorno completamente aislado para cada máquina virtual y administra la comunicación entre el hardware subyacente de la maquina física y los sistemas operativos de las máquinas virtuales.

Los hipervisores se clasifican según su tipo o según su diseño [10].

- **Hipervisor Tipo 1:** Funciona como sistemas operativos y se ejecuta en el hardware de la maquina física (figura 2). Además, suministra varios entornos de virtualización (máquinas virtuales) independientes entre sí, en donde se instalan los sistemas operativos deseados.

  Algunos hipervisores Tipo 1 como Microsoft Hyper-V, Citrix XenServer, VMware ESX Server y Oracle VM, se encuentran disponibles en el mercado.

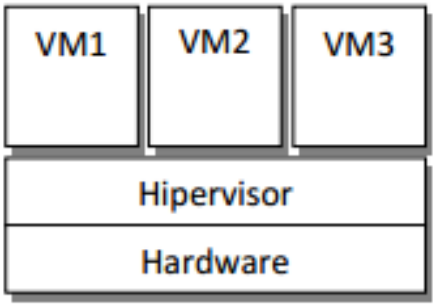

Figura 2. Hipervisor Tipo 1. Fuente: [10]

- **Hipervisor Tipo 2:** Se conoce como hipervisor "hosted", estos se ejecutan en un sistema operativo como si fuera una aplicación más.

  Actualmente, los hipervisores más utilizados de este tipo son Microsoft Virtual Server, VMware Server y Virtual Box.

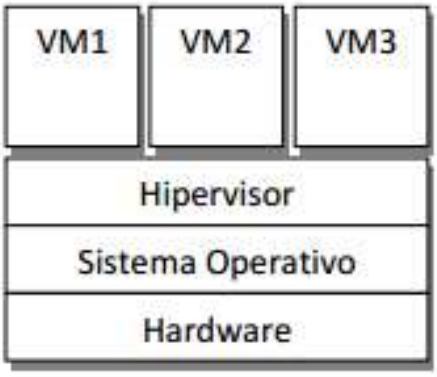

Figura 3. Hipervisor Tipo 2. Fuente: [10]

- **Hipervisor Monolítico:** El diseño de este hipervisor implica el uso de controladores de dispositivos hardware alojados y gestionados por el hipervisor, Algunos de estos, crean una máquina virtual, la cual realiza funciones de consola de servicio (figura 4).

  El hipervisor monolítico más conocido es VMware ESX Server.

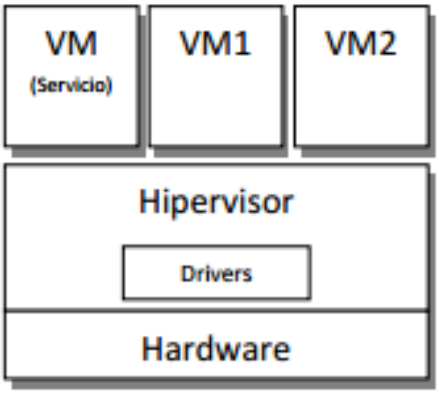

Figura 4. Hipervisor Monolítico. Fuente: [10]

- **Hipervisor con Microkernel:** No necesitan controladores. Estos usan un método para acceder a los dispositivos por medio de una máquina virtual conocida como “root” o “parent” dependiendo del hipervisor, en donde se instala cualquier sistema operativo y se agregan los controladores de cada dispositivo. Las demás máquinas virtuales ingresan a los terminales por medio de la máquina “root”.

  Actualmente el Hipervisor con microkernel más conocido en el mercado es Microsoft Hyper-V.

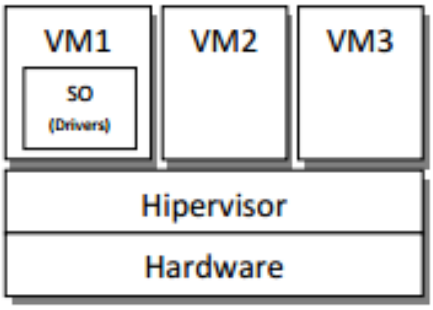

Figura 5. Hipervisor con Microkernel. Fuente: [10]

2.1.2. Tipos de Virtualización

2.1.2.1. Virtualización de Servidores.

Se ejecutan varios sistemas operativos de manera simultánea en un único servidor físico por medio de máquinas virtuales. De esta forma, se reducen los costos y se distribuyen las cargas de trabajo, mejoran el rendimiento de las aplicaciones y la disponibilidad.

Gracias a la virtualización, se utiliza la totalidad de la capacidad de los servidores ya que comúnmente, estos funcionan a menos del 15% de su capacidad total, causando mayor complejidad y aumento de servidores físicos [11].

Entre los métodos más populares de virtualización de servidores se encuentran: Emulación, Virtualización Completa, Paravirtualización y Virtualización a nivel de Sistema Operativo.

2.1.2.2. Virtualización de Redes

Es la simulación completa de una red física en software. Permite que las aplicaciones se ejecuten en una red virtual del mismo modo que en una red física pero con ventajas operacionales independientes del hardware. Esta red virtualizada brinda dispositivos y servicios de red lógicos (puertos, switches, enrutadores, etc.).

2.1.2.3. Virtualización de Escritorios

Es el proceso de separación entre el escritorio (datos y programas del usuario) y la maquina física, es decir, el escritorio se almacena remotamente en un servidor central en lugar del disco duro del equipo del usuario. Por lo cual, la virtualización de escritorios es un servicio administrado que le permite a los usuarios ingresar remotamente a sus escritorios desde cualquier dispositivo.

2.1.2.4. Almacenamiento definido por software (SDS)

En este tipo de virtualización el software de almacenamiento es independiente del hardware, es decir, se separan los discos y las unidades flash y se combinan con el fin de formar un espacio de almacenamiento de alto rendimiento, proporcionado como software. El almacenamiento definido por software brinda una mayor eficiencia operativa y disponibilidad [11].

2.1.3. Ventajas de la Virtualización

La virtualización se caracteriza por aumentar la escalabilidad, flexibilidad y agilidad del hardware físico, ayudando a reducir significativamente los costos. A continuación se describen algunas ventajas que trae consigo la virtualización [11]:

- Disminuye el número de servidores físicos.
- Permite desplegar múltiples sistemas operativos en un solo host físico.
- Centraliza, gestiona y automatiza procesos.
- Reduce los costos gracias al aumento de la eficiencia y la flexibilidad en el uso de los recursos.
- Recuperación rápida en caso de fallos.
- Facilita las copias de seguridad.
- Se adapta a las variaciones de la carga de trabajo.

2.2. REDES NGN

Las redes de nueva generación surgen para satisfacer las necesidades de los usuarios, con el fin de facilitar la convergencia entre diferentes redes y servicios. Esta convergencia no implica la sustitución de las redes existentes, sino por el contrario, permite que las redes convencionales puedan evolucionar, adaptarse y hacer parte de las NGN, ya que estas redes conforman la principal infraestructura para el transporte de la información y la conectividad de la personas.

Según la ITU en la recomendación Y.2001 “Una red de próxima generación (NGN) es una red basada en paquetes capaz de proveer servicios, incluyendo los servicios de telecomunicaciones, utilizando múltiples anchos de banda y

tecnologías de transporte con QoS habilitada, en las cuales las funciones relacionadas con los servicios son independientes de las tecnologías de transporte subyacentes. Además, ofrece a los usuarios acceso sin restricciones a diferentes proveedores de servicios y soporta movilidad generalizada, lo cual permitirá la provisión consistente y ubicua de servicios a los usuarios" [12].

2.2.1. Arquitectura

En general, existen cambios en la arquitectura que traen las redes de nueva generación con respecto a las redes tradicionales, en las cuales el ambiente vertical se convierte en horizontal (figura 6). En la arquitectura vertical cada servicio tiene su propia red, la cual es gestionada de manera independiente, sin ninguna interacción entre los servicios generando múltiples perfiles de usuarios. Por otra parte, la red NGN separa la capa de servicio de la de transporte, manteniendo una estructura de red unificada, en donde se utiliza la misma red para todos los servicios lo que permite el uso de un único perfil de usuario.

Los cambios en la arquitectura de las redes NGN, se especifican a continuación:

- **Núcleo de red:** Unión de diversas redes de transporte construidas a partir de varios servicios individuales, basados en protocolos IP y Ethernet.

- **Red de acceso:** Migración del canal tradicional de voz y datos hacia las infraestructuras convergentes que integran voz IP, cambiando las actuales redes conmutadas que multiplexan por diferentes canales la voz y los datos.

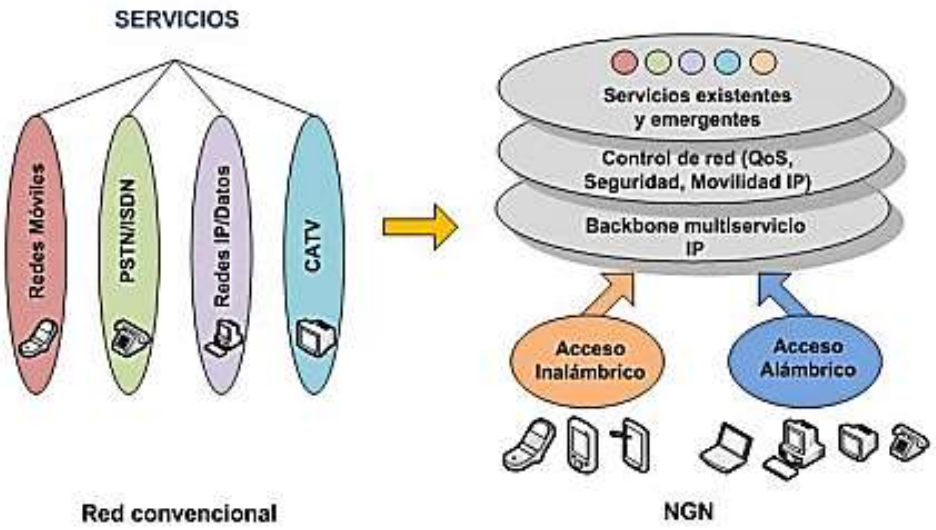

Figura 6. Evolución de la red clásica a la red NGN. Fuente: [13].

La arquitectura de las redes de nueva generación está compuesta por 4 capas que dan flexibilidad y escalabilidad a la red, las cuales están conectadas por medio de interfaces abiertas que facilitan la interconexión e integración de los nuevos servicios [14].

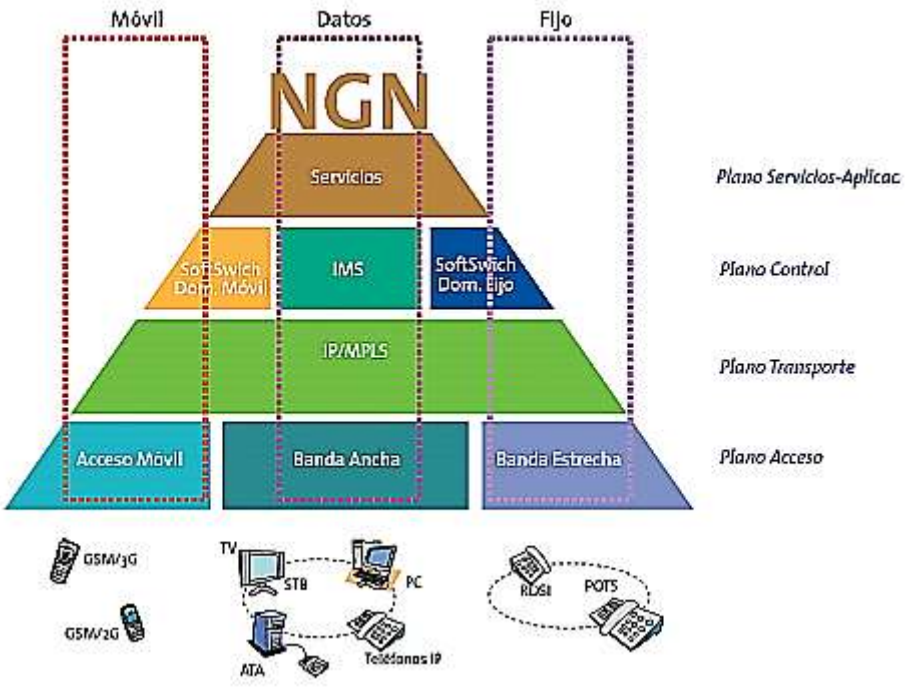

Figura 7. Arquitectura Red NGN. Fuente: [13].

2.2.1.1. Capa de Acceso

Combina todas las tecnologías alámbricas e inalámbricas de la red para conectar los clientes finales e incluye diferentes Media Gateways (MG) que soportan la conexión. Además, convierte el formato de información, de circuito a paquete o de paquete a circuito.

2.2.1.2. Transporte

Es el Core de la red. Transporta los paquetes por la red IP cumpliendo los requerimientos del Servicio/Aplicación y permite la interconexión e interoperabilidad entre redes de acceso. También, es el responsable de proporcionar QoS requerido de extremo a extremo.

2.2.1.3. Capa de Control

Brinda las funciones de gestión de los servicios y de la red. En esta capa se encuentra el servidor de llamadas que proporciona las funciones de control de llamadas en tiempo real, control de acceso de la Media Gateway, asignación de recursos, procesamiento de protocolo, enrutamiento y autenticación.

La capa de control está conformada por los equipos de señalización (SG, Signaling Gateway) y de procesamiento de llamadas (MGC, Media Gateway Controller). El MGC es también conocido como Softswitch, servidor de llamadas o agente de llamadas.

2.2.1.4. Capa de Servicio

En esta capa se encuentran los servicios y aplicaciones disponibles a la red. Estos servicios son independientes de la tecnología de acceso y de la ubicación del usuario, dispuestos de manera centralizada con el fin de obtener mayor eficiencia.

2.2.2. Características

A continuación se presentan algunas características principales de las redes NGN, descritas en la recomendación ITU Y.2001 [12]:

- La transferencia basada en paquetes.
- Las funciones de control están separadas en capacidades de portador, llamada/sesión, y aplicación/servicio.
- La separación de la prestación del servicio y el transporte, y el suministro de interfaces abiertas.
- El soporte de servicios, aplicaciones y mecanismos que se basan en la construcción del servicio por bloques (incluye los servicios en tiempo real/de flujo continuo en tiempo no real y multimedia).
- Dispone de capacidades de banda ancha asegurando la calidad de servicio (QoS) de extremo a extremo.
- Implementación de interfaces abiertas para el interfuncionamiento con las redes tradicionales.
- La movilidad generalizada.
- Acceso de los usuarios a diversos proveedores de servicios sin limitaciones.
- Esquemas de identificación variados.
- Características unificadas para el mismo servicio desde la percepción del usuario.
- Convergencia entre el servicio fijo y móvil.
- Independencia entre las funciones relacionadas con el servicio y las tecnologías de transporte subyacentes.
- Soporte de varias tecnologías de última milla.
- Cumplimiento de las exigencias reglamentarias relacionadas con las comunicaciones de emergencia, seguridad, privacidad, interceptación legal, etc.

2.3. IMS

IP Multimedia Subsystem (IMS) fue desarrollado por la organización de estandarización 3GPP (Third Generation Partnership Project), en la cual IMS es definida como una arquitectura unificada que admite diferentes servicios multimedia con mecanismos de calidad de servicio (QoS) sobre una red totalmente IP. IMS soporta las sesiones aplicativas en tiempo real (voz, video, conferencia, etc.) y sesiones multimedia en tiempo no real (push to talk, presencia, mensajería instantánea, etc.), Además, integra el concepto de convergencia de servicios soportados por redes de naturaleza distinta como fijo, móvil o Internet [15].

IMS es concebido para ofrecer a los usuarios la posibilidad de establecer sesiones multimedia usando todo tipo de acceso de alta velocidad y conmutación de paquetes IP, con el fin de proveer una red IP multi-servicio, multi-acceso, confiable y segura. El propósito principal de la infraestructura IMS es proporcionar control de la sesión en el núcleo (core) de la red al tiempo que permite otro tipo de apoyo necesario para proporcionar esos servicios, independientemente del medio.

2.3.1. Arquitectura IMS

IMS presenta una arquitectura horizontal en capas, cada una con diferentes funcionalidades que proporcionan flexibilidad e independencia de las tecnologías de acceso [16].

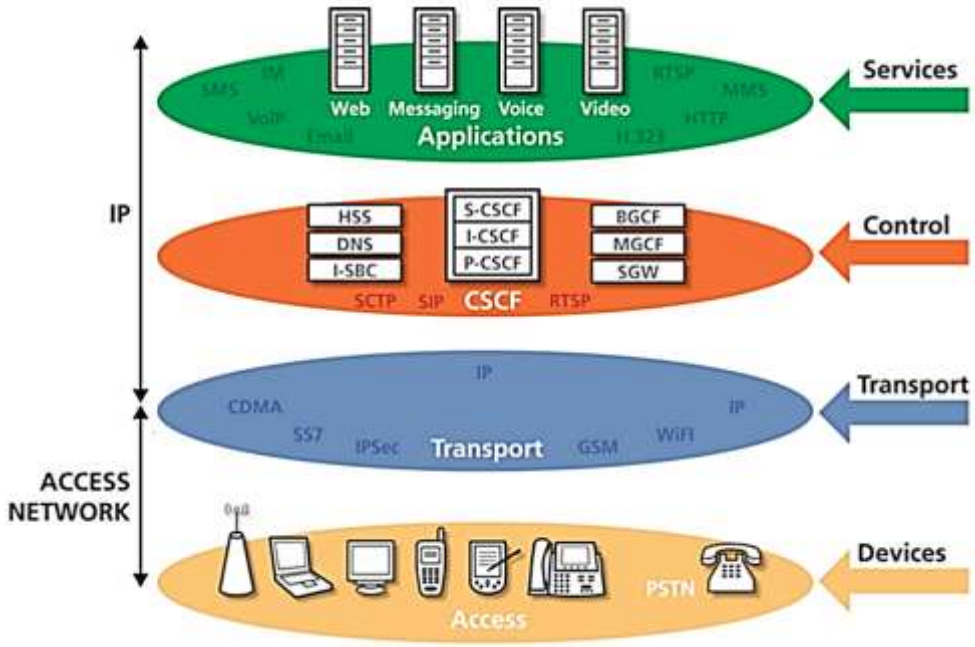

Figura 8. Arquitectura IMS. Fuente: [17].

- **Capa de acceso:** Representa todo acceso de alta velocidad tal como: “UMTS Terrestrial Radio Access Network” o “UTRAN”, “CDMA2000”

tecnología de acceso de banda ancha, “xDSL”, redes de cable, “Wireless IP”, “WiFi”, etc., con el fin de acceder a la red IMS. Estas redes de acceso deben tener la capacidad de soportar la conmutación de paquetes, ya que IMS utiliza este mecanismo. Si por el contrario, se requiere de la conmutación de circuitos se debe realizar la conversión de la información de circuitos a paquetes.

- **Capa de transporte:** Puede representar una red IP. Esta red IP podrá integrar mecanismos de calidad de servicios con MPLS (Multi-Protocol Label Switching), Diffserv (Differentiated services), RSVP (Resource Reservation Protocol), etc. La capa de transporte está compuesta de enrutadores conectados por una red de transmisión y distintas pilas de transmisión que pueden ser contempladas para la red IP.

- **Capa de control:** Es la capa donde se centra IMS. Se compone principalmente de controladores de sesión, quienes toman el control del encaminamiento de la señalización entre usuarios y de la petición de servicios. Estos nodos son identificados como CSCF (Call Session Control Function) que es un conjunto de servidores SIP o proxy que procesan paquetes de señalización SIP (Session Initiation Protocol) en IMS, por lo cual se introduce un ámbito de control de sesiones sobre el campo de paquetes.

- **Capa de aplicación:** Esta capa está compuesta por los servidores de aplicación: AS (Application Server) y servidores de media: MRF (Multimedia Resource Function) o IP MS (IP Media Server), que brindan los servicios a los usuarios. En esta capa el operador puede integrar nuevos servicios y funcionalidades, ofrecidos por el mismo o por terceros, gracias a la capa de control.

En la arquitectura IMS, se pueden agrupar los principales nodos según la función que desempeñan.

2.3.1.1. CSCF

El CSCF (Call Session Control Function) es considerado como el núcleo de IMS, ya que realiza funciones de control de sesión y de direccionamiento. El CSCF se clasifica según su funcionalidad en tres entidades:

- **P-CSCF (Proxy CSCF):** Primer punto de contacto del usuario con la red IMS. Funciona como entrada y salida de la red debido a que las peticiones que inicia un terminal IMS, son recibidas por este nodo [16]. El P-CSCF puede actuar como un Proxy Server SIP cuando dirige los mensajes SIP al destinatario y como un User Agent SIP cuando termina la llamada por errores en el mensaje SIP recibido. Entre sus funciones están admitir solicitudes SIP, la compresión/descompresión de mensajes SIP, encaminamiento y envió de métodos SIP emitidos por el terminal S-CSCF, generación de CDR (Call Detail Record), entre otros [18].

- **I-CSCF (Interrogating CSCF):** Es el punto de contacto entre la red IMS y el operador. Es un nodo intermedio que asigna un S-CSCF a cada usuario en la fase de registro, ayuda a otros nodos a encontrar el siguiente salto de los mensajes SIP, a orientar la señalización y a obtener la dirección del S-CSCF por parte de HSS (Home Subscriber Server). Además, posee una interfaz de comunicación con el HSS, utilizando el protocolo DIAMETER [16].

- **S-CSCF (Serving CSCF):** Es el nodo central en el plano de señalización. Su función principal es realizar el control y mantenimiento de la sesión del usuario en la red IMS con el fin de poder demandar servicios. Además, emula diferentes funciones como Registrar (acepta métodos SIP de registro), Proxy Server (acepta métodos SIP y los encamina) y User Agent (termina métodos SIP). El S-CSCF al igual que el I-CSCF, utiliza el protocolo DIAMETER en la interfaz de comunicación con el HSS [16].

2.3.1.2. HSS

El Home Subscriber Server (HSS) es la principal base de datos que contiene toda la información necesaria de suscripción relacionada con el usuario para el establecimiento de las sesiones multimedia en la red. Este nodo almacena la información del perfil de cada usuario como identidad, datos de registro, parámetros de acceso, información de seguridad, ubicación, el S-CSCF designado para cada usuario y el perfil de los servicios asignados.

Pueden existir varios HSS en una misma red de tal manera que hay que conocer el HSS al cual el usuario está asociado. Para ello es necesario utilizar el nodo SLF (Subscription Location Function), que dirige una petición de registro hacia el HSS en donde se encuentra la información del usuario para validar las identidades de los mismos [18]. Las identidades de los usuarios pueden ser públicas donde otros usuarios pueden usarlas para conectarse con el usuario en cuestión, o privadas en donde la identidad es asignada en la red del operador y solo está disponible para procesos de registro.

2.3.1.3. AS

Los servidores de aplicaciones (AS, Application Servers) de la red son entidades SIP que albergan y ejecutan servicios IMS y se encuentran en la capa de aplicación. En una red existen varios servidores de aplicaciones que pueden implementar uno o diferentes servicios creando una experiencia unificada para el usuario [19].

2.3.2. Protocolos en IMS

A continuación se describen los principales protocolos de comunicación utilizados en IMS [16].

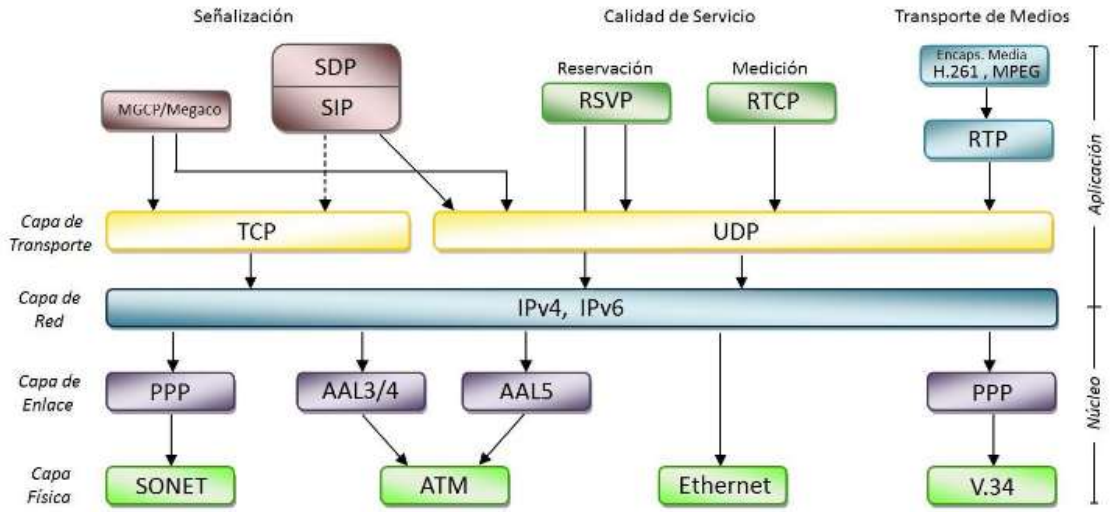

Figura 9. Protocolos usados en IMS. Fuente: [19].

2.3.2.1. SIP

El protocolo de inicio de sesión SIP (Session Initiation Protocol) fue definido por el grupo IETF (Internet Engineering Task Force) en la RFC 3261 para el control y señalización de sesiones. Razón por la cual, el grupo 3GPP lo seleccionó como protocolo principal para la tecnología de IMS.

SIP es un protocolo de control simple y flexible que permite registrar, establecer, modificar y finalizar sesiones multimedia, en donde se intercambia voz, video, texto, imágenes u otros archivos en tiempo real entre usuarios finales. SIP realiza la comunicación entre usuarios (llamadas, videoconferencia o mensajería instantánea) por medio de los protocolos RTP, RTCP y SDP, en donde RTP/RTCP transportan los datos multimedia en tiempo real y SDP describe las características y capacidades en la sesión.

Una de las características de SIP es que está basado en texto, ya que es un protocolo de señalización y no un protocolo de intercambio de datos de usuario, lo que hace que su implementación sea simple y flexible. Además, se fundamenta en el modelo cliente/servidor al igual que HTTP, en donde intercambia mensajes cortos de señalización y no grandes volúmenes de datos, y utiliza el concepto “Uniform Resource Location” o “URL SIP” al igual que SMTP, empleando así algunas funcionalidades de los protocolos de internet existentes como HTTP y SMTP.

2.3.2.1.1. Entidades SIP

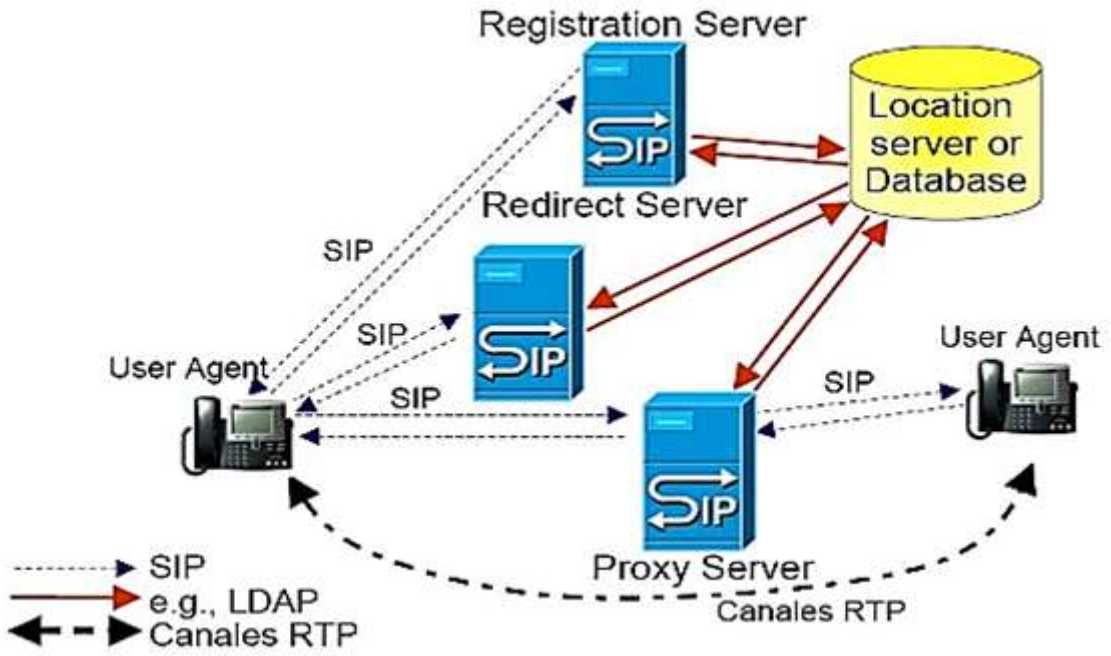

Figura 10. Entidades SIP. Fuente: [20].

SIP define dos tipos de entidades: los clientes y los servidores [16]:

- **Servidor Proxy (Proxy Server):** Recibe las solicitudes de los clientes, hace las modificaciones pertinentes y las encamina a otros servidores.
- **Servidor de Redireccionamiento (Redirect Server):** Acepta solicitudes SIP, traduce la dirección SIP de destino en una o varias direcciones de red y las devuelve al cliente.
- **Agentes de Usuario (AU, User Agent):** Es una aplicación sobre un equipo de usuario final (un PC o un teléfono IP) que emite y recibe solicitudes SIP.
- **Registrador (Registration Server):** Gestiona el registro de los usuarios, aceptando las solicitudes SIP REGISTER. El usuario indica al registrador por un mensaje REGISTER la dirección donde está localizado (dirección IP) y el registrador actualiza una base de datos de localización.

2.3.2.1.2. Mensaje SIP

Los mensajes SIP son utilizados para establecer la conexión y el control de las llamadas. Existen dos tipos de mensajes SIP: Request (petición) y Response (respuesta) [19].

- Mensaje de petición

  Existen seis métodos básicos SIP que describen las peticiones de los clientes al establecer una comunicación:

MENSAJE	FUNCION
INVITE	Inicio de llamada.
ACK	Confirma el establecimiento de una sesión.
OPTION	Consulta a otro host sobre las capacidades requeridas por la sesión.
BYE	Finaliza una sesión previamente establecida.
CANCEL	Cancela una petición pendiente.
REGISTER	Registra un cliente SIP.

Tabla 1. Clases de peticiones SIP.

- Mensaje de respuesta

  Una vez recibidas y procesadas las solicitudes SIP, el destinatario de este requerimiento devuelve una respuesta SIP. Existen seis clases de respuestas:

CLASE	INTERPRETACION	DESCRIPCION
1xx	Información	La petición fue recibida y está en proceso de tratamiento.
2xx	Éxito	La petición fue recibida, entendida y aceptada.
3xx	Redirección	La petición necesita otros procesos antes de determinar si se realiza o no.
4xx	Errores del cliente	La petición no es soportada o interpretada por el cliente.
5xx	Errores del servidor	La petición supuestamente valida ha fracasado por fallas en el servidor.
6xx	Fallas globales	La petición no puede ser procesada por ningún servidor.

Tabla 2. Clases de respuestas SIP.

2.3.2.2. SDP

SDP (*Session Description Protocol*) es un protocolo de la capa de aplicaciones definido por el grupo de estandarización IETF, el cual describe los parámetros que permiten notificar, iniciar y establecer sesiones multimedia. Por medio de SDP, los usuarios de una sesión pueden mostrar sus capacidades multimedia y especificar el tipo de sesión que desean establecer. Además, SDP permite a los usuarios finales decidir el flujo multimedia que tendrá la sesión como voz,

audio, video, etc., los códecs necesarios para soportar cada flujo y las configuraciones de los mismos [19].

La carga SDP se encuentra en el cuerpo de los mensajes SIP, donde la información dentro del cuerpo SDP se encuentra estructurada de la siguiente manera [21]:

- Descripción de la sesión.
- Descripción de tiempos.
- Descripción multimedia.

2.3.2.3. DIAMETER

El desarrollo del protocolo Diameter está basado en el protocolo RADIUS, el cual da soporte a las políticas y protocolos de autenticación, autorización y contabilidad (*AAA, Authentication, Authorization and Accounting*) en el acceso de los usuarios a la red IMS. En IMS, el S-CSCF, I-CSCF y los servidores de aplicación SIP usan Diameter en la capa de servicio y en el intercambio de la información de usuario con el servidor de base de datos HSS [19].

El protocolo Diameter se puede dividir en [22]:

- **Protocolo Base Diameter:** Usado para negociar las capacidades y el control de errores.
- **Aplicaciones Diameter:** Define funciones de cada aplicación disponible.

2.3.2.4. RTP

El protocolo de transmisión en tiempo real RTP (*Real Transport Protocol*) es definido por el grupo IETF en la RFC 3550, el cual permite el intercambio en tiempo real de servicios multimedia como voz, video, datos, etc., entre dos usuarios finales. Además, permite identificar el tipo de códec, implementar los números de secuencia de los paquetes IP, monitorización en el envío de los paquetes al destino e identificar el tipo de información transmitida [22].

RTP transporta los mensajes sobre paquetes, los cuales tiene la siguiente estructura:

- Cabecera de longitud fija de 12 octetos.
- Cuerpo del mensaje o carga útil de longitud variable.

RTP trabaja sobre el protocolo de transporte UDP/IP, pero es importante aclarar que no garantiza la entrega de los paquetes en el destino y no proporciona calidad de servicio (QoS) en una comunicación en redes IP, razón por la cual, RTP utiliza el protocolo de control RTCP.

2.3.2.5. RTCP

El protocolo de control de tiempo real RTCP (Real-Time Transport Control Protocol) controla el flujo multimedia enviado por RTP. RTCP es fundamental, ya que transmite mensajes de control periódicos a todos los participantes de la sesión, enviando el identificador de origen RTP y monitorizando los parámetros de QoS de la sesión, ya que incluye información sobre los paquetes perdidos, retardo, "jitter", entre otras [22].

El control del flujo multimedia se realiza por medio de cinco tipos de paquetes [16]:

- **SR (Sender Report):** Tiene estadísticas para la transmisión y recepción de paquetes IP de los participantes de la sesión que son emisores.
- **RR (Receiver Report):** Tiene estadísticas de recepción de una sesión para los usuarios que son receptores pero no son emisores.
- **SDES (Source Description):** Brinda datos de descripción de los usuarios, tales como: nombre, e-mail, teléfono, etc.
- **BYE:** Indica a un usuario cuando ha terminado su participación en una sesión.
- **APP:** Paquete de señalización específico para una aplicación.

2.4. QoS

Quality of service (QoS) según la UIT se define como "La totalidad de las características de un servicio de telecomunicaciones que determinan su capacidad para satisfacer las necesidades explícitas e implícitas del usuario del servicio" [6]. La QoS se puede medir en diferentes escenarios, haciendo uso de diversos parámetros tales como la tasa de errores, el ancho de banda, el rendimiento, la disponibilidad, el "*jitter*" (variabilidad temporal), el retraso en la transmisión, el tráfico, la perdida de paquetes, etc., esto con el fin de gestionar o evitar la congestión de la red y así asegurar un adecuado nivel de QoS para la transmisión de los datos.

2.4.1. Modelos de QoS

Existen tres modelos de implementación de QoS [23], [24]:

- **Best-Effort:** Es un modelo simple de servicios aplicado en internet, ya que no tiene políticas explícitamente definidas. En este modelo no se aplica QoS en el tráfico, es decir, no garantiza algún tratamiento o recurso especifico al flujo de datos, debido a que todo paquete es tratado de la misma forma, sin preferencia alguna.

  Este modelo se caracteriza por ser altamente escalable, no garantiza recursos ni diferencia el tipo de servicio y no necesita mecanismos o

configuraciones específicas. Además, no asegura tasa de transferencia, retraso o flexibilidad y utiliza el modelo de cola FIFO (First In First Out) para realizar las transmisiones de datos en la red.

- **IntServ (Integrated Services):** Es un modelo de implementación de servicios bajo demanda, el cual debe garantizar los recursos necesarios (ancho de banda, retardo, etc.) a una aplicación específica desde el origen hasta el destino. Este modelo asegura las condiciones de operación de cada una de las sesiones que se ejecutan en la red.

  Este modelo no es escalable para administrar grandes flujos de tráfico, ya que requiere de una gran cantidad de señalización para gestionar la disponibilidad de QoS, lo que hace que el modelo sea difícil de implementar.

  IntServ es caracterizado por negociar las condiciones de QoS antes de iniciar una comunicación, reservar los recursos de la red para un servicio en particular, utilizar los servicios de RSVP (Resource Reservation Protocol) y adaptarse a las demandas de los tipos de tráfico y aplicaciones.

- **DiffServ (Differentiated Services):** La infraestructura de la red es la encargada de reconocer las distintas clases de tráfico y aplicar diferentes políticas para cada tipo de tráfico, de manera escalable por toda la red. Los paquetes enviados tienen un código especifico DSCP (Differentiated Services Code Point) que les permite identificar el tipo de tráfico al cual pertenecen. De esta manera, DiffServ asigna prioridades a los paquetes enviados en la red.

  Este modelo se caracteriza por ser flexible y escalable, le permite a los routers modificar su comportamiento de envío, controla el tráfico y reduce la carga de los dispositivos de red.

2.5. Video Llamada

El servicio de video llamada o videotelefonía es definido por la UIT en la recomendación F.700, como: “Teleservicio conversacional audiovisual que proporciona una transferencia simétrica y bidireccional en tiempo real de voz e imágenes a color y en movimiento entre dos ubicaciones (de persona a persona) a través de las redes implicadas. El requisito mínimo es que, en condiciones normales, la calidad de las imágenes transmitidas sea suficiente para representar de manera adecuada y con fluidez los movimientos de una persona, que se muestra en un plano de cabeza y hombros” [25].

Una video llamada permite establecer una comunicación entre dos o más personas en lugares diferentes a través de la red. Este es un servicio

compuesto por audio y video, cada uno con niveles de calidad diferentes, en donde ambos deben sincronizarse de tal manera que a los usuarios les parezca que las percepciones visuales y auditivas coincidan de forma natural con los acontecimientos, es decir, que el movimiento de los labios coincida con la voz.

3. MODELO DE CALIDAD DE SERVICIO (QoS)

Los servicios basados en paquetes se ven afectados por diferentes parámetros y condiciones de red que pueden deteriorar la QoS de los servicios que se transmiten, principalmente los que se prestan en tiempo real como juegos en línea, VoIP, Video Streaming, entre otros. Por esta razón, la red debe ofrecer condiciones favorables para mantener o mejorar la QoS, garantizando una calidad de experiencia óptima a los usuarios finales. Para garantizar una QoS óptima se debe realizar una correcta gestión de los recursos de red, que permita brindar condiciones adecuadas para la transmisión de datos a las aplicaciones, cumpliendo ciertos requerimientos que son exigidos por los diferentes servicios, como trasmisiones de datos sin pérdidas, ancho de banda mínimo, retardo mínimo de extremo a extremo, mínima variación de retardo, buenos tiempos de respuesta, entre otros comportamientos esperados.

En el presente capítulo se realiza un análisis de los diferentes parámetros que permiten medir la calidad de servicio de una video llamada, exponiendo claramente sus significados, dependencias y ambientes de red, clasificándolos según la dependencia que exista entre ellos y exponiendo los parámetros seleccionados que serán propósito de análisis. Además, se describe la infraestructura general de la red utilizada, especificando el hardware, el software y la topología. Por último, se realiza una caracterización de la red, analizando la escalabilidad tanto vertical como horizontal para verificar la eficiencia y el rendimiento que esta tiene al momento de implementar un servicio.

3.1. PARAMETROS DE CALIDAD

3.1.1. Factores de Calidad

Después de revisar los documentos, recomendaciones y artículos anexos en el estado del arte acerca de las variables que afectan la QoS en redes IMS, redes convergentes y redes virtualizadas, se presenta una recopilación de los principales parámetros que son propósito de estudio, con el fin de determinar cuáles de ellos son los más apropiados para estructurar un modelo de QoS. Estos parámetros se escogieron porque afectan la comunicación y pueden ser medidos en este tipo de redes.

A continuación, se da una breve descripción de cada uno de ellos.

3.1.1.1. Ancho de banda

En el contexto de redes digitales, el ancho de banda se define como la capacidad de transporte del canal para la transmisión de los datos de un punto a otro en un periodo de tiempo dado (segundos) [26]. El ancho de banda, medido en bits por segundo (bit/s) o múltiplos de esta unidad (Kbit/s, Mbit/s y

Gbit/s), depende de las condiciones de la red, de los medios físicos que la conforman, así como la cantidad de información que fluirá a través de ella.

3.1.1.2. Retardo "extremo a extremo" o latencia

Es el tiempo que demora un paquete en ser enviado desde el transmisor hacia el receptor a través de la red. Los servicios de comunicación en tiempo real son muy sensibles a este efecto, por lo que el retardo máximo en un sentido entre emisor y receptor es de 150 mseg para lograr alta calidad; el retardo entre 150 y 400 mseg es aceptable pero no es un valor ideal y retardo superior a 400 mseg disminuye la calidad del servicio y puede obstaculizar la interactividad en la comunicación [27], [28].

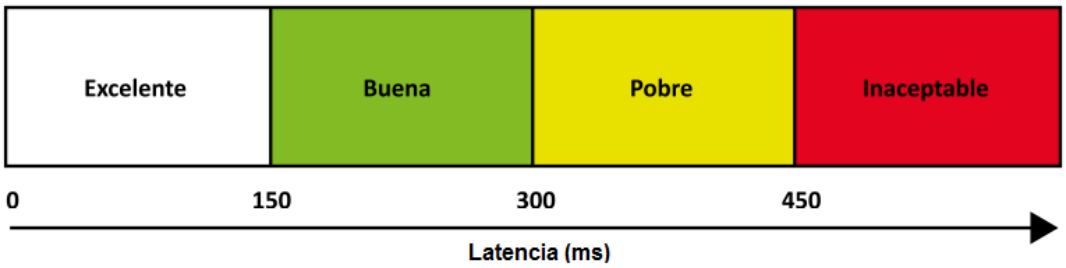

Figura 11. Tiempo de transmisión en un sentido. Fuente: [26].

La latencia se compone básicamente de dos categorías de retardo:

- **Retardo de red fijo**: Es un valor predecible y constante que siempre se presenta en el tráfico de la red. El retardo se compone del tiempo de serialización, el cual depende de la velocidad de la interfaz y del tamaño de la trama; y del tiempo de propagación, definido como el tiempo de desplazamiento a través del medio de transmisión, en algunos casos es despreciable pero cuando se habla de transmisiones vía satélite (enlaces de larga distancia) es muy relevante.

- **Retardo de red variable**: Este valor de retardo depende del tráfico o del nivel de congestión, en este también se incluye el retardo de cola que es la cantidad de tiempo que está un paquete en un buffer de salida.

3.1.1.3. Variabilidad del retardo o "jitter"

El "jitter" está definido como la variación en el tiempo de llegada de los paquetes al ser transmitidos por la red. Los paquetes son enviados en intervalos regulares desde el emisor, por lo que la variación en el tiempo de llegada de los paquetes en el receptor puede ser ocasionada por la congestión de la red, falta de sincronización o por las diferentes rutas que toman los paquetes para llegar al destino. Este problema en la comunicación se puede solucionar utilizando el "Jitter Buffer", el cual almacena temporalmente los paquetes y los envía en intervalos regulares al receptor.

El “jitter” debe ser menor a 100 ms para que pueda ser compensado y no afecte la calidad del servicio [29]. Un ejemplo muy sencillo que explica este parámetro es cuando un paquete llega con un retardo de 300 ms y el siguiente llega con un retardo de 320 ms, entonces el “jitter” es la diferencia entre estos dos valores que es 20 ms.

3.1.1.4. Pérdida de paquetes

La pérdida de paquetes es la relación entre los paquetes perdidos y el total de paquetes enviados. Existen varios factores en la red que ocasionan la pérdida de paquetes como congestión o saturación en el canal donde se están trasmitiendo los datos, fallas en el enlace, o un ancho de banda insuficiente. El efecto que produce la pérdida de paquetes son las pausas ocasionales en la reproducción de video y el audio, por lo cual la calidad del servicio se ve afectada [30].

En protocolos no orientados a conexión como RTP no se utiliza el reenvío de paquetes para aplicaciones en tiempo real, debido a que aumentan el tiempo de espera congestionando aún más la red, y reduce la velocidad de trasmisión, por estas razones este tipo de aplicaciones se ejecutan sobre UDP.

3.1.1.5. Velocidad de bits de vídeo

También conocida como Tasa de bits, representa el proceso de codificación por cada imagen de video, en otras palabras, es la compresión de vídeo o multimedia en términos de la cantidad de datos por unidad de tiempo (segundos). Este parámetro es medido en bits por segundo (bps) o megabits por segundo (Mbps) y juega un papel importante en la calidad de video, ya que, cuanto mayor sea la velocidad de bits, mayor será la calidad de la imagen y el vídeo será más claro y nítido [31].

3.1.1.6. Pérdida de fotogramas de video

Es el número promedio de imágenes o fotogramas de video perdidos durante el establecimiento de una llamada. Este parámetro puede llegar a ser más útil en la evaluación de calidad de video en comparación con la perdida de paquetes ya que un fotograma de video se puede distribuir en varios paquetes [31].

3.1.1.7. Tiempo de retardo Round-Trip (RTT)

Se define como el tiempo que tarda un paquete de datos en ir desde el cliente hasta el servidor y de regreso al cliente, pasando por los nodos intermedios que componen la ruta. Para una comunicación de voz es el tiempo necesario para que el interlocutor escuche la respuesta a su mensaje. Según la UIT G.114 [28], este retardo no puede superar los 300 ms en servicios de voz. El retardo RTT depende de muchos factores como retardos de propagación, de

empaquetamiento y de procesamiento de paquetes, números de nodos intermedios en la ruta y cantidad de tráfico en la red [27].

3.1.1.8. Tasa de fotogramas

Este parámetro se representa en frames por segundo (FPS) y es la tasa o frecuencia con la cual un dispositivo muestra consecutivamente un cierto número de fotogramas o frames que se producen en un determinado tiempo.

3.1.1.9. Densidad de ráfaga

La densidad de ráfaga es la fracción de paquetes de datos RTP en un periodo de ráfaga determinado, que son extraviados o descartados desde el comienzo de la recepción. Un periodo de ráfaga es el tiempo en que los paquetes se pierden o son eliminados por no llegar en el tiempo establecido. Este parámetro se expresa como un número entero y se calcula dividiendo el número total de paquetes perdidos o descartados dentro de periodos de ráfaga por el número total de paquetes calculados dentro de dichos periodos y por último multiplicando el resultado de la división por 256, limitando su valor máximo a 255 y tomando la parte entera de este resultado [32].

Los siguientes parámetros son tomados de la recomendación ITU-T Y.1540 [33] en donde se especifican diferentes términos acerca de parámetros relacionados con el rendimiento en la trasferencia de paquetes IP y la recomendación ITU-T Y.1541 [34] donde se definen diferentes clases de calidad de servicio según los parámetros usados. En esta recomendación también se especifican los diferentes límites o rangos apropiados y según estos datos se clasifica la calidad de servicio.

3.1.1.10. Retardo de transferencia de paquetes IP (IPTD)

IPTD (IP packet transfer delay) es definido como el retardo unidireccional entre los puntos de medida a la entrada y salida de la red IP. IPTD es el tiempo (t2 - t1) entre el suceso de dos eventos de referencia de paquetes IP, donde t2 > t1 y (t2 - t1) ≤ Tmax.

3.1.1.11. Variaciones de retardo de paquetes IP (IPDV)

IPDV (IP packet delay variations) se aplica para un conjunto de paquetes determinado que va desde un punto de medición MP1 al punto de medición MP2. IPDV se define como la diferencia entre el retardo en un solo sentido de los paquetes seleccionados. Es importante que las aplicaciones de streaming conozcan cuánto varía el retardo en la red para evitar desbordamientos de búferes y sobrecarga. Las pequeñas variaciones de retardo no son importantes, pero las grandes pueden causar retransmisiones de paquetes innecesarias o retrasos largos antes de retransmitir.

3.1.1.12. Relación de error de paquetes IP (IPER)

IPER (IP packet error ratio) se refiere a la porción de paquetes con errores de todos los paquetes recibidos y se define como la relación entre el resultado de paquetes IP con errores, con el total de paquetes IP trasferidos con éxito más los resultados erróneos de paquetes IP, como se observa en la siguiente ecuación:

$$IPER = \frac{N_{Erroneos}}{N_{Exitosos} + N_{Erroneos}} \qquad (1)$$

Donde,

N: Paquetes IP.

3.1.1.13. Relación de perdida de paquetes IP (IPLR)

IPLR (IP packet loss ratio) hace referencia a la cantidad de paquetes perdidos y se define como la relación del total de paquetes IP perdidos con el total de paquetes IP transmitidos, es decir,

$$IPRL = \frac{N_{Perdidos}}{N_{Transmitidos}} \qquad (2)$$

Donde,

N: Paquetes IP.

3.1.1.14. Tasa de paquetes IP espurios (SPR)

SPR (Spurious IP packet rate) es la tasa de paquetes IP espurios o falsos observados durante un intervalo de tiempo específico dividido entre la duración del intervalo de tiempo, lo que es equivalente al número de paquetes IP espurios por segundo de servicio.

3.1.1.15. Relación de paquetes IP reordenados (IPRR)

IPRR (IP packet reordered ratio) es la relación entre el total de paquetes reordenados y el total de paquetes IP transmitidos con éxito.

3.1.1.16. Relación de bloques de pérdida severa de paquetes IP (IPSLBR)

IPSLBR (IP packet severe loss block ratio) es la relación de los bloques de pérdida severa de paquetes con el total de bloques en una población de interés.

3.1.1.17. Relación de paquetes IP duplicados (IPDR)

IPDR (IP packet duplicate ratio) es la relación entre el total de paquetes IP duplicados con respecto al total de paquetes IP transmitidos con éxito menos el número de paquetes IP duplicados.

3.1.1.18. Relación de paquetes IP replicados (RIPR)

RIPR (replicated IP packet ratio) es la relación entre el total de paquetes IP replicados con el total de paquetes IP transmitidos con éxito menos el resultado de paquetes IP duplicados.

En la recomendación UIT-T G.1020 [32], analizan otros parámetros que se deben tener en cuenta en servicios sobre IP, los cuales se definen a continuación.

3.1.1.19. Evento de pérdida de paquetes consecutivos

Este parámetro se produce cuando varios paquetes se pierden al ser enviados en un tren periódico, especificando que los paquetes que se pierden deben ser de manera consecutiva. Esta longitud de paquetes perdidos debe registrarse para cada evento. Por lo general, para hacer esta medición de perdida de paquetes consecutivos se utiliza el número de secuencia de los encabezamientos de los paquetes.

3.1.1.20. Segundo degradado

Un segundo degradado ocurre cuando la relación del total de paquetes perdidos en la UNI (Interfaz Usuario-Red) de entrada y salida de un cierto bloque de paquetes que es analizado durante un intervalo de 1 segundo supera un porcentaje D. Este porcentaje por lo general se establece en 15%, el cual puede variar dependiendo de las condiciones de la red. Por ejemplo, si se tiene que en un segundo se envían 100 paquetes y de estos se pierden 16, se dice que hay una pérdida del 16% en este bloque. Este valor supera al porcentaje D, por lo cual se puede concluir que éste es un segundo degradado.

Para medir este parámetro se puede usar los números de secuencia y las indicaciones de tiempo que contienen los encabezamientos de los paquetes.

3.1.1.21. Cuantificación de fluctuación/variación del retardo IP a corto plazo ($IPDV_{Short\_term}$)

El parámetro de variación de retardo a corto plazo se puede definir como el IPTD máximo menos el IPTD mínimo durante un intervalo de medición corto, donde su unidad de medida es en segundos. Este método es bastante preciso para calcular IPDV en tiempo real.

$$IPDV_{Short\_Term} = IPTD_{max} - IPTD_{min} \qquad (3)$$

3.1.1.22. Ventana de descarte

Este parámetro es el intervalo de tiempo para la llegada de paquetes aceptables, teniendo en cuenta el tiempo de inactividad del paquete (cantidad fija de tiempo que es añadida a la marca de tiempo RTP, dicho tiempo añadido incluye el tiempo que toma el paquete para atravesar la red y cualquier memoria intermedia), con una tolerancia para la llegada temprana o tardía del paquete.

Cuando un paquete llega al destino, se calcula el tiempo de inactividad y si este tiempo ha sobrepasado los límites de la ventana de descarte, el paquete es considerado como descartado.

3.1.2. Clasificación de los parámetros

En esta sección se presenta la clasificación de los parámetros anteriormente analizados, ya que existe cierta dependencia entre ellos. Esto con el fin de seleccionar las variables más apropiadas para estructurar el modelo y así darle mayor eficiencia al mismo.

3.1.2.1. Características físicas de la red

Los siguientes parámetros dependen de las características de la red para la transmisión y procesamiento de los paquetes, como el tipo de conexión, congestión, equipos utilizados, entre otros.

- Ancho de banda
- Velocidad de bits de video
- Tasa de fotogramas

3.1.2.2. Retardo

Estos parámetros están relacionados con el retardo de paquetes entre emisor y receptor en la red. Aunque el retardo Round Trip tiene en cuenta el tiempo de ida y vuelta entre los dos extremos de una comunicación, este valor debe estar por debajo de 300 ms para que la calidad del servicio sea buena. Este valor también es aceptable para la comunicación de extremo a extremo, razón por la cual este parámetro se encuentra dentro de esta categoría.

- Retardo “extremo a extremo” o latencia
- Tiempo de retardo Round-Trip (RTT)

- Retardo de transferencia de paquetes IP (IPTD)

Los siguientes parámetros afectan los retardos en la comunicación pero no perjudican notoriamente la calidad del servicio, es por esto que se los ha seleccionado dentro de esta categoría, ya que, toma tiempo llevar a cabo procesos de reordenamiento o cuando los paquetes son replicados o duplicados se pueden presentar retardos de procesamiento en el receptor.

- Relación de paquetes IP duplicados (IPDR)
- Relación de paquetes IP replicados (RIPR)
- Relación de paquetes IP reordenados (IPRR)

3.1.2.3. Variación del retardo

Los siguientes parámetros están relacionados entre sí, ya que analizan la variabilidad del retardo de los paquetes IP dentro de la red de comunicación.

- Variabilidad del retardo o “jitter”
- Variación de retardo de paquetes IP (IPDV)
- Cuantificación de fluctuación/variación del retardo IP a corto plazo

3.1.2.4. Pérdida de información

Los parámetros clasificados dentro de esta categoría están relacionados con la pérdida de información al establecer una comunicación en la red.

- Pérdida de paquetes
- Perdida de fotogramas de video
- Densidad de ráfaga
- Relación de error de paquetes IP (IPER)
- Relación de pérdida de paquetes IP (IPLR)
- Tasa de paquetes IP espurios (SPR)
- Relación de bloques de pérdida severa de paquetes IP (IPSLBR)
- Evento de pérdida de paquetes consecutivos
- Segundo degradado
- Ventana de descarte

3.1.3. Parámetros Seleccionados

Después de la anterior clasificación, se determinan los parámetros más significativos en cada una de las categorías mencionadas para la estructuración principal de un modelo que permita medir la calidad de servicio de un servicio de video llamada. Los parámetros seleccionados deben abarcar todas o la gran mayoría de las características de las demás variables clasificadas dentro de la categoría, de esta forma se busca que al obtener los

resultados de las medidas, éstas puedan de alguna forma representar la totalidad de la clasificación, simplificando de la mejor manera el modelo y reduciendo el esfuerzo al tomar una globalidad de muestras que arrojan los mismos resultados.

Con el fin de escoger los parámetros más relevantes, se realiza un análisis de cada uno de ellos, estableciendo claramente las variables que representan las cuatro categorías seccionadas en el anterior punto, de tal forma que al tomar las medidas de estos parámetros se pueda estimar el comportamiento de los demás.

3.1.3.1. Características físicas de la red

En esta categoría el primer parámetro a analizar es la velocidad de bits de video, el cual representa la cantidad de datos almacenados en un segundo. Esta variable depende del tipo de codificación o procesos llevados a cabo al momento de empaquetar la información, al igual que el ancho de banda. En cuanto a la tasa de fotogramas es un parámetro que se mide al final de la transmisión, es decir, en el receptor. Este depende de la frecuencia con la cual un dispositivo muestra consecutivamente los fotogramas que le llegan en cierto tiempo.

De este modo, se puede decir que estos parámetros son fijos, y que se establecen antes de iniciar la comunicación, es decir, no varían una vez realizada la video llamada. Razón por la cual el parámetro seleccionado para esta categoría es el ancho de banda, ya que abarca las variables mencionadas anteriormente y es uno de los parámetros más importantes en la red para ofrecer una buena calidad de servicio. Esta variable se explica con más detalle a continuación.

3.1.3.1.1. Ancho de banda

Este es uno de los parámetros más importantes en la red para ofrecer una buena calidad de servicio a los usuarios, ya que entre más información se pueda trasportar en un intervalo de tiempo, los servicios y aplicaciones funcionaran mejor, según sea la capacidad que el servicio requiera. El ancho de banda está directamente relacionado con el tipo de códec que se esté usando, ya que si no se cuenta con un nivel adecuado de ancho de banda que soporte el códec puede presentarse saturación en el canal por insuficiencia en la capacidad y esto puede afectar la calidad de servicio.

En el caso de un servicio de video llamada, es necesario tener en cuenta las características del códec de video y de audio, donde el ancho de banda total equivale a la suma del ancho de banda consumido por el códec de audio y el de video. Es importante aclarar que este valor de ancho de banda del servicio

puede variar de acuerdo al medio de transmisión, la cantidad de usuarios y el tráfico en la red.

El ancho de banda del video depende de diferentes factores, entre ellos se encuentra el tipo de códec, la resolución de la imagen aceptada por el receptor, la velocidad de fotogramas y las características hardware y software del usuario final. En la tabla 3 se observan los diferentes valores de resolución de imagen y velocidad de fotogramas, los cuales crean diferentes experiencias en cuanto a la calidad del servicio y requisitos de ancho de banda, según el códec de video más utilizado.

Códec	Configuración del Cliente	Resolución	Frames/seg	Calidad	Ancho de banda (Kbps)
DviX	Ancho de banda muy bajo	160X120	2	Muy baja	10-20
	Ancho de banda bajo	160x120	10	Baja	60-120
	Ancho de banda medio	320x240	10	Media	150-300
	Alto ancho de banda	352x288	15	Alta	400-800
H.263, H.263+, H.264/MPEG-4	Ancho de banda muy bajo	176x144	2	Muy baja	10
	Ancho de banda bajo	176x144	10	Baja	64
	Ancho de banda medio	352x288	10	Media	192
	Ancho de banda alto	352x288	15	Alta	512
	Ancho de banda muy alto	640x480	30	Muy alta	768

Tabla 3. Ancho de banda según el códec de video. Fuente: [35]

En la tabla 4, se observan las características de diferentes tipos de códec de audio y la estimación del ancho de banda para una sola llamada.

Información de códec				Cálculos de ancho de banda					
Velocidad de bits y códec (kbps)	Ejemplo de tamaño del códec (bytes)	Ejemplo de intervalo del códec (ms)	Mean Opinion Score (MOS)	Tamaño de la carga útil de voz (bytes)	Tamaño de la carga útil de voz (ms)	Paquetes por segundo (PPS)	Ancho de banda MP o FRF.12 (Kbps)	Ancho de banda c/cRTP MP o FRF.12 (kbps)	Ancho de banda Ethernet (Kbps)
G.711 (64 Kbps)	80 bytes	10 ms	4.1	160 bytes	20 ms	50	82.8 Kbps	67.6 Kbps	87.2 Kbps

G.729 (8 Kbps)	10 bytes	10 ms	3.92	20 bytes	20 ms	50	26.8 Kbps	11.6 Kbps	31.2 Kbps
G.723.1 (6.3 Kbps)	24 bytes	30 ms	3.9	24 bytes	30 ms	33.3	18.9 Kbps	8.8 Kbps	21.9 Kbps
G.723.1 (5.3 Kbps)	20 bytes	30 ms	3.8	20 bytes	30 ms	33.3	17.9 Kbps	7.7 Kbps	20.8 Kbps
G.726 (32 Kbps)	20 bytes	5 ms	3.85	80 bytes	20 ms	50	50.8 Kbps	35.6 Kbps	55.2 Kbps
G.726 (24 Kbps)	15 bytes	5 ms		60 bytes	20 ms	50	42.8 Kbps	27.6 Kbps	47.2 Kbps
G.728 (16 Kbps)	10 bytes	5 ms	3.61	60 bytes	30 ms	33.3	28.5 Kbps	18.4 Kbps	31.5 Kbps
G722_64k (64 Kbps)	80 bytes	10 ms	4.13	160 bytes	20 ms	50	82.8 Kbps	67.6 Kbps	87.2 Kbps
ilbc_mode_20 (15.2Kbps)	38 bytes	20 ms	NA	38 bytes	20 ms	50	34.0 Kbps	18.8 Kbps	38.4 Kbps
ilbc_mode_30 (13.33Kbps)	50 bytes	30 ms	NA	50 bytes	30 ms	33.3	25.867 Kbps	15.73 Kbps	28.8 Kbps

Tabla 4. Ancho de banda según el códec de audio. Fuente: [36]

3.1.3.2. Retardo

En esta categoría el parámetro que representa los retardos presentes en la red al establecer una comunicación, es el retardo “extremo a extremo” o latencia. Para esta categoría se tratan de clasificar los retardos durante el establecimiento del servicio, desde el momento en que un usuario comienza una video llamada, pasando por todos los protocolos de señalización, memorias intermedias, procesos de empaquetamiento, codificación y envío de información hasta la llegada de la información al receptor, la interpretación de la información, etc. Se descartan los demás parámetros de ésta categoría, ya que, en primera instancia el tiempo de retardo de Round-Trip (RTT), tiene en cuenta el tiempo de ida y vuelta de un paquete entre los dos extremos de una comunicación, pero en un servicio de video llamada implementada sobre una red IMS, objetivo del presente trabajo de grado, utiliza el protocolo RTP para enviar los paquetes entre los dos extremos. RTP es transportado sobre UDP, el cual es un protocolo no orientado a conexión (no utiliza el reenvío de paquetes para aplicaciones en tiempo real), por lo que solo es necesario tener en cuenta los retardos presentes en un solo sentido de la comunicación y así eliminar este parámetro dentro de esta clasificación. Por otra parte, el retardo de transferencia de paquetes IP (IPTD), es un parámetro que solo tiene en cuenta dos puntos de referencia en el trayecto del flujo de paquetes y no tiene en cuenta los demás retardos que se pueden presentan en la red.

Los demás parámetros como relación de paquetes IP duplicados (IPDR), relación de paquetes IP replicados (RIPR), relación de paquetes IP reordenados (IPRR) clasificados dentro de esta categoría, no afectan de manera drástica la calidad del servicio, ya que son procesos que se realizan en el receptor y producen retardos mínimos, por lo cual se puede decir que se

encuentran dentro de la latencia, por esta razón no se tienen en cuenta para esta categoría.

3.1.3.2.1. Retardo de “Extremo a Extremo” (latencia)

Como se había mencionado anteriormente, el retardo de extremo a extremo o latencia es el parámetro escogido para esta categoría, ya que es la suma de los retardos temporales que experimenta un paquete en la red. Dentro de los retardos que constituyen la latencia se encuentran los siguientes retardos:

- **Retardo de propagación:** Tiempo que tarda la señal en llegar desde el inicio al final del sistema de transmisión.
- **Retardo de trasmisión:** Tiempo que tarda el transmisor en colocar los bits del paquete en el canal. Este retardo depende del tamaño del paquete.
- **Retardo de procesamiento:** Tiempo que tarda el router en determinar el siguiente salto del paquete.
- **Retardo de cola:** Tiempo que el paquete espera en el buffer de salida del router hasta ser transmitido. Si no queda memoria en el buffer para almacenar el paquete, este se pierde.

La latencia depende de la distancia, medio de transmisión, protocolos de transmisión y de la cantidad de dispositivos intermedios que existen entre emisor y receptor.

3.1.3.3. Variación del retardo

El parámetro más destacable y que representa a los demás, tratando de capturar todas las características de la categoría es el “jitter”. En esta categoría los parámetros seleccionados analizan la variabilidad de los retardos analizados en el receptor. En primer lugar, en las variaciones de retardo de paquetes IP (IPDV) se analiza en un punto a otro punto de medición y se establece como la diferencia entre el retardo en un solo sentido de los paquetes seleccionados para las pruebas. De la misma manera, el parámetro de la cuantificación de fluctuación/variación del retardo IP a corto plazo se usa para calcular el IPDV, el cual analiza los paquetes en un intervalo de tiempo corto tomando solo el retardo máximo y el retardo mínimo para tomar la decisión, por lo tanto, se pueden descartar estos parámetros con el fin de analizar el “jitter”.

3.1.3.3.1. “Jitter”

El “jitter” es un componente crucial en el retardo de extremo a extremo, debido a que el tiempo que tarda un paquete en ir desde la fuente hasta llegar al receptor puede variar de paquete en paquete.

Se debe tener en cuenta éste valor a la hora de evaluar la calidad de servicio ya que si el receptor ignora el “jitter” y va reproduciendo los paquetes tan pronto como llegan entonces el audio puede llegar a ser ininteligible para el usuario y el video puede tener pausas o aceleraciones. Para eliminar este efecto, a menudo se utilizan los números de secuencia, las marcas de tiempo que están en las cabeceras de los paquetes, un retardo de reproducción o la utilización de un Jitter Buffer.

3.1.3.4. Pérdida de información

En esta categoría se encuentran todos los parámetros relacionados con pérdida de información, lo cual puede ocurrir durante la trasmisión en la red, debido a que los datos pasan por el buffer de los enrutadores (colas de paquetes) y en ocasiones puede suceder que uno o más de los buffers, en el recorrido del paquete al receptor, estén llenos, en cuyo caso el paquete que llega puede ser desechado, de tal forma que no llega a su destino. La solución para este efecto es el reenvío de paquetes a través de TCP, pero en el caso de servicios o aplicaciones de voz, video o aplicaciones en tiempo real, esto es inaceptable ya que aumentaría la latencia, es por esta razón que para este tipo de aplicaciones se usa UDP.

El parámetro que permite representar toda la categoría y que podría ser medido, según la información que llega al receptor, es la perdida de paquetes. Se desechan los demás parámetros, ya que en primera instancia la perdida de fotogramas de video es difícil de medir en la red ya que un fotograma de video se puede distribuir en varios paquetes, de tal forma que es más complejo calcular cuando un fotograma completo se pierde durante el establecimiento de una llamada. Por otra parte, la densidad de ráfaga es un valor que se puede calcular a partir de los paquetes que se pierden o son eliminados por no llegar en el periodo establecido. De la misma forma, el IPER se refiere a paquetes que se pierden en la recepción porque son considerados paquetes erróneos. En cuanto al parámetro IPLR es una tasa media de los paquetes perdidos en relación con los paquetes que llegan exitosos, por lo que se puede decir que es un parámetro dependiente de la perdida de paquetes. En el mismo orden se encuentra la relación del bloque de pérdida severa de paquetes (IPSLBR). Otra variable que depende de este parámetro es el evento de pérdida de paquetes consecutivos, ya que este sucede cuando hay pérdida de paquetes de forma consecutiva. Por otro lado, SPR que es la tasa de paquetes IP espurios solo indica un valor de paquetes espurios o falsos en un intervalo de tiempo. Así mismo, para identificar un segundo degradado solo se analizan los paquetes perdidos en un segundo y según esto se clasifica si es o no un segundo degradado. Finalmente, la ventana de descarte es el parámetro que se establece al final del camino para cada paquete, diciendo que paquete está en el rango aceptable.

3.1.3.4.1. Pérdida de paquetes

La pérdida de información puede llegar a ser fatal para algunos servicios que necesitan fidelidad en los datos. En las aplicaciones en tiempo real es necesario que los paquetes lleguen en el tiempo establecido para que la calidad del servicio no se vea afectada, pero cuando este parámetro se presenta de forma excesiva o consecutiva, no debe superar un porcentaje para que la calidad de voz y video sea aceptable para el usuario.

Una pérdida de paquetes del 1% puede generar congelamiento en el video y/o pérdida de audio, una pérdida del 2% puede hacer que se degrade la calidad del video, aunque el audio pueda ser aceptable, pero una pérdida de paquetes por encima del 2% es considerado inaceptable [37].

3.2. INFRAESTRUCTURA GENERAL DE LA RED

Para estructurar un modelo que permita medir la QoS de una video llamada en una red IMS virtualizada, es necesario definir e identificar el hardware y software a utilizar, así como la topología general de la red, en donde se ejecutara el servidor y los clientes IMS.

3.2.1. Hardware

A continuación, se describe el hardware a utilizar para el establecimiento del servicio de video llamada sobre una red IMS.

3.2.1.1. Switch HP 5500

Es un switch capa 2 y 3 Gigabit Ethernet con capacidad de multiservicio para grandes redes (empresas o campus universitarios), el cual proporciona una conectividad resistente y segura, utilizando últimas tecnologías de priorización del tráfico para mejorar las aplicaciones en redes convergentes (figura 12).

Figura 12. Switch HP 5500. Fuente: [38]

Características:

- 24 puertos RJ-45 Gigabit Ethernet.
- 4 puertos fijos Gigabit Ethernet SFP.
- 2 puertos SFP+ de 10 Gigabit Ethernet.

- Tasa de transferencia soportada: 10/100 Mb/s.
- Estándares de red: IEEE 802.3, IEEE 802.3ab, IEEE 802.3u.
- Memoria interna: 1024 MB.
- Memoria Flash: 512 MB.

3.2.1.2. Virtual Connect

HP VC Flex-10/10D module (figura 13) es una opción de interconexión que simplifica la conectividad del servidor con redes de datos y almacenamiento, proporcionando un ajuste dinámico al aprovechar al máximo el ancho de banda de la red. Además, disminuye significativamente los costos de infraestructura al aumentar el número de NICs por conexión sin agregar módulos de entrada/salida adicionales y así reducir los enlaces de cableado a la red del centro de datos. HP Virtual Connect FlexFabric conecta servidores y máquinas virtuales a datos y redes de almacenamiento utilizando los protocolos Ethernet, canal de fibra e iSCSI.

Figura 13. HP Virtual Connect. Fuente: [39]

Características:

- 4 NICs individuales con su propio ancho de banda dedicado.
- Velocidades de transferencia de NICs de 100Mb a 10 Gb por conexión.
- Una interfaz interna al módulo Onboard Administrator de BladeSystem clase C.

3.2.1.3. Blade BL460 Gen 9

Es un servidor diseñado para una amplia gama de opciones de configuración y despliegue, el cual proporciona flexibilidad con un tamaño adecuado de almacenamiento para diferentes volúmenes de actividades. Este equipo permite fortalecer los componentes y servidores físicos, manteniendo la misma capacidad de rendimiento y carga de trabajo (figura 14).

Figura 14. Servidor Blade BL460c Gen 9. Fuente: [40]

Características:

- 2 procesadores Intel® Xeon® E5-2600 v3.
- 12 núcleos en cada procesador.
- Memoria de 2 TB con DIMM de 128 GB.
- 16 Ranuras de memoria DIMM.

3.2.1.4. MSA 2040

Las matrices de almacenamiento HPE MSA 2040 mejoran las necesidades de protección de datos y almacenamiento compartido de los clientes, incrementando el rendimiento y la disponibilidad al utilizar tecnologías como unidades de estado sólido (SSDs) y unidades de autocifrado (SEDs). Este rendimiento ayuda a los clientes que requieren dar soporte a redes consolidadas y virtuales (figura 15).

Figura 15. MSA 2040. Fuente: [41]

Características:

- 6 Discos, cada uno de 600 Gb.

- Capacidad de máximo 768 TB incluyendo expansión, dependiendo del modelo.
- Controlador SAN de 4 puertos: Canales de fibra de 8Gb/16Gb y/o iSCSI de 1GbE/10GbE.
- Controlador SAS de 4 puertos: SAS de 6Gb/12Gb (conmutación automática).

3.2.1.5. Host

En los equipos portátiles se ejecutan los clientes IMS. Estos no tienen unas características específicas, ya que sólo requieren del sistema operativo Ubuntu 14.04 en donde se instala un cliente IMS Fokus Monster. En la tabla 5 se especifican las características de los equipos disponibles para realizar las pruebas de video llamada.

CARACTERISTICAS	HOST 1	HOST 2
Marca	ASUS-X550L	DELL-Inspiron
RAM	8 GB	6 GB
Procesador	Intel Core i5	Intel Core i5
Sistema operativo	Ubuntu 14.04 64 bits	Ubuntu 14.04 64 bits

Tabla 5. Características de los hosts

3.2.2. Software

En esta sección se describen las características del software a utilizar.

3.2.2.1. VMWare ESXi

VMWare ESXi es un hipervisor del tipo Bare-metal (tipo 1), el cual se instala directamente sobre el hardware sin necesidad de un sistema operativo host como Windows o Linux, permitiendo desplegar y ejecutar varias máquinas virtuales sobre el mismo equipo físico (figura 16).

Figura 16. Sistema básico de gestión. Fuente: [42]

VMWare ofrece seguridad mejorada, mayor confiabilidad y administración simplificada del software hipervisor, en donde la administración se puede realizar directamente desde su consola o por medio de VMware vSphere Client, el cual es una aplicación de escritorio para sistemas operativos Windows que permite conectar, administrar y supervisar simultáneamente todas las máquinas virtuales de manera remota [43].

3.2.2.2. Ubuntu 14.04 LTS

Ubuntu es una distribución de Linux que brinda un sistema operativo enfocado a equipos de escritorio. La versión escogida es la 14.04, ya que es compatible con la versión del Open IMS Core utilizado. Esta instalación sobre versiones anteriores de Ubuntu requiere del despliegue de más paquetes que permitan soportar las características del core IMS.

Los requisitos mínimos [44]:

- Procesador Intel x86 a 700 MHz.
- Mínimo 512 Mb de memoria RAM.
- Disco Duro de 5 GB (swap incluida).
- Tarjeta gráfica y monitor capaz de soportar una resolución de 1024x768
- Unidad de DVD o puerto USB para realizar la instalación.
- Conexión a Internet.

3.2.2.3. Open IMS Core

Open IMS Core es una implementación Open Source desarrollada por el Instituto Fraunhofer de Sistemas de Comunicación Abierta (FOKUS) en Berlin, Alemania con el fin de promover la adopción de la tecnología IMS en las redes de telecomunicaciones de nueva generación y capacitar a un grupo de desarrolladores para crear nuevos servicios basados en IMS. Desde entonces, Open IMS Core sirve como una implementación de referencia para realizar pruebas y prototipos de la tecnología IMS [45].

Open IMS Core cuenta con las siguientes características:

- Flexibilidad y rendimiento robusto.
- Módulo de interconexión PSTN/PLMN.
- Accesibilidad a través de nombres DNS y direcciones de IP públicas.
- Funciones de control de servicio de llamadas basadas en SER (SIP Express Router).
- El bajo recurso de hardware permite realizar simulaciones extensas y realistas a un costo mucho menor que los ambientes de prueba de telecomunicaciones tradicionales.

3.2.2.4. Fokus Monster Client

Fokus Monster es un cliente IMS de Fraunhofer que proporciona una API de alto nivel para acceder a los servicios de IMS. Esta API oculta los detalles de la tecnología IMS y expone el soporte del nivel de servicio para permitir el fácil desarrollo de las aplicaciones IMS. Además, cuenta con una serie de servicios como chat, llamada, video llamada, presencia, llamada en espera, entre otros.

En el cliente Fokus se configura la información de la red IMS, ingresando los datos del cliente registrado en el core, como usuario, contraseña, dominio, puerto, etc. Además, permite seleccionar el códec tanto de video como de audio a utilizar para el establecimiento de la video llamada.

3.2.2.5. Wireshark

Wireshark es un analizador de protocolos de red de código abierto disponible para plataformas Windows y Unix, el cual permite examinar el tráfico a través de la interfaz de red ya sea cableada o inalámbrica. Gracias a wireshark se pueden filtrar los más de 1100 protocolos que soporta actualmente, por medio de una interfaz de usuario sencilla e intuitiva, que permite extraer y visualizar los datos de cada uno de los paquetes monitorizados. De esta manera wireshark proporciona una gran cantidad de posibilidades para el análisis del tráfico al administrador de redes [46].

Algunas de las características de este analizador de paquetes de red, se describen a continuación:

- Captura en tiempo real los paquetes desde una interfaz de red.
- Guarda la información del paquete capturado.
- Permite abrir los archivos almacenados en el equipo desde otros analizadores de tráfico como tcpdump, WinDump, entre otros.
- Despliega de manera detallada la información del protocolo capturado.
- Cuenta con una variedad de posibilidades para filtrar y buscar los paquetes de red.
- Crea varias estadísticas a partir de los datos obtenidos.
- Genera graficas en función del tiempo de las variables del protocolo analizado.

3.2.2.6. Wondershaper

Wondershaper es un sencillo controlador de tráfico para Linux, el cual permite limitar el ancho de banda tanto para el enlace de subida como de bajada de una interfaz de red. Esta herramienta permitirá limitar el ancho de banda de los hosts, con el fin de realizar un banco de pruebas para analizar el rendimiento y la calidad del servicio de una video llamada.

3.2.3. Topología general de la red

La plataforma Telco 2.0 de la Universidad del Cauca cuenta con una infraestructura de red, en donde será instalado el servidor (Core IMS) que permitirá establecer la comunicación del servicio de video llamada entre los usuarios finales.

La topología que se observa en la figura 17, es parte de la arquitectura con la que cuenta la plataforma, la cual será utilizada al momento de implementar el servidor IMS y establecer el servicio de video llamada.

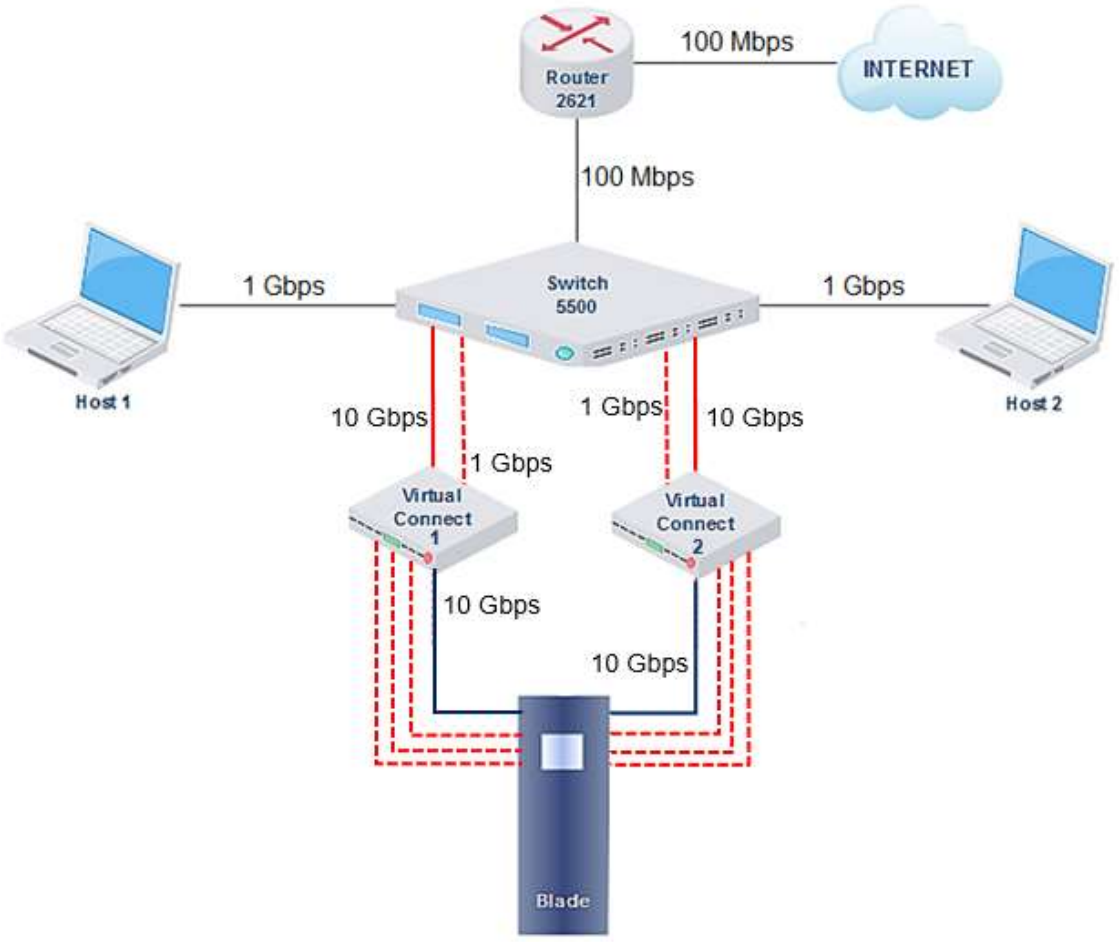

Figura 17. Topología general de la red. Fuente: Propia

El Router y el switch son dispositivos que permiten direccionar los datos dentro de la red, los cuales están conectados por medio de un enlace Fast Ethernet de 100 Mbps. El switch 5500 se conecta con los virtual connect, utilizando un enlace Gigabit Ethernet de 10 Gbps y un cable RJ45 de 1 Gbps para redundancia. Los virtual connect se enlazan con el Blade por medio de conexiones Gigabit Ethernet de 10 Gbps y habilitan enlaces de red virtuales para redundancia.

El Blade, dispone de una distribución de VMWare que le permite desplegar máquinas virtuales a los usuarios dependiendo de las características del servicio que se desea brindar. Dentro de una de estas máquinas virtuales se encuentra instalado el Core IMS, el cual es el servidor del servicio estudiado. En la figura 18 se observa el diagrama jerárquico de implementación de dicha distribución dentro del Blade.

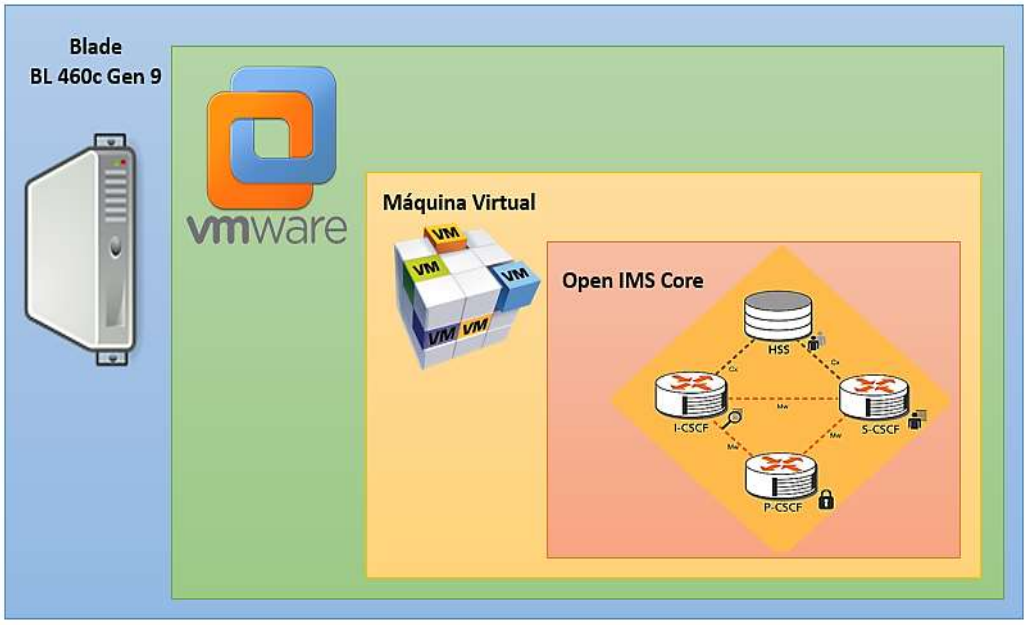

Figura 18. Diagrama jerárquico de implementación del Core IMS. Fuente: Propia

En cuanto al equipo terminal (host) tiene como sistema operativo una de las distribuciones de Linux, en este caso Ubuntu 14.04, ya que es compatible con la versión del Open IMS Core utilizado. Es aquí donde se instala la aplicación de escritorio Fokus Monster Client, como se observa en la figura 19.

Figura 19. Diagrama jerárquico de implementación del cliente IMS. Fuente: Propia

3.3. CARACTERIZACIÓN DE LA RED

La caracterización de la red consiste en definir las variaciones que van a aplicarse sobre su dimensionamiento o topología, de manera que aplicando el modelo QoS definido se pueda establecer los efectos sobre la calidad en el servicio de video llamada. La red física de la figura 17, garantiza una calidad de servicio fija ya que esta depende, además del tráfico y de la congestión de la red, de las características y especificaciones de los equipos utilizados al implementar el servicio. Por esta razón es necesario establecer una red virtual que permita modificar las características y las condiciones de la red, con lo cual se evidencien variaciones en cuanto a la QoS del servicio.

Para identificar la QoS del servicio de video llamada sobre la red IMS virtualizada, es necesario realizar un análisis de la escalabilidad de la red, ya que es un factor que permite identificar el rendimiento y estudiar el comportamiento de la red al aumentar el número de usuarios que utilizan, en este caso, el servicio de video llamada. De esta manera se genera un patrón que permita identificar hasta qué punto se garantiza una QoS de acuerdo a las características de la red virtual.

Se distinguen dos tipos de escalabilidad, vertical y horizontal, pero no existe una regla general que permita identificar qué tipo sobresale más que el otro, ya que cada uno tiene sus costos y sus beneficios, es por ello que se analiza cada uno de estos tipos de escalabilidad, en donde se define la topología y las características de la red que serán propósito de estudio.

3.3.1. Escalabilidad Vertical

La escalabilidad vertical consiste en añadirle más recursos a un nodo en particular, es decir, aumentar la memoria del disco y el número de núcleos del procesador, aumentando así la capacidad de procesamiento de información. Lo ideal sería que, al incrementar los recursos del sistema, también aumente la capacidad del servicio prestado, pero el sistema va a llegar a un punto de saturación en el que por más que se aumenten los recursos, la capacidad del servicio siempre será la misma (figura 20).

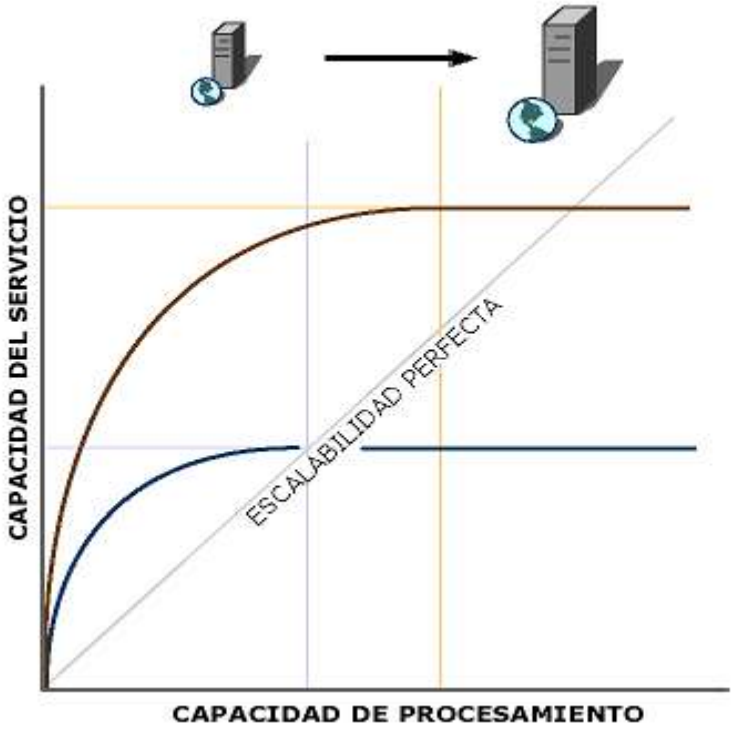

Figura 20. Escalabilidad Vertical. Fuente: Propia

Para esta investigación, lo que se busca es aumentar la capacidad de la máquina virtual en donde estará el servidor (Core IMS) y de esta manera analizar las características de calidad del servicio para encontrar el punto de saturación del sistema en general. En la figura 21, se observa la topología de la red, al utilizar la escalabilidad vertical.

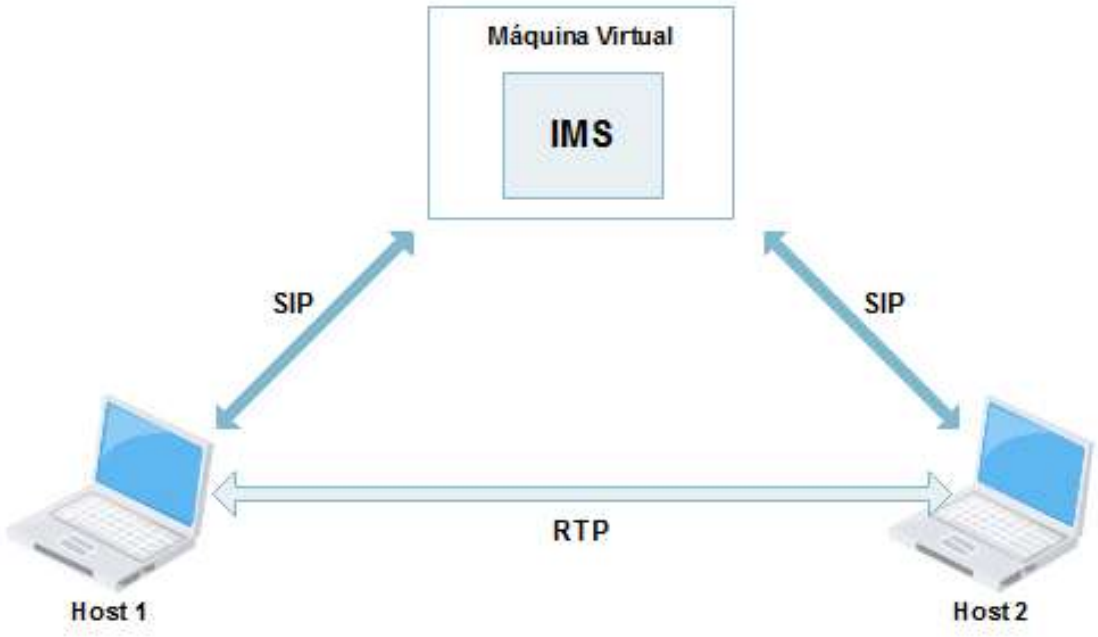

Figura 21. Topología Escalamiento Vertical. Fuente: Propia

3.3.2. Escalabilidad Horizontal

La escalabilidad horizontal consiste en distribuir la carga en más de un servidor, para ello es necesario utilizar un balanceador de carga que divida el tráfico o la cantidad de usuarios entre los servidores. Al igual que en la escalabilidad vertical, lo ideal sería que al aumentar la cantidad de servidores, aumente la capacidad del servicio, pero va a llegar a un punto en que la capacidad del servicio va a permanecer constante por más de que se aumenten servidores en la red, como se observa en la figura 22.

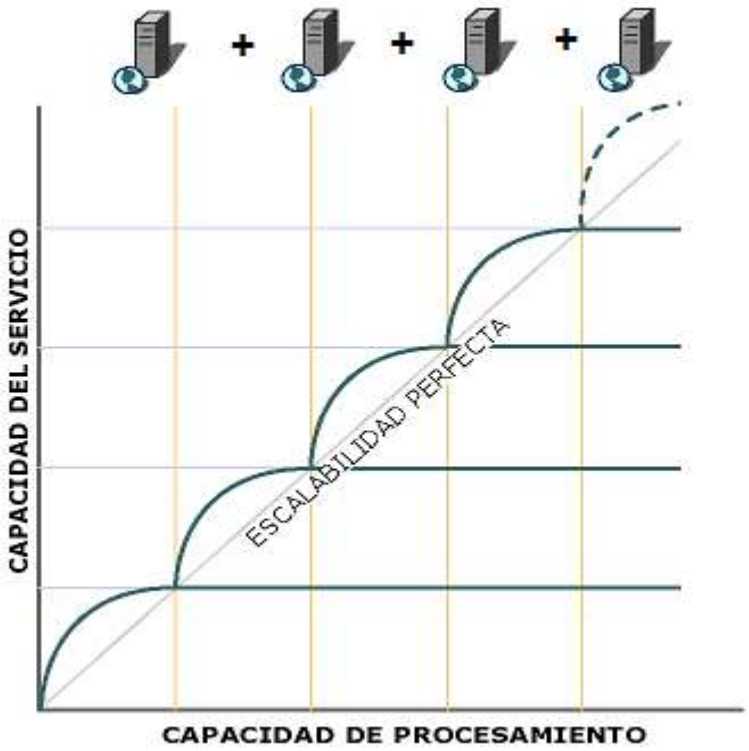

Figura 22. Escalabilidad Horizontal. Fuente: Propia

En este caso, se busca incrementar el número de servidores IMS, con el fin de aumentar la cantidad de usuarios en la red para analizar las características de calidad del servicio y encontrar el punto de saturación en el que la capacidad del servicio se mantiene constante. En la figura 23 se observa la topología de la red al utilizar una escalabilidad horizontal.

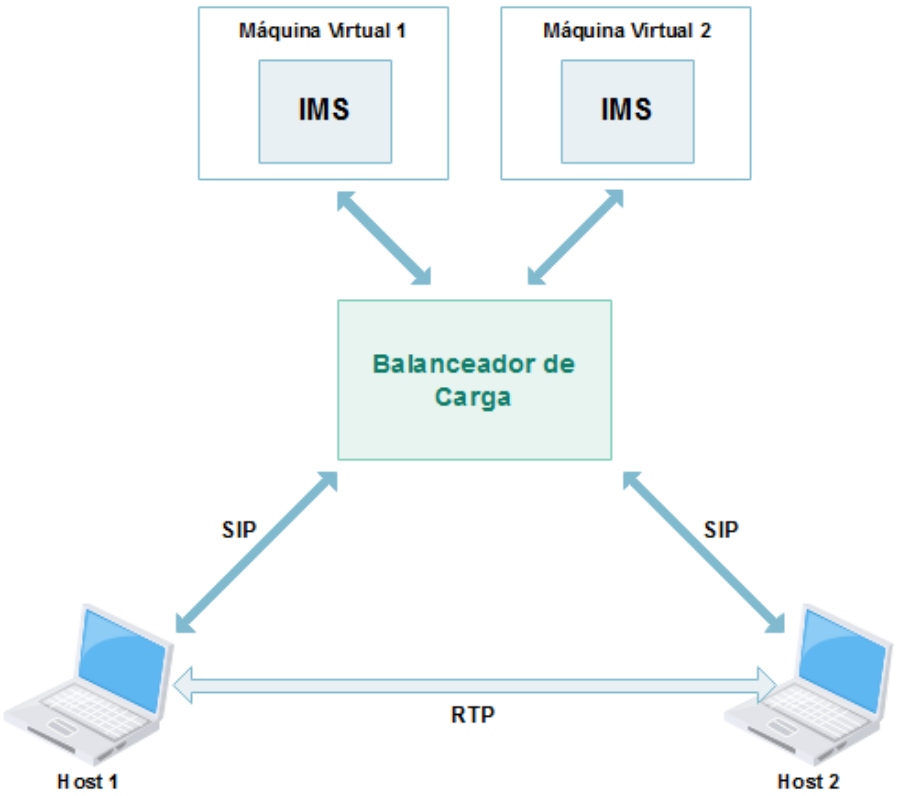

Figura 23. Topología Escalamiento Horizontal. Fuente: Propia

El modelo propuesto no se basa en una metodología secuencial para medir la QoS de una video llamada, si no que busca analizar la QoS teniendo en cuenta tanto los parámetros de calidad como la distribución del tráfico en la red y la variación de las características de las máquinas virtuales, con el fin de examinar los resultados obtenidos y así establecer una aproximación del rendimiento y la capacidad de la red al comparar los dos tipos de escalabilidad implementados.

3.4. CARACTERIZACIÓN DEL SERVICIO DE VIDEO LLAMADA

Después de caracterizar la red en general, es indispensable definir las propiedades que tiene el servicio de video llamada, ya que este integra un servicio de telecomunicaciones típico como lo es la transmisión de voz con un servicio multimedia web (video). Por medio de diagramas de secuencia, se ilustran los mensajes SIP intercambiados entre IMS y los clientes, incluyendo los mensajes que se envían entre si los componentes de IMS y los mensajes RTP que tratan los clientes al establecer la comunicación. Esto con el fin de comprender de manera gráfica, el orden de envío de los paquetes y la secuencia en que los componentes involucrados reciben y envían dichos mensajes al establecer una video llamada entre dos usuarios.

3.4.1. Propiedades de una video llamada

El servicio de video llamada, según la UIT se define como un “servicio audiovisual conversacional que permite la transferencia bidireccional, simétrica y en tiempo real, de sonido y video con movimiento entre dos lugares (de persona a persona) a través de redes” [47]. Como se dijo anteriormente, el servicio de video llamada integra dos servicios de gran demanda, voz y video. Razón por la cual es necesario tener en cuenta las propiedades de cada uno de ellos.

3.4.1.1. Propiedades de audio

Las señales de audio analógicas (voz) son transformadas en datos digitales, a través del muestreo, cuantificación y codificación de la señal, que se transmiten sobre la red de internet hacia una red IP determinada. La señal original de audio es recuperada en el receptor al decodificar la señal. En este proceso las señales de alta frecuencia pueden perderse, para evitar esto se aumenta la frecuencia de muestreo y la señal decodificada puede tener una aproximación más fiel a la señal analógica original.

El audio tiene una demanda de ancho de banda significativamente más baja que la del video, por lo que sus propiedades deben ser consideradas al implementar un servicio multimedia. Según el Códec de audio utilizado en la transmisión, se utilizará más o menos ancho de banda. El ancho de banda utilizado suele ser directamente proporcional a la calidad de los datos transmitidos.

Es este caso el códec de audio utilizado por el cliente IMS para el establecimiento de la video llamada es el códec G.711, el cual proporciona un flujo de datos de 56 o 64 Kbit/s y representa la voz humana mediante palabras de 8 bits con una tasa de 8000 muestras por segundo.

3.4.1.2. Propiedades de video

Un video es una secuencia de imágenes que se muestran a una velocidad constante, de tal forma que la percepción humana no detecte estos cambios. Para transmitir una señal de video por la red IP se debe transformar la señal analógica en digital. Al igual que el audio, la señal de video se muestrea, cuantifica, codifica y se transmite hacia el usuario final, en donde se realiza el proceso inverso para decodificar la señal digital en analógica.

Transmitir una señal de video en su formato original requiere de un gran ancho de banda, por lo que es necesario comprimir la señal antes de transmitirla. Al comprimirla se logra representar las muestras con la menor cantidad de bits sin perder información, con el fin de aumentar la velocidad de transmisión. El vídeo en una aplicación de video llamada se puede comprimir en tiempo real para

proporcionar la mejor calidad de vídeo según el ancho de banda disponible de extremo a extremo entre usuarios, dependiendo del códec utilizado.

El códec de video utilizado por el cliente IMS es el códec H.263, el cual soporta un conjunto limitado de tamaños de imagen y especifica un algoritmo de codificación y decodificación de señales de video en tiempo real. El propio estándar no especifica una tasa de bits concreta, pero utiliza velocidades bajas para la transmisión.

3.4.2. Secuencia de mensajes

Para establecer una comunicación entre dos usuarios, es necesario conocer en detalle los mensajes intercambiados entre IMS y los clientes, incluyendo los paquetes que se envían entre si los componentes de IMS y los mensajes RTP que los clientes se envían entre si al establecer la comunicación. Para ello, se utilizan los diagramas de secuencia que permiten visualizar la serie de mensajes intercambiados.

Cuando el cliente desea conectarse con el core IMS, envía una secuencia de mensajes SIP antes de establecer la comunicación entre los clientes, los cuales se explicarán con más detalle a continuación:

- Register

Este es el primer mensaje que el cliente envía al core IMS. El cliente intenta registrarse enviando una solicitud “REGISTER” al P-CSCF, el cual repite el mismo mensaje al I-CSCF. El I-CSCF se comunica directamente con la base de datos (HSS), en donde se decide a que S-CSCF debe ser enviada la petición. Después de esto, el I-CSCF envía el paquete al S-CSCF correspondiente. Este autentica los datos con el HSS y envía un mensaje de respuesta “401 UNAUTHORIZED” junto con unos valores de autenticación y el P-CSCF envía el mismo mensaje al cliente IMS. Aquí tanto el cliente como la red han almacenado algunos datos de autenticación del usuario encriptados utilizando el algoritmo MD5.

El cliente envía nuevamente el mensaje “REGISTER” al P-CSCF junto con los datos de autenticación encriptados. El P-CSCF envía esta petición al I-CSCF y este lo remite al S-CSCF. El S-CSCF compara estos datos con la información almacenada en la base de datos y determina si se debe permitir el registro del usuario. Si estos valores coinciden, el S-CSCF envía una respuesta “200 OK” al P-CSCF, el cual envía nuevamente la respuesta al cliente. Este proceso se observa en el diagrama de secuencia de la figura 24.

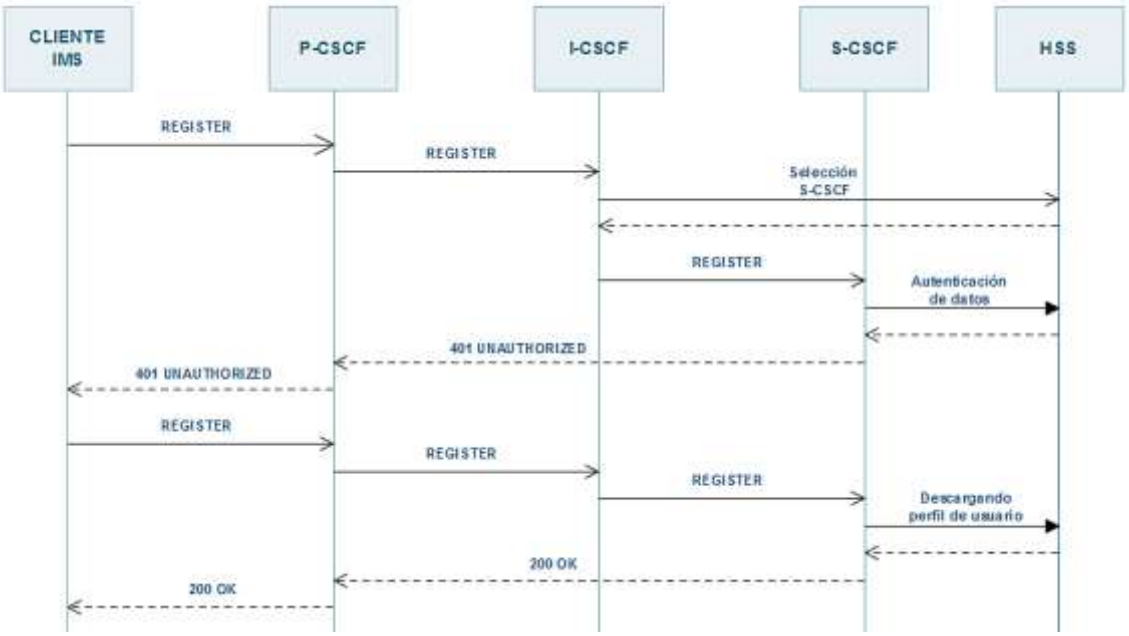

Figura 24. Diagrama de secuencia del mensaje Register. Fuente: Propia

- Subscribe

Al terminar las peticiones de registro, el cliente envía un mensaje "SUBSCRIBE" para obtener las actualizaciones de presencia (disponibilidad) de otros usuarios, cuando estos cambian la información de registro. Si esto ocurre, el cliente recibe las actualizaciones con el mensaje "NOTIFY" sobre un usuario determinado, de lo contrario el usuario recibe un "200 Subscription to REG saved" (figura 25).

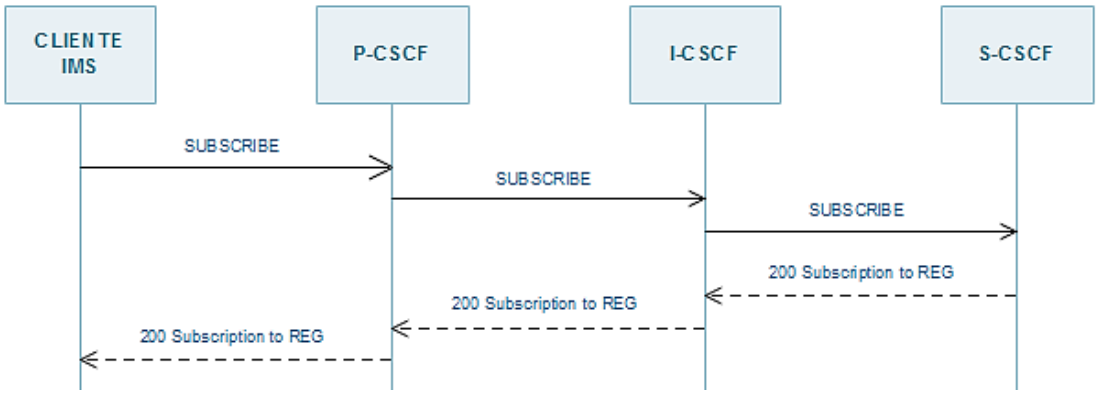

Figura 25. Diagrama de secuencia del mensaje Subscribe. Fuente: Propia

Este mismo procedimiento, lo debe realizar el otro cliente con el que se desea establecer comunicación. Este debe registrarse en el core IMS y obtener las actualizaciones de la información de registro del otro cliente sobre la disponibilidad del mismo.

Una vez registrados los dos usuarios, uno de ellos envía la petición para el establecimiento de la video llamada, como se describe a continuación.

- Invite

El mensaje SIP enviado para el establecimiento de la video llamada es “INVITE”, este paquete es remitido por el solicitante de la comunicación (cliente 1) hacia el core IMS, quien recibe, analiza los datos del usuario y envía la petición hacia el destinatario (cliente 2). Los mensajes “100 Trying” son respuestas provisionales para hacer saber al nodo anterior que las peticiones han llegado correctamente. El receptor, envía un mensaje “180 Ringing” para indicar que el mensaje ha sido recibido. Si el destinatario acepta la video llamada, este envía un “200 OK” que indica que el proceso correspondiente a la solicitud “INVITE” ha finalizado, es decir, se completa el establecimiento de la comunicación. Finalmente, el cliente 1 envía un “ACK”, el cual muestra el éxito de la configuración de la ruta y se confirma con un “200 OK”, para que la transmisión multimedia pueda comenzar entre los usuarios. En la figura 26 se observa el diagrama de secuencia de los mensajes al establecer una comunicación.

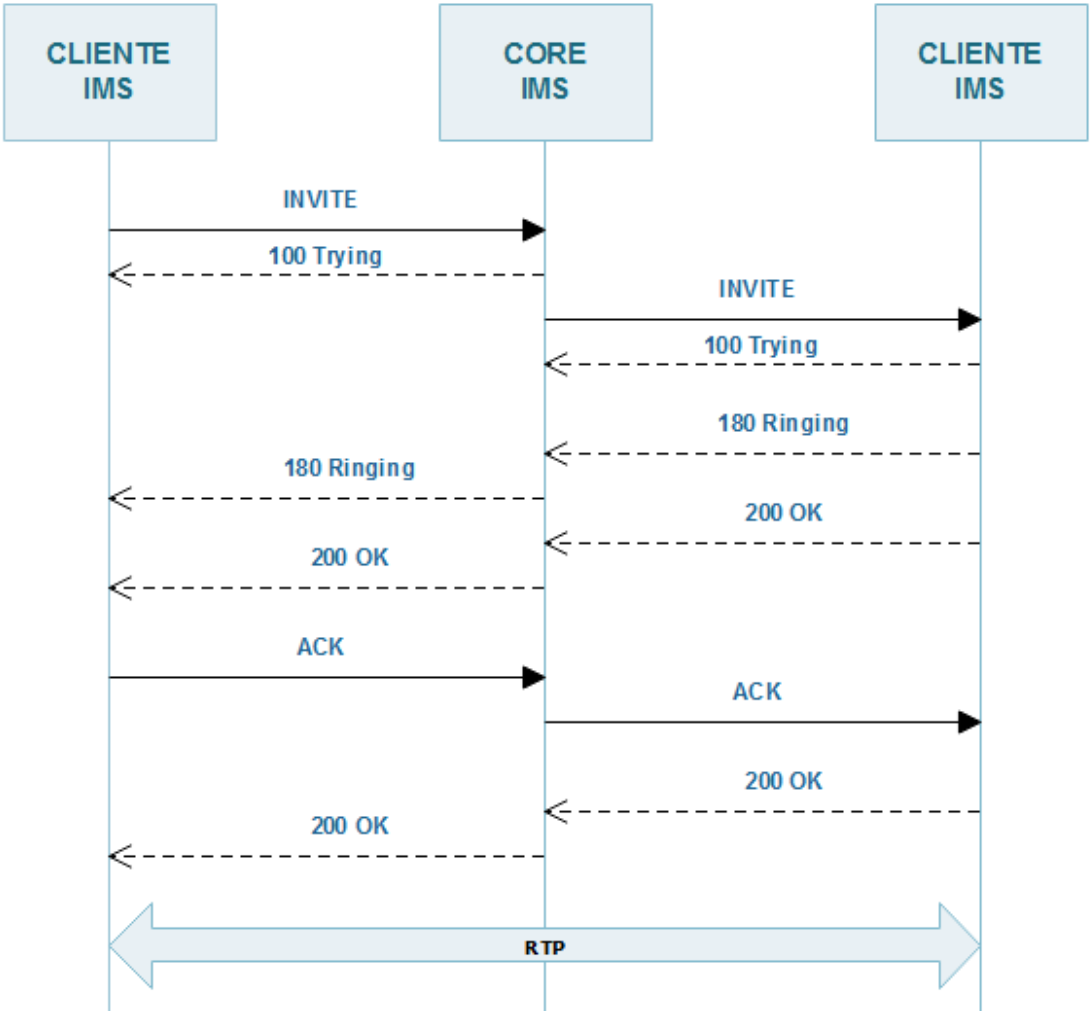

Figura 26. Diagrama de secuencia del mensaje Invite. Fuente: Propia

Al establecer la comunicación, los mensajes “RTP” se envían únicamente entre los usuarios y el core IMS solo supervisa y está pendiente de la petición de

alguno de los usuarios para finalizar el establecimiento de la video llamada. Dentro de los mensajes “RTP” se envían las señales digitales tanto de audio como de video en el payload del paquete, utilizando los estándares para la codificación de la ITU-T G.711 y H.263 respectivamente (figura 27).

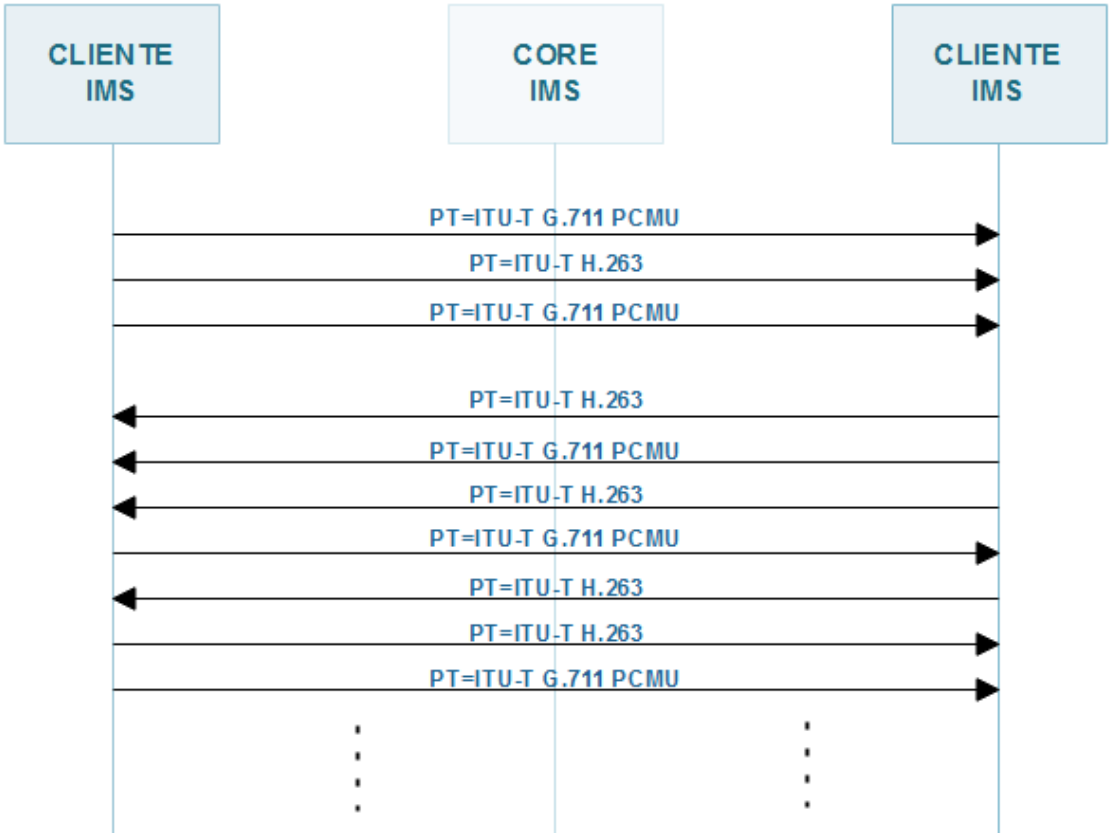

Figura 27. Diagrama de secuencia de mensajes RTP. Fuente: Propia

- Bye

Si alguno de los usuarios desea terminar la sesión, este envía un mensaje “BYE” al core IMS, quien reenvía la petición hacia el otro usuario. Se envía un mensaje “200 OK” para confirmar la finalización de la video llamada (figura 28).

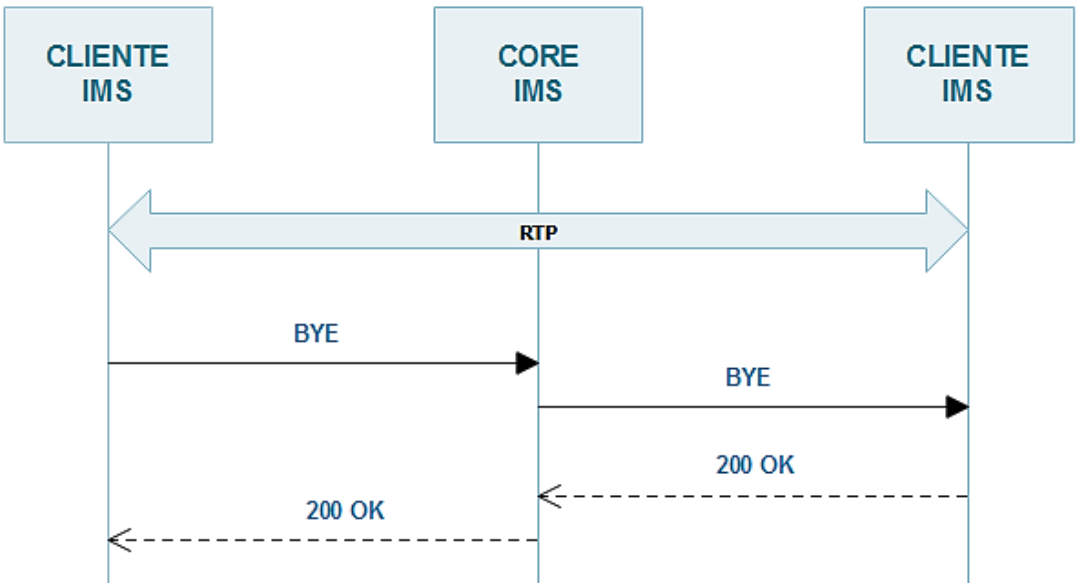

Figura 28. Diagrama de secuencia del mensaje Bye. Fuente: Propia

De la misma manera que se realiza el proceso de registro, se anula el registro con el Core IMS, como se observa en la figura 29. El cliente elimina el registro al cerrar la aplicación, enviando una solicitud “UNREGISTER” al P-CSCF, el cual repite el mismo mensaje al I-CSCF. Después, el I-CSCF envía el paquete al S-CSCF correspondiente. Este autentica los datos con el HSS y envía un mensaje de respuesta “401 UNAUTHORIZED” junto con unos valores de autenticación y el P-CSCF envía el mismo mensaje al cliente IMS.

El cliente envía nuevamente el mensaje “UNREGISTER” al P-CSCF junto con los datos de autenticación encriptados. El P-CSCF envía esta petición al I-CSCF y este lo remite al S-CSCF. El S-CSCF compara estos datos con la información almacenada en la base de datos y admite anular el registro del usuario. Si estos valores coinciden, el S-CSCF envía una respuesta “200 OK” al P-CSCF, el cual envía nuevamente la respuesta al cliente.

De esta manera se elimina el registro del usuario quedando libre el enlace para establecer otra comunicación entre los usuarios registrados en el Core IMS.

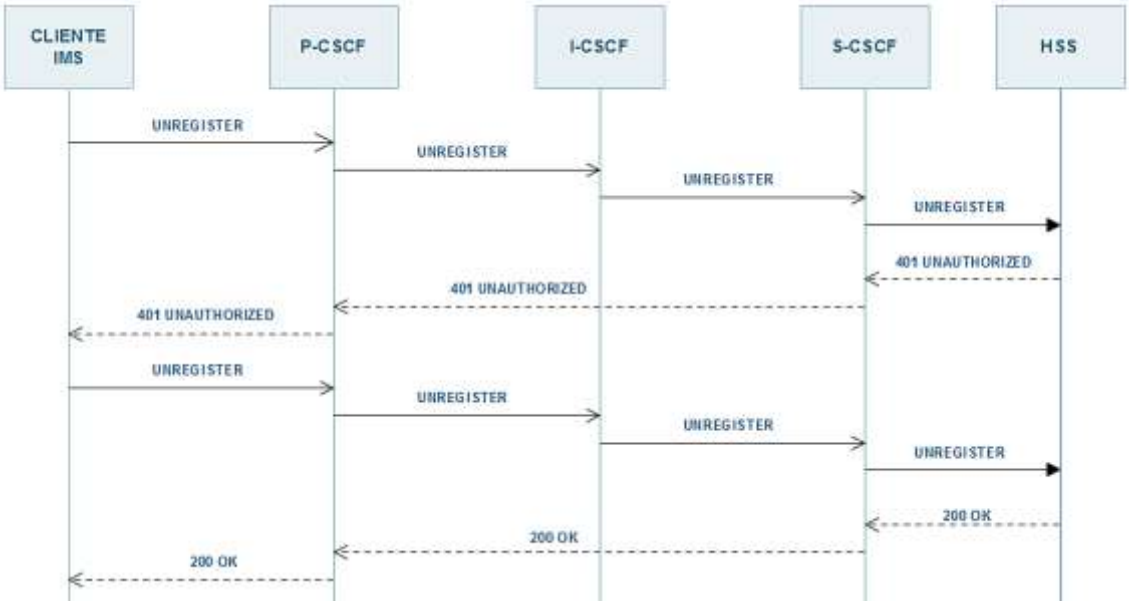

Figura 29. Diagrama de secuencia para anular el registro en el Core IMS. Fuente: Propia

4. PLAN DE PRUEBAS Y ANALISIS DE RESULTADOS

En este capítulo se describe el plan de pruebas realizado en la red con la arquitectura, infraestructura y componentes descritos en el anterior capitulo, con el fin de obtener resultados que se puedan interpretar para determinar los valores adecuados de los parámetros seleccionados en el modelo definido y por ende determinar la QoS del servicio de video llamada en diferentes condiciones y ambientes. Según los casos de estudio definidos en el plan de pruebas, estos valores se pueden clasificar, ordenar y graficar para tener una vista más amplia de los resultados que se están obteniendo, haciendo énfasis en los parámetros elegidos para el modelo.

El objetivo de elaborar un plan de pruebas es estructurar los diferentes tipos de escenarios y casos de estudio de una forma eficiente, entendible y que aborde los aspectos, parámetros y características que componen la red obteniendo resultados claros de los parámetros y de esta forma cumplir con el tercer objetivo específico planteado en el presente trabajo de grado el cual es validar el modelo propuesto. Además, por medio del plan de pruebas se pueden encontrar falencias en la ejecución de la video llamada o fallas en los componentes de la red, ya sea código de los módulos o configuraciones de hardware o software, lo que permite buscar las soluciones a posibles errores encontrados.

4.1. ESTRUCTURA DE LAS PRUEBAS

Con el fin de establecer un orden a la ejecución de las pruebas, se define una estructura en donde se abarcan escenarios favorables y desfavorables para la video llamada, obteniendo valores o rangos de valores que permiten determinar la calidad de la comunicación según los parámetros escogidos para el modelo. Estos resultados pueden ser usados como condiciones iniciales en posteriores investigaciones, reduciendo considerablemente el número de pruebas que podrían ser ejecutadas.

En primera instancia se examina la escalabilidad vertical ya que por medio de este estudio se puede determinar el comportamiento de Open IMS Core al aumentar o disminuir los recursos hardware de la máquina virtual en donde se encuentra instalado. Por otro lado, se estudia la escalabilidad horizontal usando las características de las máquinas virtuales según los resultados obtenidos en la sección de pruebas de la escalabilidad vertical, con el fin seleccionar el tipo de escalabilidad que sea más eficiente. Por último, se analizan los parámetros de calidad como retardo, perdida de paquetes y “jitter”, al variar el ancho de banda utilizado por los usuarios, ya que es importante escoger el valor óptimo para que una video llamada funcione correctamente y no desperdiciar recursos de la red.

4.1.1. Escalabilidad Vertical

Para estudiar la escalabilidad vertical del Core se instanciaron 4 máquinas virtuales en el hipervisor VMWare ESXi, cada una de ellas con características hardware diferentes empezando por una máquina virtual básica, de tal forma que por medio de diferentes casos de estudio se pueda determinar si es viable la escalabilidad vertical para Open Core IMS.

4.1.2. Escalabilidad Horizontal

Para poner a prueba la escalabilidad horizontal se instancian varias máquinas virtuales con las mismas características físicas, cada una de ellas con el Core IMS instalado. Además, cuenta con un balanceador de carga que distribuye las peticiones entre los diferentes Core IMS asociados al balanceador. Se realizan pruebas de estrés similares a las pruebas hechas en la sección de escalabilidad vertical, con el fin de determinar en qué tipo de escalabilidad se obtienen mejores resultados.

4.1.3. Parámetros de Calidad

En esta sección de pruebas se analizan los parámetros de calidad seleccionados para el modelo de QoS, para ello se varía el ancho de banda de dos usuarios que establecen una llamada entre ellos, haciendo uso de una máquina virtual que tenga Open IMS Core instanciado. Las variaciones de ancho de banda permiten analizar los parámetros de calidad de la comunicación entre los usuarios, más específicamente los parámetros relacionados con el flujo de paquetes RTP que existe entre ellos. Además, se simula la conexión simultánea de una cantidad considerable de usuarios en la red, los cuales generan tráfico en el servidor hasta que la llamada finalice con el objetivo de retrasar el tiempo de respuesta del servidor.

4.2. DESARROLLO DE LAS PRUEBAS

A continuación, se explica detalladamente el desarrollo de las pruebas planteadas anteriormente, exponiendo de forma clara todos los procedimientos realizados en cada uno de los casos de estudio para cada prueba.

4.2.1. Escalabilidad Vertical

El objetivo de analizar la escalabilidad vertical es determinar el comportamiento de Open IMS Core al aumentar o disminuir los recursos hardware de la máquina virtual en donde se encuentra instalado. Para ello se establecieron 4 casos de estudio en donde solo se varían las características físicas que el hipervisor permite modificar, como la memoria RAM y el número de procesadores asignados.

En la documentación oficial que se encuentra en la página web de Open IMS Core [48], no se especifican valores exactos acerca de las características físicas necesarias para el correcto funcionamiento del Core, solo recomiendan disponer de varios Gigabytes de RAM y de varios CPUs/Cores. En este caso se tomaron unos rangos de valores que permiten analizar el funcionamiento de las máquinas virtuales en el peor y en el mejor de los casos, como se observa en la tabla 6.

Nombre de la máquina virtual	Memoria RAM	Número de procesadores
CoreIMS1	512 MB	1
CoreIMS2	1 GB	1
CoreIMS3	1 GB	2
CoreIMS4	2 GB	2

Tabla 6. Características físicas de las MV

Para analizar las capacidades de Open IMS Core se pone a prueba su tiempo de respuesta a las peticiones entrantes y los paquetes que no son atendidos debido a la saturación del Core. Para ello, es necesario simular la conexión de varios usuarios al Core y analizar dichos parámetros.

4.2.1.1. Componentes de pruebas

Para analizar los tiempos de respuesta del Core IMS, es necesario generar peticiones SIP que simulen la conexión entre el cliente y el servidor IMS. Para ello se realizó una búsqueda en la red de posibles simuladores de mensajes SIP, encontrando algunas opciones como StarTrinity, VoIP Monitor, SIPp, entre otros. Estos softwares, permiten enviar mensajes SIP desde un host transmisor hacia un host receptor, en este caso el transmisor es el simulador y el receptor es el Core IMS. Estos simuladores no funcionaron correctamente ya que no cumplían con los requisitos de compatibilidad de funcionamiento del Core, en ocasiones se perdían los paquetes o el simulador no respondía correctamente los mensajes para que el Core los pudiera interpretar. Debido a estos problemas de compatibilidad presentados por los simuladores y el Core, se optó por desarrollar un simulador de mensajes SIP que fuera compatible con el Core IMS.

El simulador de mensajes SIP fue desarrollado utilizando el lenguaje de programación Python y algunas librerías que permiten enviar y recibir paquetes UDP. El simulador consta de tres módulos, el primer módulo escucha los puertos por donde se desea establecer el enlace y responde los mensajes enviados por el Core. El segundo módulo envía los paquetes SIP según el parámetro lambda y el rango de usuarios registrados. El tercer módulo realiza el cálculo del número de paquetes perdidos y el promedios de los tiempos de respuesta, tanto de conexión como de desconexión de cada paquete en el Core. En el anexo B.3, se explica detalladamente el código utilizado para la implementación del simulador de mensajes SIP.

El segundo componente del simulador envía peticiones de registro al Core, estas peticiones se realizan en tiempos aleatorios e independientes, por lo cual es necesario encontrar el tiempo de concurrencia entre peticiones consecutivas atendidas por el Core IMS. A continuación, se explica cómo se obtienen dichos tiempos.

4.2.1.1.1. Tiempo de concurrencia entre peticiones consecutivas

Los usuarios se registran en el Core IMS en tiempos aleatorios e independientes (proceso estocástico). Según la teoría de colas [49], la concurrencia de llamadas se puede modelar por medio de una distribución de probabilidad Poisson. Esta distribución es de tipo discreta y es muy útil para caracterizar eventos como solicitudes de llamadas, establecimiento de llamadas en una central telefónica, entre otros comportamientos aleatorios. La distribución Poisson expresa la probabilidad de que ocurran un determinado número de eventos en un lapso de tiempo establecido.

Para poder manipular dicha distribución y generar los tiempos de concurrencia de cada petición, se utiliza la herramienta de simulación Matlab. Este software utiliza algunas instrucciones matemáticas que permite determinar aleatoriamente una cantidad considerable de tiempos, dependiendo del número de llamadas que se desean generar (N) y el número promedio de llamadas por segundo (λ). En el anexo B.1, se analiza el código utilizado para generar dichos tiempos de concurrencia entre cada petición.

4.2.1.2. Plan de pruebas

Se realiza una amplia gama de pruebas para determinar los tiempos de respuesta del Core IMS, para ello es necesario conocer los rangos de promedios de llamadas (λ) y números de usuarios. Se analizan los casos de eficiencia y saturación del Core, por tal motivo se estructuran las primeras pruebas como se describe a continuación.

Se combina el número de usuarios con el promedio de llamadas por minuto, en donde se varían los usuarios en intervalos de 50 (50, 100, 150, 200, 250 y 300) y se combinan con valores de lambda tomados como sigue 50, 100, 150, 200, 250, 500 y 1000 llamadas por minuto, para un total de 42 pruebas, cada una de ellas corroborada 3 veces para sacar un promedio de los tiempos de respuesta y promedio de paquetes perdidos.

En cada prueba, se pone a correr el simulador de mensajes SIP, en el cual se establece un número determinado de conexiones, el número de llamadas por minuto (λ), la dirección IP del Core IMS y el nombre con la cual se desea almacenar los valores obtenidos en la base de datos. Además, calcula los tiempos de respuesta de cada mensaje SIP enviado al Core.

El anterior conjunto de pruebas se realizó para las 4 máquinas virtuales mencionadas anteriormente (tabla 6), en donde los resultados obtenidos se observan a continuación.

4.2.1.3. Resultados de las pruebas

Con el objetivo de estudiar el tiempo de respuesta de cada Core instalado en las máquinas virtuales descritas para cada caso de estudio, se utiliza el simulador de mensajes SIP desarrollado, el cual envía y recibe 6 mensajes SIP para el registro y 4 mensajes para la finalización de la sesión de un usuario en el Core IMS. El simulador almacena el tiempo exacto en el que la petición sale del simulador y el tiempo del paquete SIP que envía el Core como respuesta. De esta forma se puede calcular el tiempo utilizado por el Core para procesar y enviar una respuesta a las peticiones del usuario. De la misma forma, se monitorean los paquetes para determinar cuántos paquetes perdidos presenta cada conexión.

A continuación, se analizan los resultados obtenidos en cada caso de estudio de escalabilidad vertical del Core IMS.

4.2.1.3.1. Caso de estudio 1

En este caso de estudio se pone a prueba el Core instalado en la primera máquina virtual, caracterizada por tener una RAM de 512 MB y 1 procesador. Se analiza el tiempo de establecimiento y el tiempo de desconexión al variar el número de usuarios y el promedio de llamadas por segundo (λ), como se observa en la figura 30 y figura 31.

En la figura 30 se observa que a medida que aumenta el número de conexiones en un rango de tiempo determinado, mayor es el tiempo de respuesta del servidor a las peticiones entrantes. La primera línea azul, que corresponde al parámetro lambda de 50 llamadas por minuto, tiene un comportamiento ideal por sus tiempos de respuesta casi inmediatos, este panorama cambia drásticamente cuando se aumenta el parámetro lambda al doble, y estos tiempos aumentan más cuando crece el número de usuarios. De esta forma se observa que a medida que aumentan las conexiones establecidas en un lapso de tiempo, el servidor toma más tiempo en responder las peticiones entrantes.

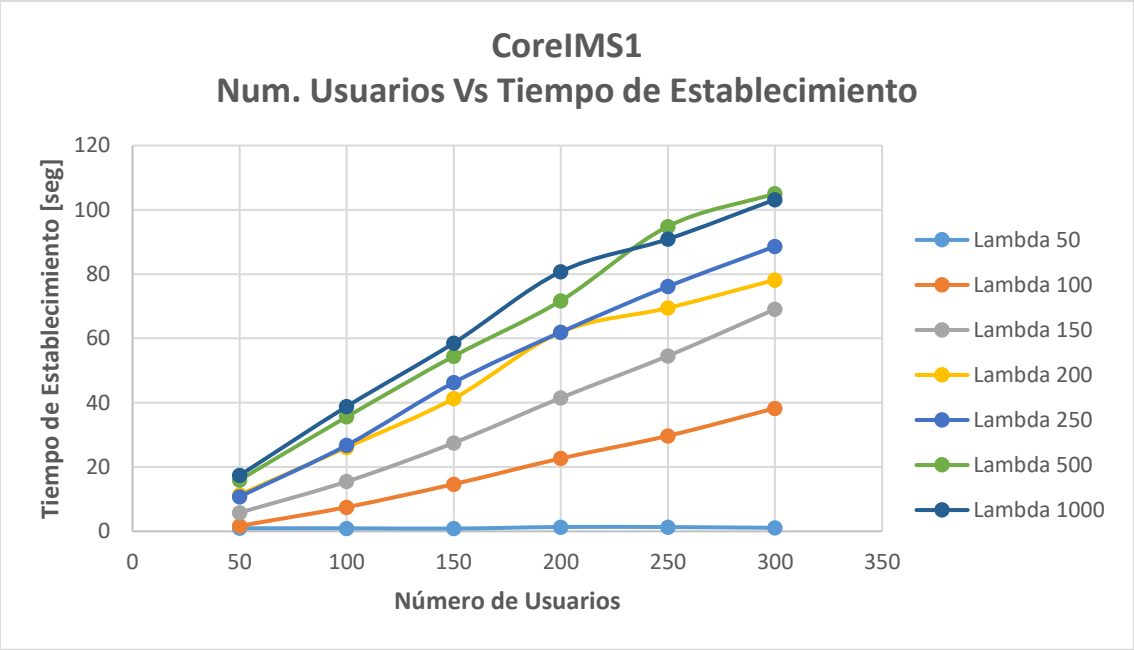

Figura 30. Tiempo de establecimiento del Core IMS 1. Fuente: Propia

También se observa que la variable que más afecta el rendimiento de Core es el número de usuarios que intentan registrarse, ya que a mayor número de usuarios que envían peticiones al Core, mayor es el número de solicitudes que debe procesar y por ende mayor es el tiempo de establecimiento de una video llamada. Por ejemplo, al comparar las líneas que corresponden a una lambda de 50 y de 100 (línea azul y naranja respectivamente), se observan los cambios en los tiempos de respuesta a medida que aumenta el número de usuarios. Cuando se tienen 50 usuarios con una lambda de 50 llam/min, el tiempo de respuesta es de 0,953696361 seg y con una lambda de 100 llam/min se tiene un tiempo de 0,95073505 seg, lo cual no muestra una diferencia significativa; pero cuando se sube a un extremo de 300 usuarios para una lambda de 50, se tiene un tiempo de respuesta de 1,089322307 seg y para una lambda de 100 el tiempo de respuesta es de 38,2736955 seg. Lo que indica que el número de usuarios es la variable que más afecta el rendimiento del Core IMS.

En la figura 31 se observa el tiempo de desconexión que se compone básicamente de la respuesta de dos mensajes SIP enviados como petición al servidor, por lo cual se tienen tiempos menores en comparación con los tiempos de conexión, pero con un comportamiento similar. Se observa que los promedios para una lambda de 50 siguen siendo tiempos pequeños y las demás líneas tienen tendencias similares, es decir, a mayor número de usuarios mayor es el tiempo de desconexión para cada lambda.

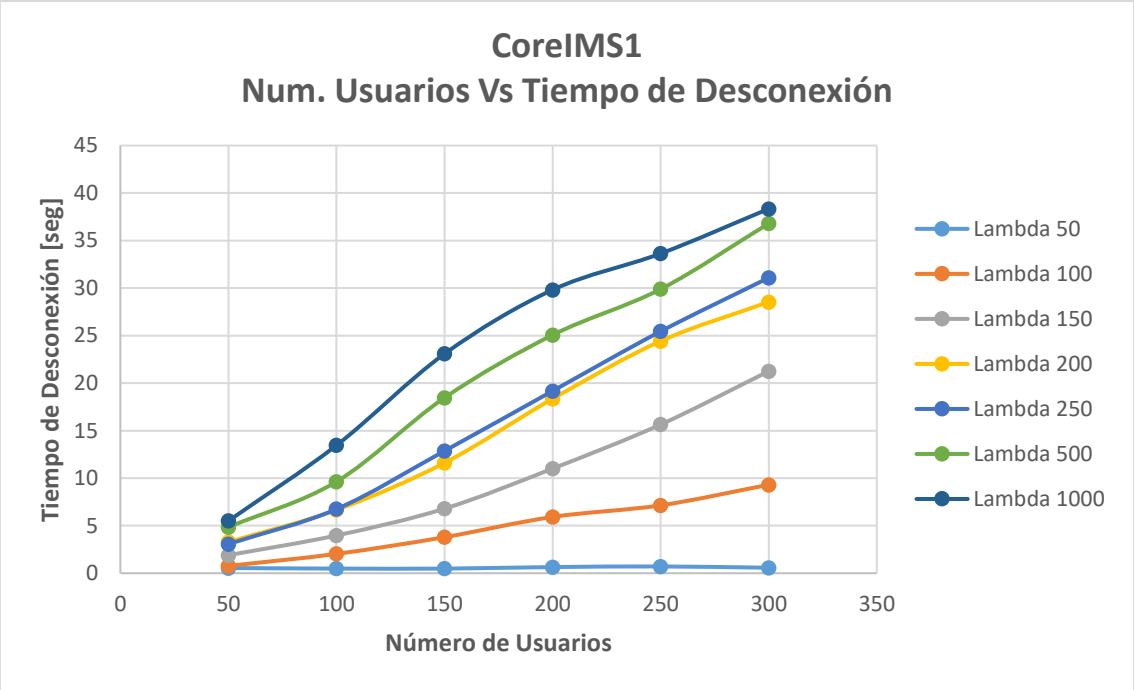

Figura 31. Tiempo de desconexión del Core IMS 1. Fuente: Propia

En cuanto a la pérdida de paquetes, se observan algunas variaciones, pero el comportamiento es similar, en donde a mayor cantidad de usuarios, mayor es el promedio de paquetes perdidos para cada una de las lambdas analizadas. En la figura 32 se observa el porcentaje del promedio de perdida de paquetes para cada lambda.

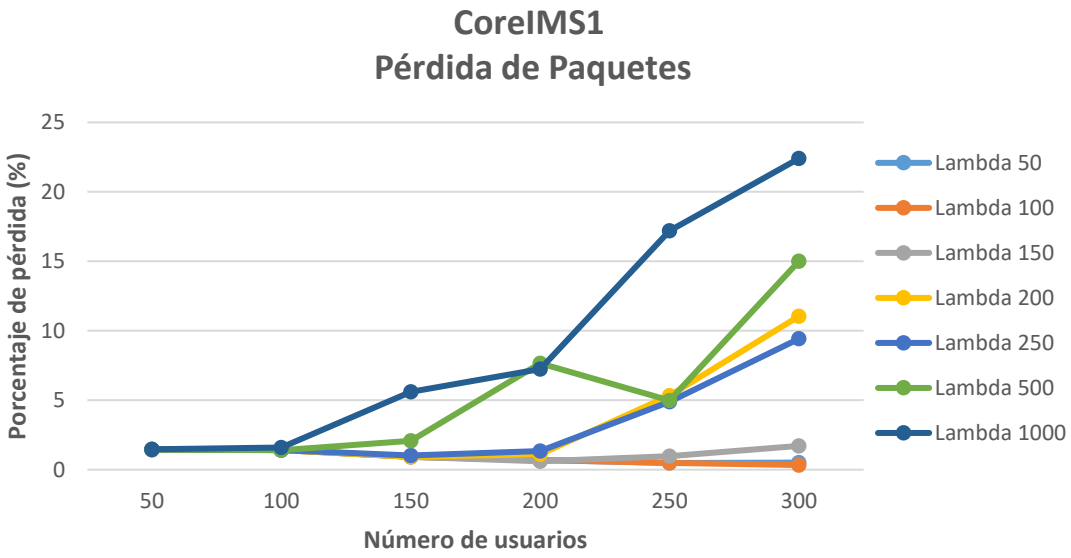

Figura 32. Perdida de paquetes del Core IMS 1. Fuente: Propia

4.2.1.3.2. Caso de estudio 2

En este caso de estudio se realizan las mismas pruebas del caso de estudio anterior, pero se evalúa el Core IMS 2 instanciado en la máquina virtual de 1GB

de RAM y 1 procesador. Los resultados obtenidos son muy similares a los que se obtuvieron del caso de estudio 1, a medida que aumentan el número de conexiones, el tiempo de establecimiento de la video llamada es mayor para cada una de las lambdas analizadas (figura 33). Esta tendencia es la misma para el tiempo de desconexión (figura 34).

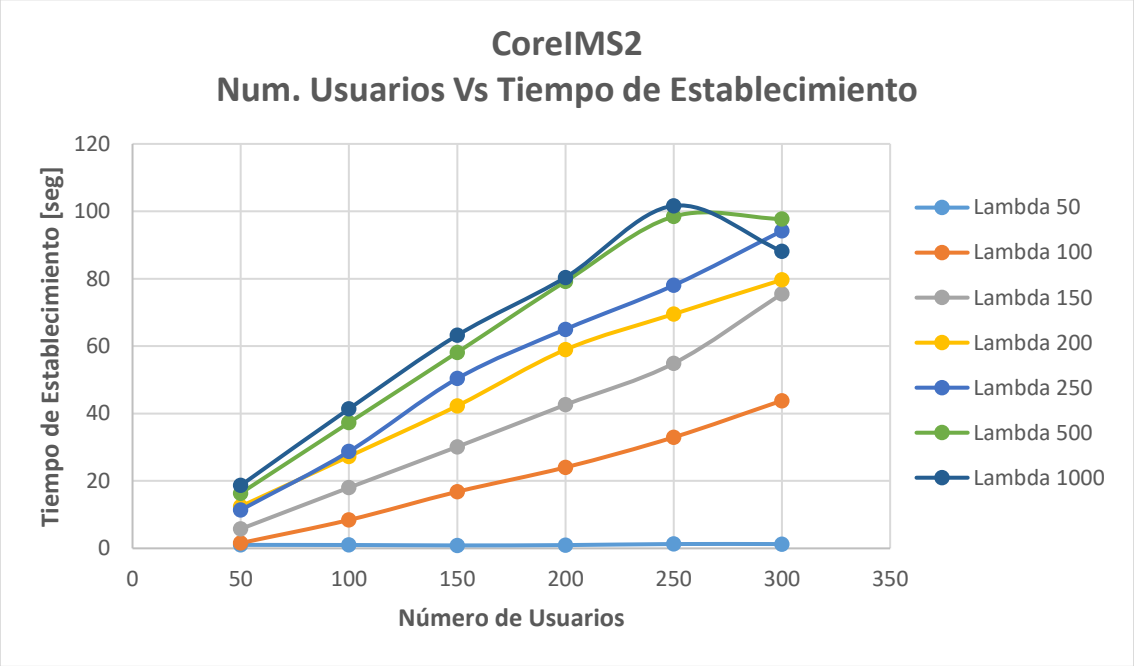

Figura 33. Tiempo de establecimiento del Core IMS 2. Fuente: Propia

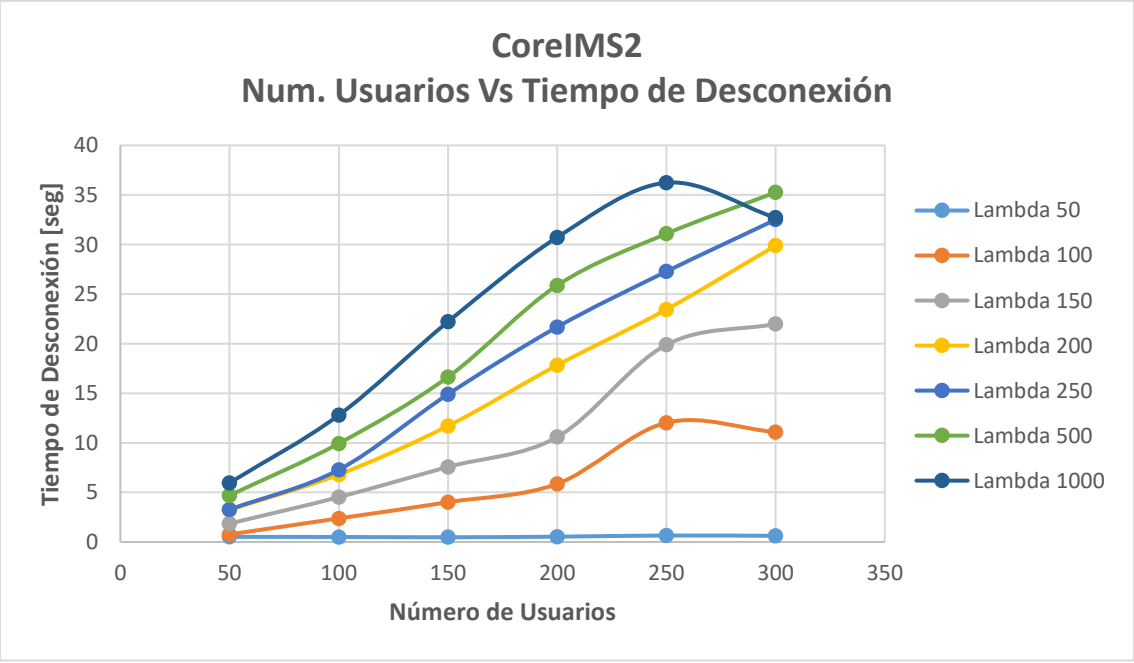

Figura 34. Tiempo de desconexión del Core IMS 2. Fuente: Propia

En cuanto a la perdida de paquetes, se mantiene la tendencia del caso de estudio 1 con ciertas variaciones. A medida que aumenta el número de usuarios el porcentaje de pérdida aumenta para cada una de las lambdas analizadas (figura 35).

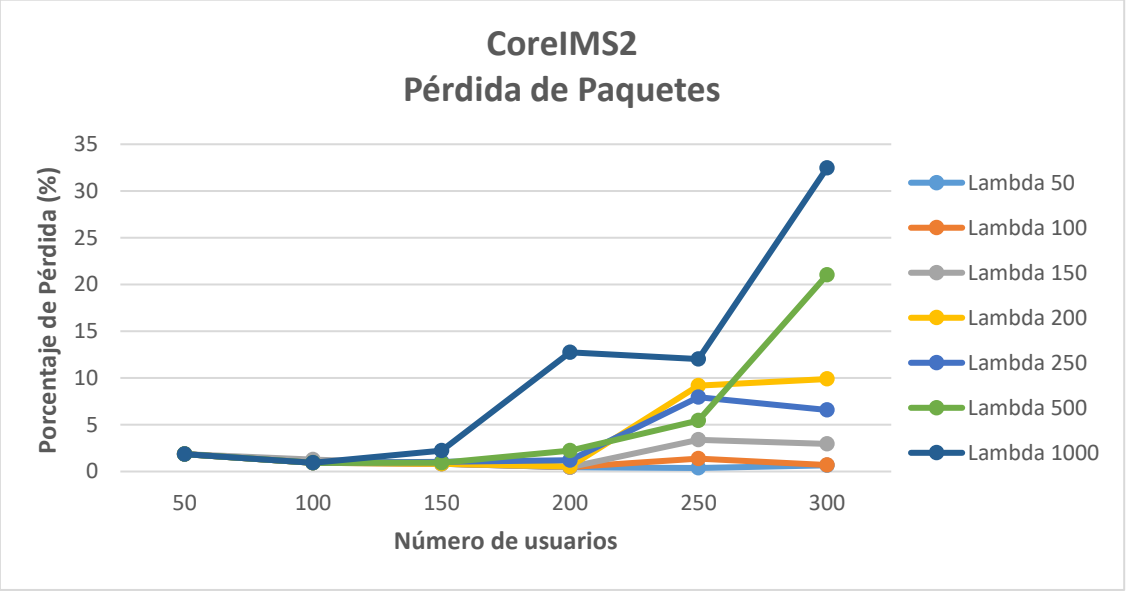

Figura 35. Perdida de paquetes del Core IMS 2. Fuente: Propia

4.2.1.3.3. Caso de estudio 3

En el caso de estudio 3, se evalúa el tercer Core instanciado en la máquina virtual de 1GB de RAM y 2 procesadores. Los resultados obtenidos son similares a los que se presentan en los casos anteriores, con las mismas tendencias como se puede apreciar en la figura 36 y figura 37, para tiempo de conexión y desconexión respectivamente. Al igual que para la pérdida de paquetes (figura 38).

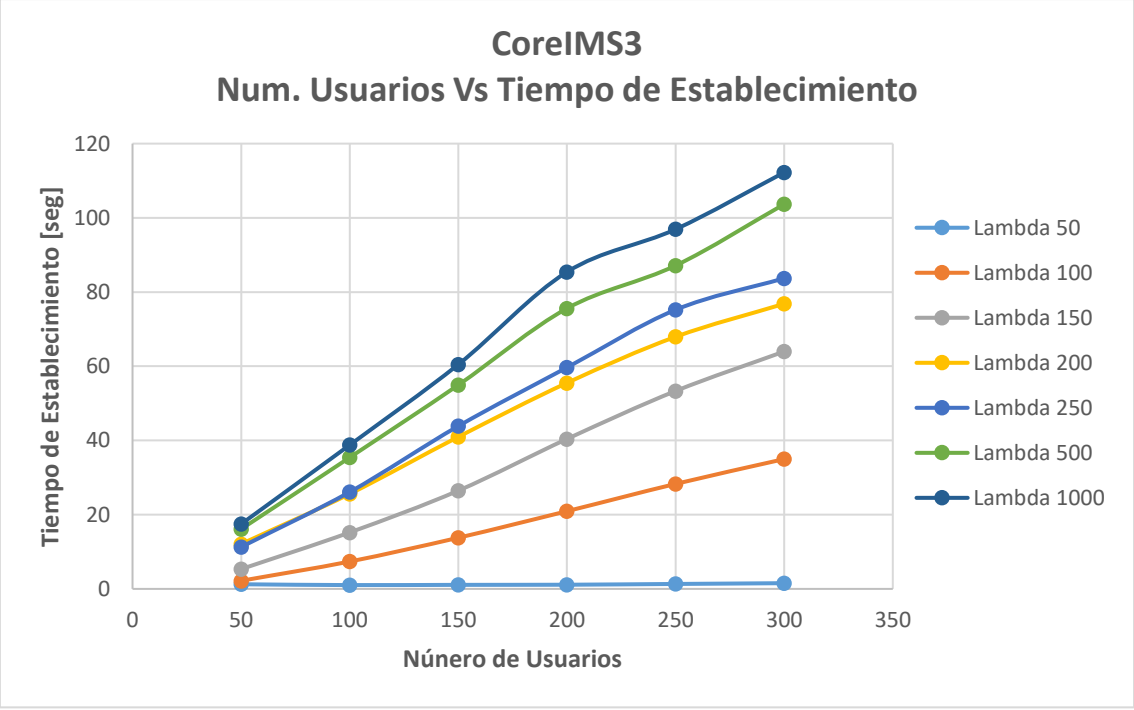

Figura 36. Tiempo de establecimiento del Core IMS 3. Fuente: Propia

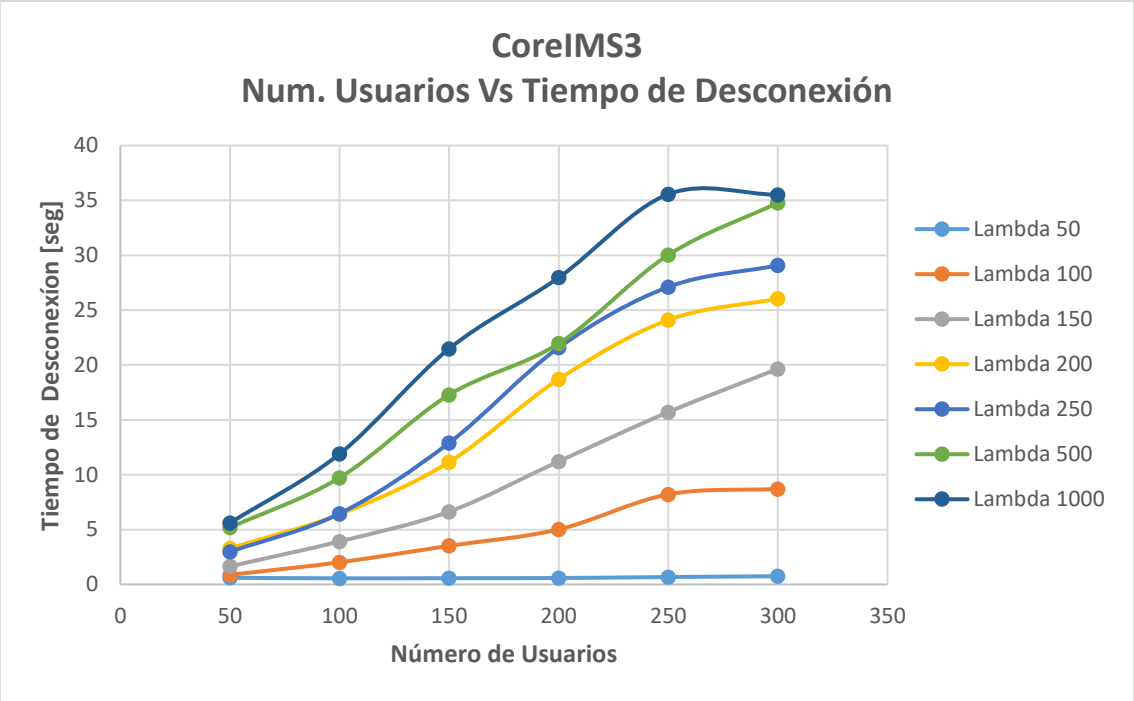

Figura 37. Tiempo de desconexión del Core IMS 3. Fuente: Propia

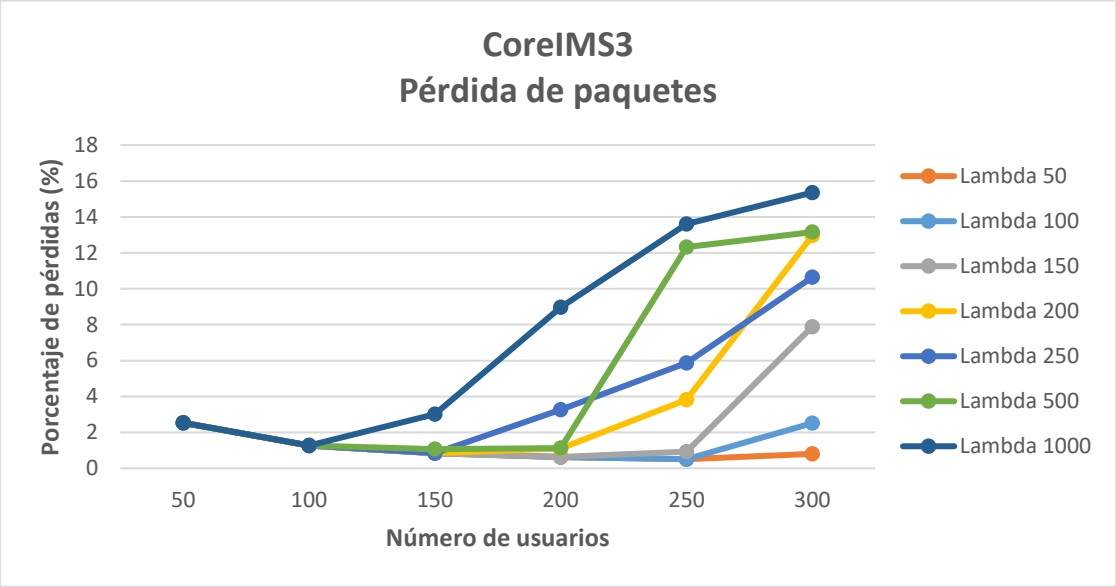

Figura 38. Perdida de paquetes del Core IMS 3. Fuente: Propia

4.2.1.3.4. Caso de estudio 4

Por último, para culminar esta sección de pruebas, se evalúa el cuarto Core instanciado en la máquina virtual de 2GB de RAM y 2 procesadores. Sus resultados tienen las mismas tendencias de los casos anteriormente estudiados como se puede apreciar en la figura 39, figura 40 y figura 41.

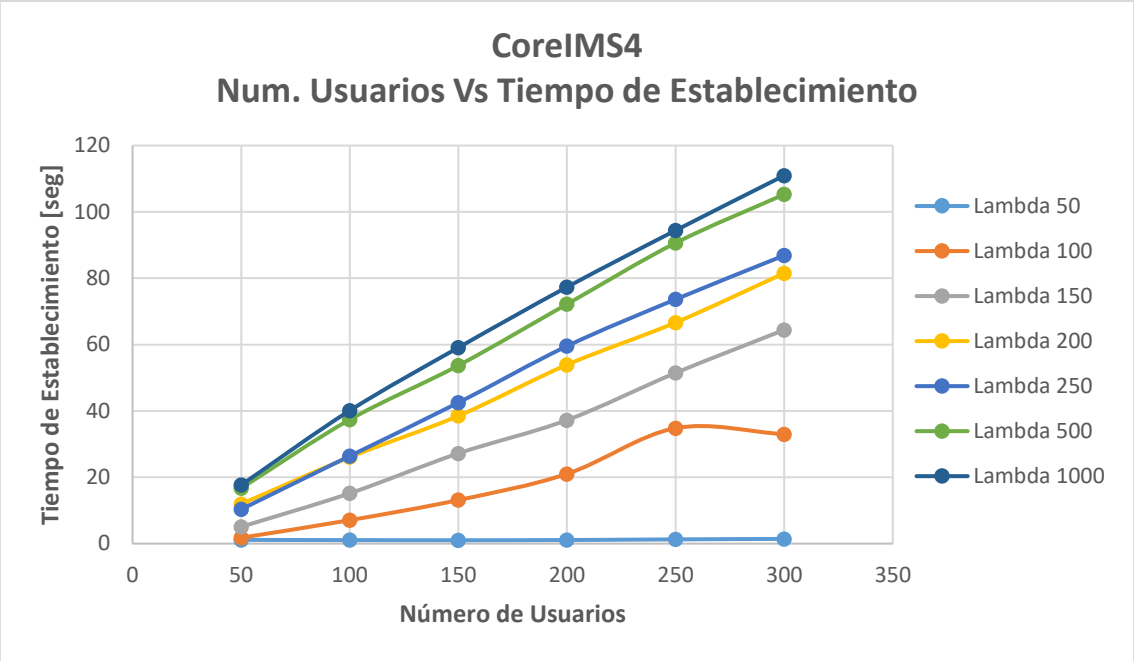

Figura 39. Tiempo de establecimiento del Core IMS 4. Fuente: Propia

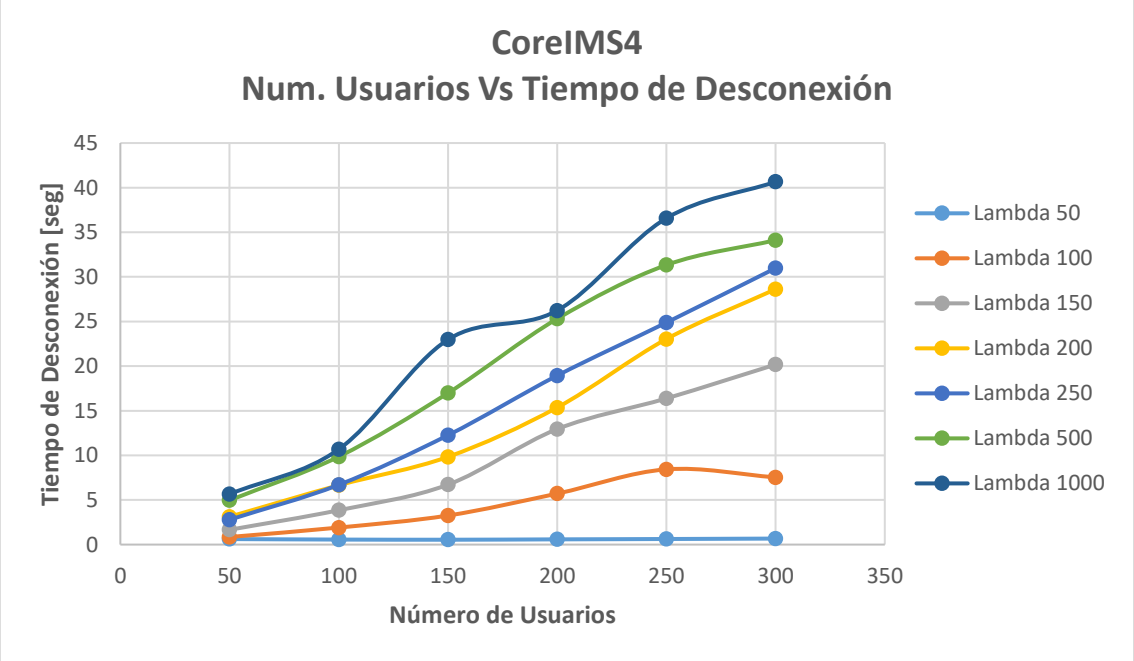

Figura 40. Tiempo de desconexión del Core IMS 4. Fuente: Propia

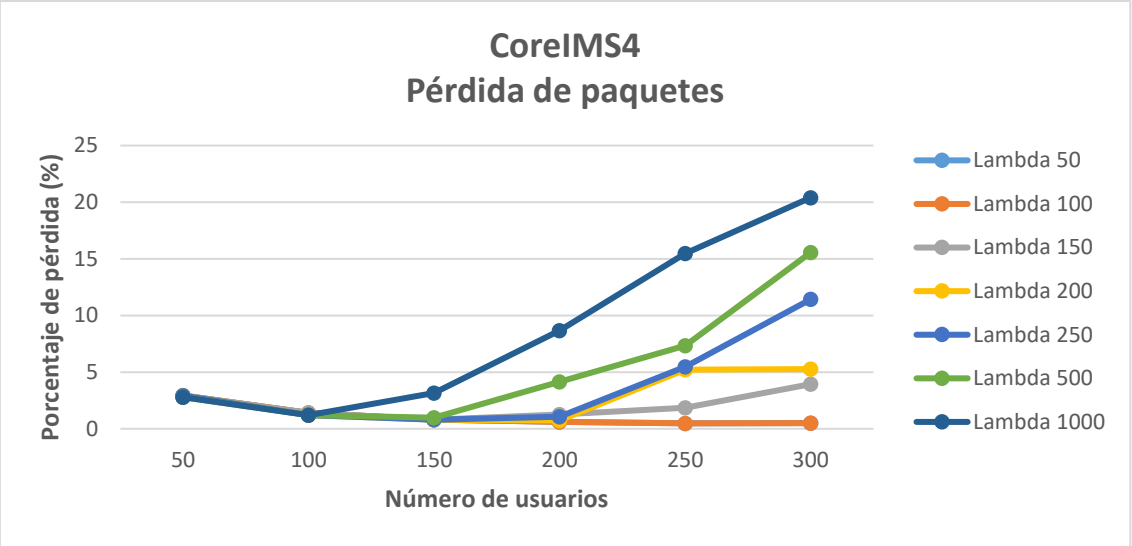

Figura 41. Perdida de paquetes del Core IMS 4. Fuente: Propia

4.2.1.4. Análisis de las pruebas

Con el fin de estudiar la escalabilidad vertical del Core IMS se analizan los resultados obtenidos para los casos de estudio propuestos. El objetivo de estas pruebas es determinar las mejoras en cuanto se van incrementando las características físicas de la máquina virtual donde está instanciado el Core. Estos resultados presentaron una gran similitud en cada caso de estudio, lo que genera una controversia a la hora de establecer una conclusión frente a las mejoras del Core, ya que se esperaba que al aumentar los recursos físicos de las máquinas virtuales se obtuvieran mejores resultados en cuanto al tiempo de respuesta, pero estos tiempos no mejoran a medida que se aumentan dichos recursos físicos. En la figura 42 se observa dicho comportamiento para cada Core IMS.

En cuanto a la perdida de paquetes, los resultados muestran que la tendencia de perdida es similar en cada caso de estudio. A mayor número de usuarios mayor será el número de paquetes perdidos en cada conexión. En la figura 43 se observa el porcentaje del promedio de pérdidas de cada caso de estudio, en donde el comportamiento mencionado anteriormente se mantiene.

A partir de estos resultados se puede indicar que la escalabilidad vertical no genera cambios sustanciales en el comportamiento del sistema al aplicarla sobre el Open IMS Core en una red virtualizada, ya que no se observan mejoras al aumentar los recursos físicos de las máquinas virtuales. Por lo cual, para las siguientes pruebas se utiliza la máquina virtual que tenga las características físicas más bajas para no desperdiciar recursos. En este caso la máquina virtual seleccionada, es la utilizada en el caso de estudio 1, la cual cuenta con 512 MB de RAM y 1 procesador.

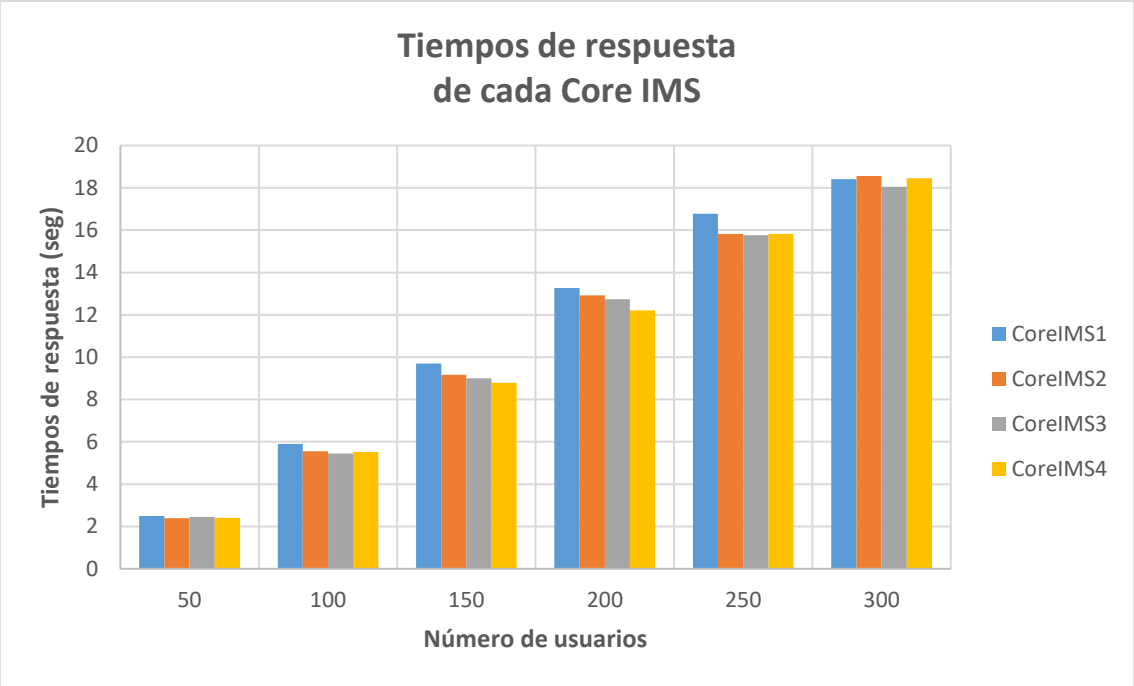

Figura 42. Tiempos de respuesta de cada Core IMS. Fuente: Propia

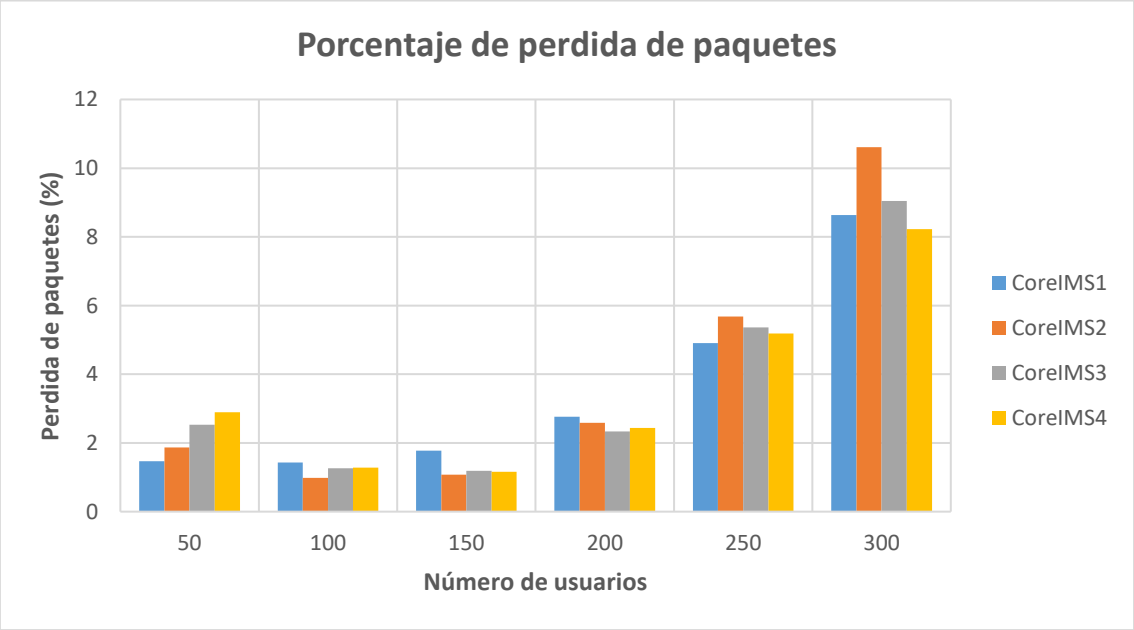

Figura 43. Perdida de paquetes de cada Core IMS. Fuente: Propia

4.2.2. Escalabilidad Horizontal

En este ítem de pruebas se pretende estudiar la escalabilidad horizontal del Core IMS al distribuir las peticiones de conexión entre diferentes Core IMS. En el análisis de la escalabilidad vertical, se concluye que no es viable implementarlo para este tipo de nodos (Open IMS Core), ya que al aumentar los recursos físicos del sistema no se ven mejoras en cuanto a la perdida de paquetes y tiempos de respuesta del Core. Por esta razón, para las pruebas de escalamiento horizontal, se usan máquinas virtuales con pocos recursos físicos

con el fin de no mal gastarlos e invertirlos en más máquinas virtuales para prestar una mejor calidad de servicio, atendiendo una mayor cantidad de usuarios que normalmente una sola máquina virtual puede atender.

En estas pruebas se pretende distribuir la carga de trabajo en varias instancias del Core IMS, tratando de que estas máquinas virtuales trabajen como una sola de tal forma que sea trasparente para el usuario.

4.2.2.1. Componentes de prueba

Como se mencionó anteriormente, las máquinas virtuales a utilizar poseen unas características bajas, ya que estas no afectan el funcionamiento del Core IMS, en este caso se utilizan máquinas virtuales de 512 MB de RAM y 1 procesador. Además, se utiliza el simulador de mensajes SIP para saturar los Core IMS y así estudiar el comportamiento de la escalabilidad horizontal, con el fin de determinar si esta escalabilidad mejora los datos obtenidos en la escalabilidad vertical.

En esta escalabilidad es necesario considerar como balancear la carga de trabajo, razón por la cual se utiliza un balanceador de carga que distribuya el tráfico entre los diferentes nodos de la red (Core IMS).

4.2.2.1.1. Balanceador de carga

Se desarrolla un balanceador de carga con el fin de distribuir las peticiones de mensajes SIP hacía varias máquinas virtuales. Se espera que el balanceador de carga ayude a minimizar tiempos de respuesta evitando la saturación y haciendo más eficiente el funcionamiento del Core.

El balance de la carga de trabajo se distribuye entre los nodos involucrados a través de un algoritmo de balanceo de carga, esto con el fin de evitar los cuellos de botella. El algoritmo de balanceo de carga escogido fue Round-Robin, ya que planifica de manera equitativa y ordenada él envió de peticiones, tratando a todos los nodos involucrados con la misma prioridad.

El balanceador de carga es desarrollado con el mismo lenguaje de programación, librerías y funcionamiento del simulador de mensajes SIP, el cual se encarga de recibir los paquetes SIP provenientes del simulador de mensajes y redireccionarlos entre las máquinas virtuales asociadas al balanceador de acuerdo con el algoritmo de balanceo de carga. Este a su vez escucha las respuestas de los Core y las envía de nuevo al usuario en cuestión, en este caso el simulador de mensajes SIP.

La lógica utilizada por el balanceador de carga consiste en recibir el primer mensaje SIP (“Register”), este se almacena en la base de datos del balanceador de tal forma que cada vez que llegue un mensaje de cierto

usuario, este lo redirija al Core en el cual se registró inicialmente el cliente. El balanceador está diseñado de una forma sencilla, practica y óptima para que el tiempo de procesamiento de paquetes sea mínimo, esto es primordial ya que a medida que aumentan los usuarios o el valor de lambda, el tráfico de paquetes SIP aumenta de una forma considerable. En el Anexo B.4 se explica el funcionamiento del balanceador de carga.

4.2.2.2. Plan de pruebas

En el plan de pruebas se realiza prácticamente la misma metodología usada en la sección 4.2.1, referente al estudio de la escalabilidad vertical del Core, con el objetivo de comparar resultados y concluir cuál de las dos escalabilidades es la más adecuada para implementar en una red IMS virtualizada.

Dicha metodología consiste en simular la conexión de usuarios en los Core asociados al balanceador de carga obteniendo un promedio de los tiempos de respuesta y promedio de paquetes perdidos. De igual manera se tiene un total de 42 pruebas, cada una corroboradas 3 veces para verificar la efectividad de los datos obtenidos.

4.2.2.3. Resultados de las pruebas

Como se realizó en las pruebas de escalabilidad vertical, se estudia el tiempo de respuesta de peticiones de los Core asociados al balanceador de carga, así como de la perdida de paquetes. Estos resultados no se verán por separado como si fueran Core diferentes, estos dan un tiempo de respuesta como si trabajara una sola máquina virtual. A este tiempo, se le adiciona el tiempo de procesamiento del balanceador de carga que, aunque es mínimo, aproximadamente 0.003 segundos, puede sumar al tiempo de respuesta total.

Se analizan 2 casos de estudio, en el primer caso de estudio se asocian dos máquinas virtuales al balanceador de carga, con el objetivo de analizar los resultados cuando se duplica el número de Core. En el segundo caso de estudio se examina la escalabilidad horizontal cuando se asocian 3 Core al balanceador de carga.

4.2.2.3.1. Caso de estudio 1

Como se mencionó anteriormente, en este caso de estudio se asocian 2 máquinas virtuales al balanceador de carga, cada una de ellas se caracteriza por tener una memoria RAM de 512 MB y 1 procesador. Se obtiene el tiempo de conexión (figura 44) y desconexión (figura 45) para los diferentes valores de lambda y número de usuarios.

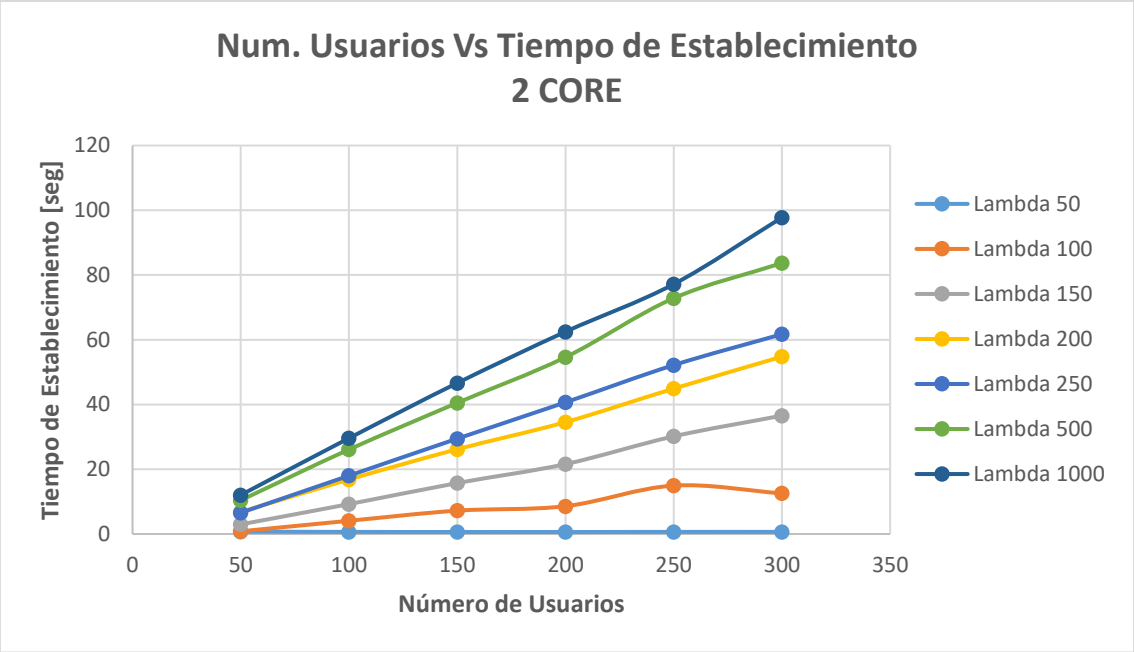

Figura 44. Tiempo de establecimiento de 2 Core IMS. Fuente: Propia

Como es de esperarse a medida que crece el número de usuarios el tiempo de respuesta de los Core asociados se eleva proporcionalmente, estos tiempos también se elevan con respecto al valor de lambda. Se obtienen muy buenos resultados con valores de lambda de 50, ya que estos tiempos no sobrepasan 1 segundo. Por otra parte, a partir de lambda 150 los tiempos de respuesta aumentan considerablemente.

En la figura 45 se observa el tiempo de desconexión para diferentes valores de lambda y número de usuarios.

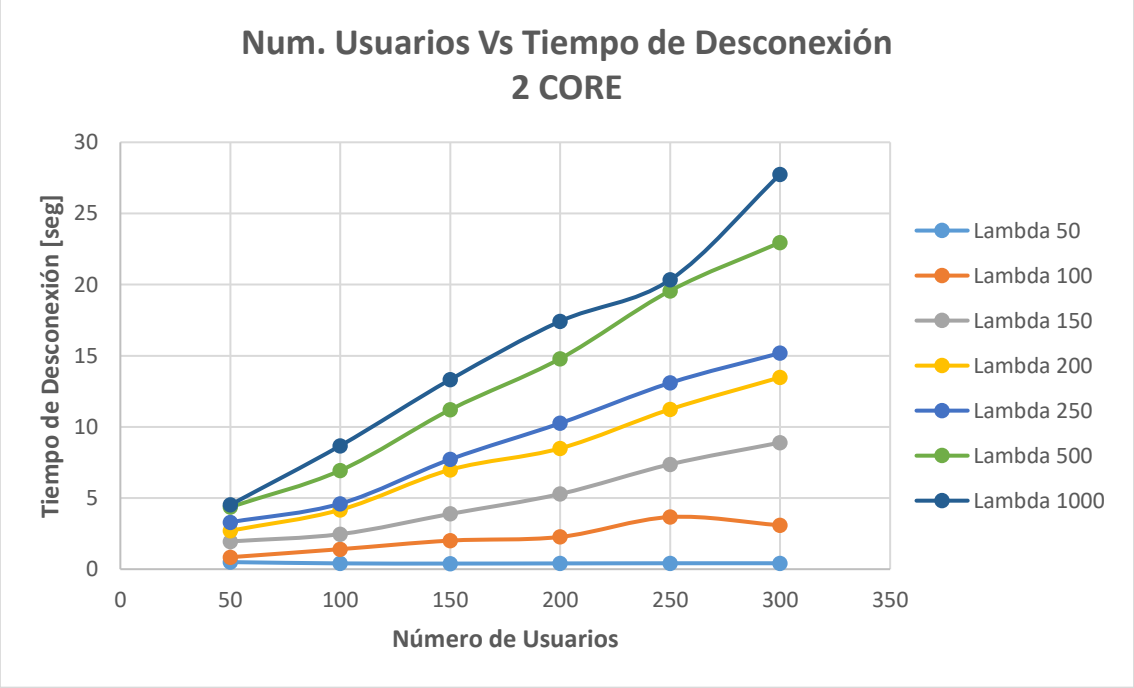

Figura 45. Tiempo de desconexión de 2 Core IMS. Fuente: Propia

Al observar la anterior grafica se observan buenos tiempos de respuesta para la desconexión en los valores de lambda de 50 y 100. Se tiene el mismo comportamiento que los tiempos de respuesta para la conexión, pero son más bajos debido a que se necesitan menos paquetes para la desconexión.

4.2.2.3.2. Caso de estudio 2

Para el segundo caso de estudio se utilizó el mismo esquema y ambiente de red usado en el anterior caso de estudio, donde el tráfico de peticiones se divide con ayuda de un balanceador de carga para diferentes Core. En este caso se asocian 3 Core al balanceador de carga, donde el tráfico SIP se divide de cuerdo al mismo algoritmo de balanceo de carga usado en el anterior caso de estudio. Las características físicas de dichas máquinas virtuales son iguales, con 512 MB de memoria RAM y 1 procesador.

En la figura 46 se muestra el tiempo de conexión para cada uno de los valores de lambda al variar el número de usuarios.

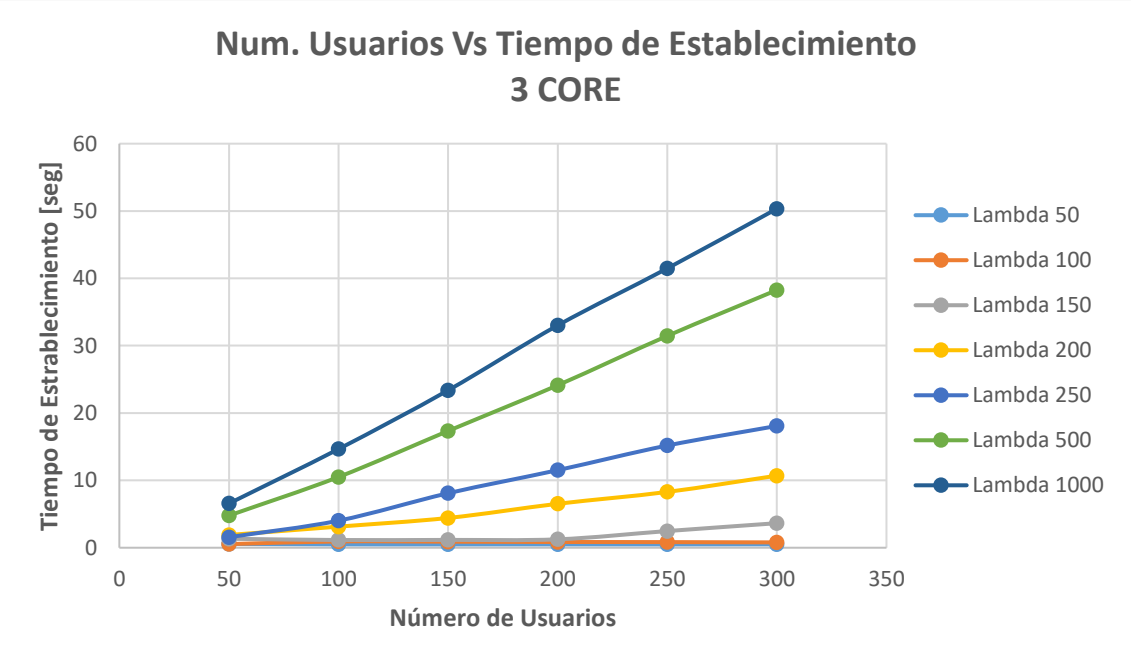

Figura 46. Tiempo de establecimiento de 3 Core IMS. Fuente: Propia

Como se observa en la anterior grafica a medida que aumenta el número de usuarios, el tiempo de respuesta también se incrementa. El comportamiento es similar al primer caso de estudio, pero se observa una mejora considerable en los tiempos de respuesta obtenidos para los usuarios que son atendidos por tres Core con respecto a los que son atendidos por dos.

También se analizan los tiempos de respuesta de desconexión (figura 47), en donde para los tres primeros valores de lambda, estos tiempos están por

debajo de 2 segundos, valores aceptables para finalizar la conexión de un usuario en el Core.

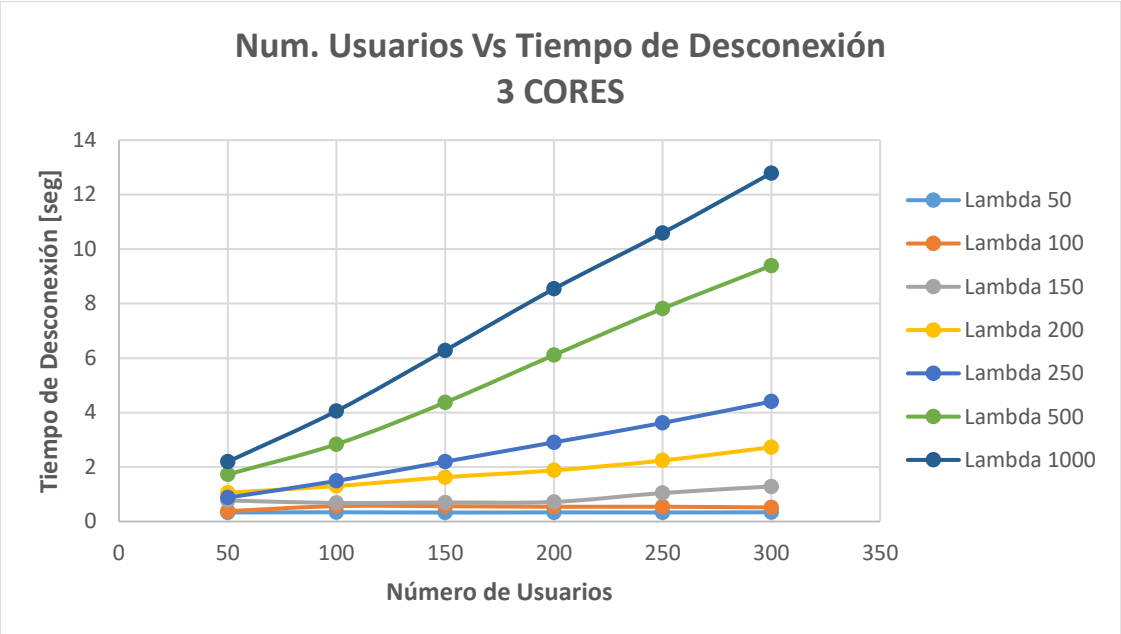

Figura 47. Tiempo de desconexión de 3 Core IMS. Fuente: Propia

4.2.2.4. Análisis de las pruebas

Con el fin de estudiar la escalabilidad horizontal del Core IMS se analizan los resultados obtenidos para los casos de estudio propuestos. En estos resultados se comparan los promedios de los tiempos de respuesta según el número de usuarios para cada caso de estudio, como se observa en la figura 48.

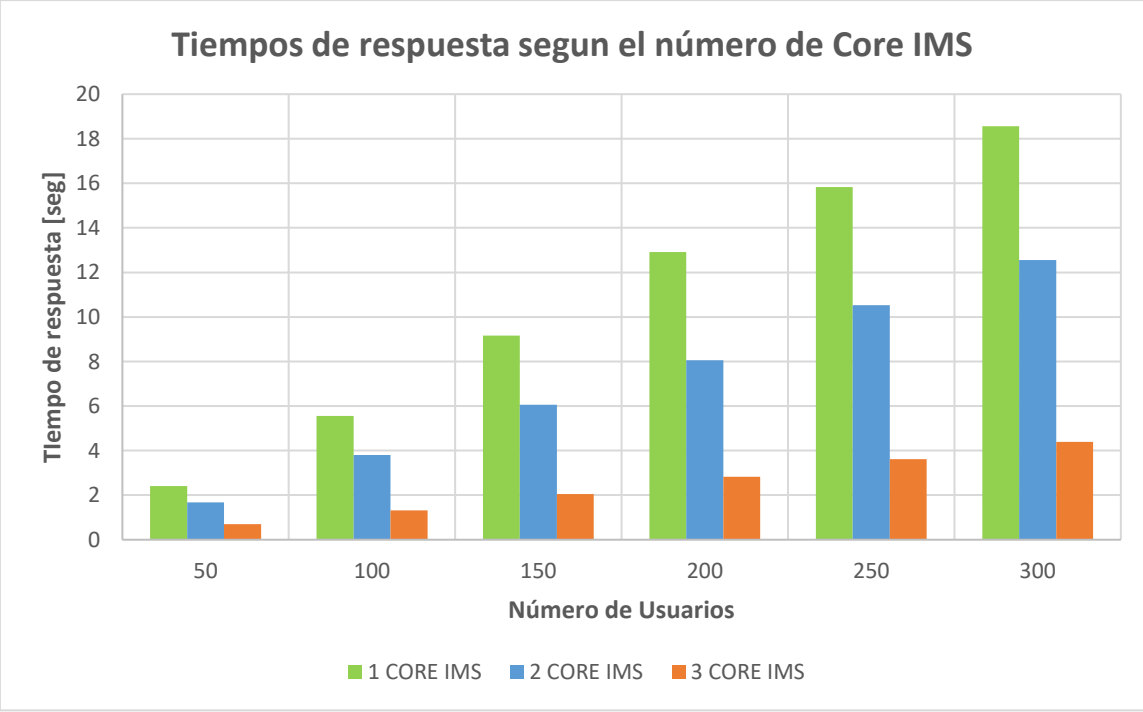

Figura 48. Tiempos de respuesta según el número de Core IMS. Fuente: Propia

Cabe notar que los promedios de los usuarios atendidos por una sola máquina virtual fueron tomados de los resultados obtenidos en las primeras pruebas donde se analiza la escalabilidad vertical del Core, utilizando las mismas características físicas de esta sección (RAM 512 MB y 1 procesador).

Como se observa en la figura 48, es muy factible implementar escalamiento horizontal a este tipo de nodos, ya que al dividir el tráfico de peticiones entre varios Core, el tiempo de respuesta disminuye considerablemente a medida que aumenta el número de usuarios en la red. Esto mejora la eficiencia de la red, ya que todos los Core asociados trabajan como uno solo, disminuyendo el tiempo de respuesta a medida que el número de usuarios crece.

Para analizar dicho progreso se grafica el porcentaje de mejora según el número de Core IMS (figura 49), en donde este porcentaje representa el promedio de los tiempos de respuesta de los usuarios atendidos por 2 y 3 Core con respecto a los clientes atendidos por un Core.

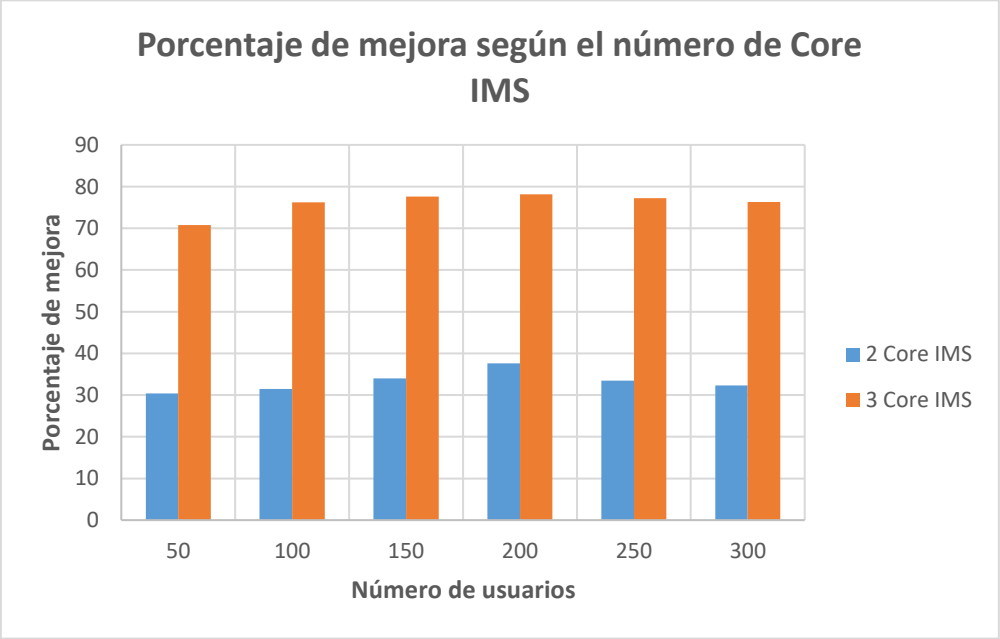

Figura 49. Porcentaje de mejora según el número de Core IMS. Fuente: Propia

En esta figura se observa la mejora al aumentar el número de Core, ya que para 2 Core se tiene un promedio de porcentaje de mejora del 33% y para 3 Core el porcentaje es del 76%, lo que indica que al aumentar el número de Core IMS el tiempo de respuesta disminuye, aumentando así el número de usuarios registrados en el Core IMS.

Por otra parte, se compara el porcentaje promedio de perdida de paquetes con respecto del número de usuarios y el número de Core asociados al balanceador de carga, como es observa en la figura 50.

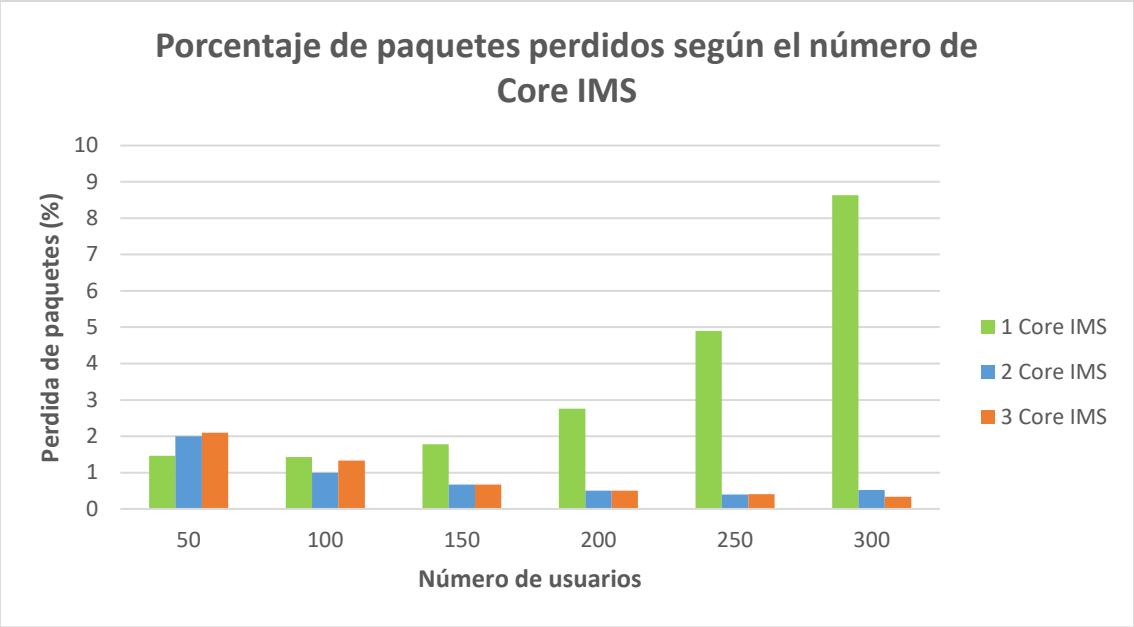

Figura 50. Perdida de paquetes según el número de Core IMS. Fuente: Propia

Como se observa en la anterior gráfica, el porcentaje de perdida de paquetes SIP disminuye considerablemente al aumentar el número de Core IMS que reciben las peticiones de los usuarios. Esto se debe a que el balanceador distribuye los paquetes entre los diferentes Core permitiéndoles procesar una mayor cantidad de peticiones.

De acuerdo con los estudios de escalabilidad vertical y horizontal, se seleccionan los valores de lambda más adecuados. La primera lambda es de 50, ya que presenta un comportamiento ideal en cuanto a tiempos de respuesta y perdida de paquetes. Una lambda de 150 la cual muestra valores aceptables y en el peor de los casos una lambda de 250, ya que los tiempos de respuestas y las pérdidas son considerables.

4.2.3. Parámetros de Calidad

En esta sección de pruebas se pretende analizar los parámetros de calidad como ancho de banda, retardo, “jitter” y perdida de paquetes, con el fin de encontrar los valores óptimos para que una video llamada funcione correctamente. La metodología de estas pruebas consiste en establecer una video llamada entre dos clientes IMS limitando el ancho de banda de los equipos donde están los usuarios. Previamente se pone en funcionamiento el simulador de mensajes SIP con los valores de lambda seleccionados anteriormente y así analizar cuanto tiempo se demora en conectar y desconectar una video llamada de un cliente IMS (Fokus Monster).

Una vez establecida la comunicación, se analizan los paquetes RTP desde que inicia hasta que termina la video llamada por medio de Wireshark, el cual permite determinar los valores de perdida de paquetes, retardo y “jitter”. Del

mismo modo, se estudia cómo se ven afectados dichos parámetros por la limitación de ancho de banda en la video llamada.

4.2.3.1. Componentes de prueba

Dentro de los componentes de pruebas está el simulador de mensajes SIP usado en las anteriores pruebas, los dos equipos correspondientes para cada usuario, y una máquina virtual donde está instalado el Core IMS. Como se mencionó en la sección de pruebas anterior, se utiliza la máquina virtual con menor cantidad de recursos físicos (512 MB RAM y 1 procesador), de esta forma no se desperdician recursos. Por otra parte, para limitar el ancho de banda se utiliza la herramienta Wondershaper y para capturar todo el tráfico de la red se usa Wireshark.

4.2.3.2. Plan de pruebas

Para realizar estas pruebas de ancho de banda se utiliza una sola máquina virtual con las características mencionadas y se establecen diferentes casos de estudio tomando diversos valores de ancho de banda para cada valor de lambda seleccionada anteriormente.

Los valores de ancho de banda están basados en el tipo de códec utilizado, en este caso los códec H.263 y G.711 como mínimo trabajan con un ancho de banda de 64 Kbps, por lo cual se toman valores múltiplos de 64 hasta llegar a 1024 para un total de 16 valores diferentes de ancho de banda. Estos valores se tomaron de esta forma ya que no se encontró en el estado del arte valores de ancho de banda óptimos para que una video llamada tenga una buena calidad de servicio. En total se realizan 48 casos de estudio, cada uno repetido tres veces para corroborar los resultados.

4.2.3.3. Resultados de las pruebas

A continuación, se presentan los resultados más relevantes y concretos de estas pruebas, analizando el efecto de la variabilidad del ancho de banda a la calidad de la video llamada.

Inicialmente se pensaba que se podían obtener resultados consistentes al añadir el simulador de mensajes SIP a las pruebas, ya que este simula el establecimiento de la conexión entre el Core y múltiples usuarios, generando tráfico al momento de establecer una video llamada entre dos clientes reales. Como abría de esperarse, cuando los clientes IMS querían registrarse en el Core o establecer una video llamada los retardos en el tiempo de respuesta eran evidentes por el usuario, ya que este no responde inmediatamente a las peticiones. Esto confirma que el simulador si cumple con la tarea de estresar el Core IMS pero los resultados no fueron muy favorables al estudiar los tiempos de respuesta de los mensajes SIP obtenidos para los usuarios IMS.

Se sabe que los tiempos de respuesta se ven afectados por el tráfico que hay en la red a causa del simulador de mensajes SIP. Esta congestión hizo que los tiempos de respuesta, obtenidos en cada una de las video llamadas de prueba, se alejen mucho entre sí, logrando una gran dispersión entre los datos e impidiendo corroborar la información obtenida. Este efecto es más notorio en los tiempos de desconexión ya que estos tiempos no son estables, en algunas ocasiones son elevados y en otras son reducidos. Esto se debe a la gran cantidad de llamadas atendidas por el Core.

Para evidenciar este efecto, se grafican los tiempos de conexión y desconexión, para cada uno de los valores de lambda y ancho de banda, como se observa en la figura 51 y figura 52 respectivamente.

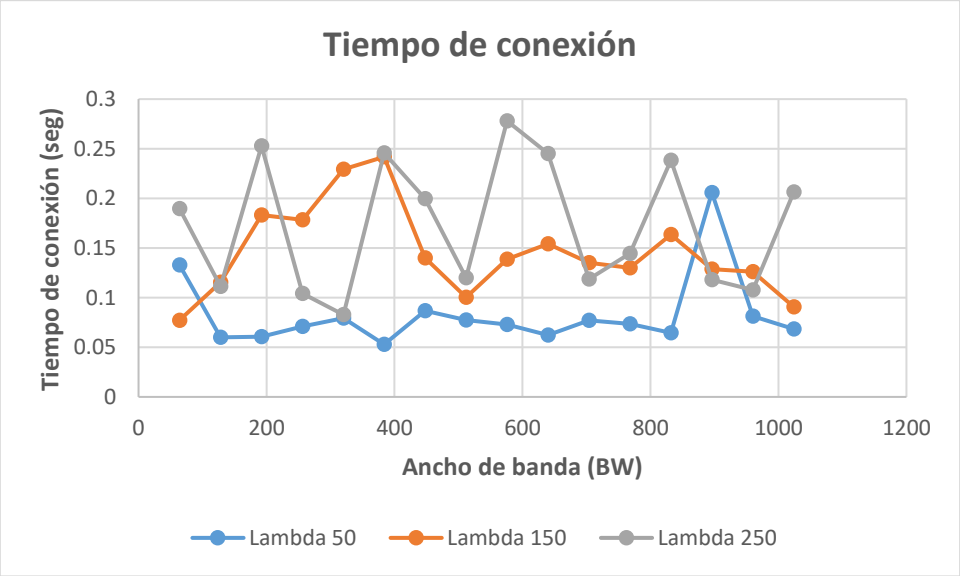

Figura 51. Tiempo de Conexión. Fuente: Propia

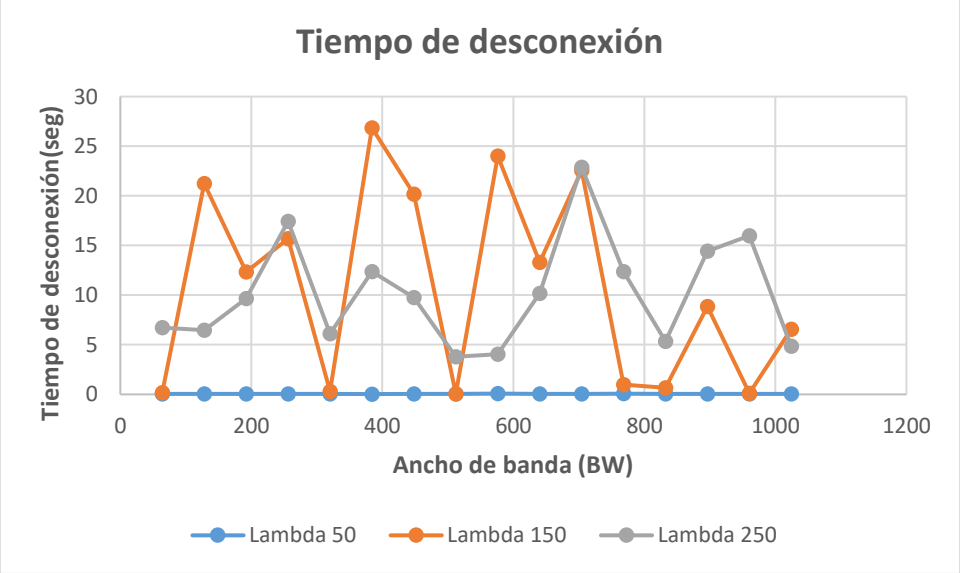

Figura 52. Tiempo de desconexión. Fuente: Propia

El parámetro lambda no afecta a las variables relacionadas con los paquetes RTP, ya que estos paquetes intercambiados entre los usuarios no tienen relación con el funcionamiento del Core. La única dependencia que tiene el servicio es al momento de registrarse, establecer una llamada o des-registrarse, ya que intercambian mensajes SIP. Por lo tanto, en el análisis de los parámetros de calidad no se tiene en cuenta el parámetro lambda.

El estudio de los parámetros de calidad se realiza por medio de Wireshark, en donde se utilizan filtros para analizar la captura de paquetes RTP desde uno de los usuarios de la comunicación. Esta herramienta examina los paquetes y determina diferentes parámetros de estudio, como dirección IP de la fuente y del destino, número de paquetes enviados, número de paquetes perdidos, tipo de códec utilizado, “jitter” máximo, entre otros. De esta manera se realiza el estudio de cada una de las pruebas realizadas, arrojando los siguientes resultados.

4.2.3.3.1. Retardo

En primer lugar, se analiza el retardo de los paquetes RTP entre usuarios. Como se explica en el estado del arte hay diferentes tipos de retardos, entre ellos se encuentran los retardos de cola, de transmisión, de propagación y de procesamiento. Para este caso de estudio, las pruebas realizadas están bajo condiciones de red óptimas (pruebas de laboratorio), por ende, no hay nodos que puedan sumar retardos de cola y de procesamiento, ni tampoco hay enlaces largos que puedan sumar retardos de trasmisión significativos. Por lo tanto, se toma el retardo de trasmisión como retardo considerable en estas pruebas.

Para calcular el retardo de transmisión se usan dos características de la transmisión, la longitud de paquetes (L) en bits y la taza de transmisión de paquetes (R) en bits por segundo, expresada por la ecuación 4 [27].

$$D_{tx}[seg] = \frac{L[bits]}{R[bits/seg]} \qquad (4)$$

Para calcular este retardo de transmisión, se toman los paquetes tanto de audio como de video. En la figura 53 se observan los resultados obtenidos para los diferentes anchos de banda.

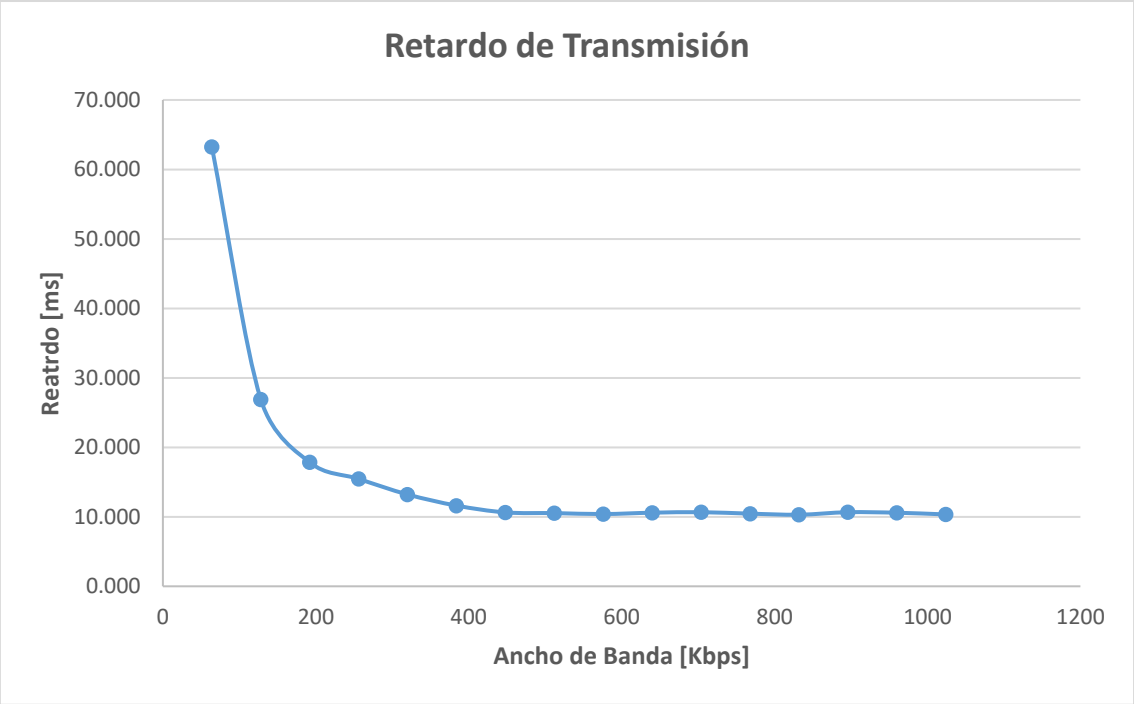

Figura 53. Retardo de transmisión. Fuente: Propia

Esta grafica tiene un comportamiento esperado, ya que a medida que aumenta el ancho de banda los retardos disminuyen. El retardo se estabiliza con un valor de 10 ms a partir de un ancho de banda de 448 Kbps, lo que indica que por más que se aumente el ancho de banda a partir de este valor no se van a tener mejoras en los retardos de transmisión.

Según la teoría, los valores aceptables deben ser inferiores a 150 ms, en este caso los valores obtenidos para cada ancho de banda son inferiores a 65 ms, lo cual indica que a partir de un ancho de banda de 64 Kbps el retardo de transmisión no afecta significativamente la comunicación.

Es necesario hacer énfasis en que estas pruebas se realizaron en un entorno ideal, razón por la cual los valores de retardo obtenidos son relativamente bajos. Si se escala la red, aumentando nodos intermedios entre los clientes, estos valores del retardo van a aumentar, incrementando el mínimo ancho de banda que la red debe tener para garantizar niveles de calidad de servicio adecuados.

4.2.3.3.2. Pérdida de paquetes

Otro de los parámetros principales que afecta la transmisión de paquetes RTP es la pérdida de paquetes. En los resultados obtenidos se tienen dos tipos de análisis, uno para paquetes de audio y otro para paquetes de video. Se dividen de esta forma ya que los paquetes de video presentan más perdidas debido a la gran demanda de ancho de banda, afectando así la calidad del video.

Se analiza el número de paquetes recibidos y el número de paquetes perdidos tanto de audio (figura 54) como de video (figura 55) al variar el ancho de banda.

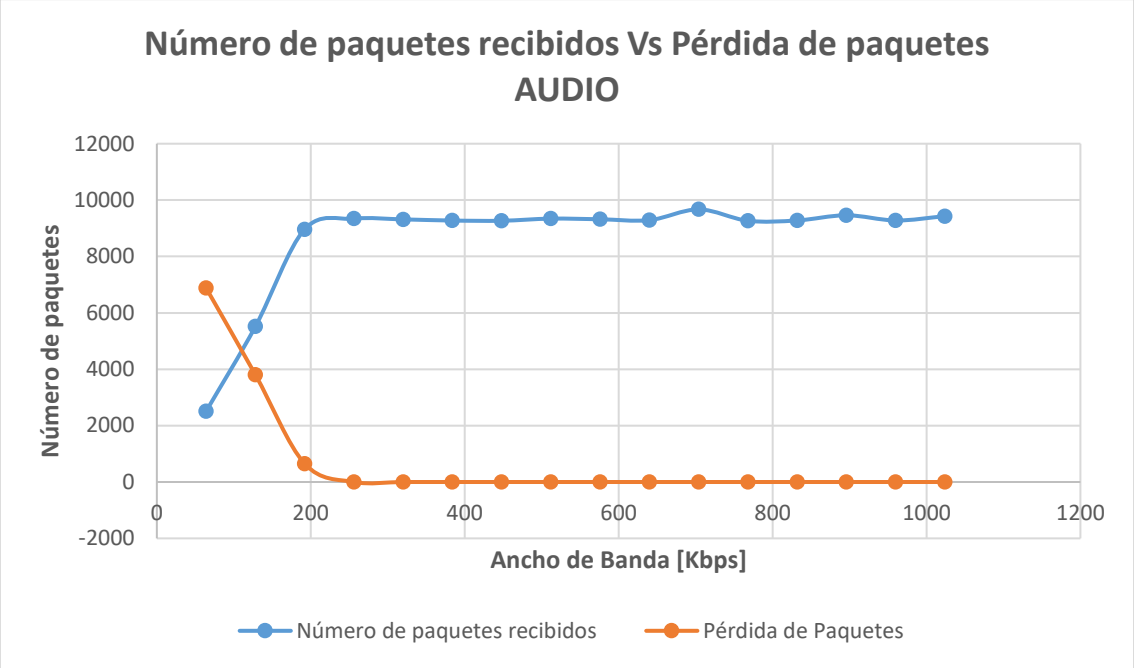

Figura 54. Perdida de paquetes de audio. Fuente: Propia

Según la teoría, una pérdida de paquetes del 1% puede generar pérdida de audio, en este caso se evidencia que las pérdidas de paquetes de audio desaparecen (0%) a partir de un valor de ancho de banda de 256 Kbps.

Los paquetes de video tienen un comportamiento similar a la gráfica de los paquetes de audio ya que, a menor ancho de banda, mayor será la perdida de paquetes.

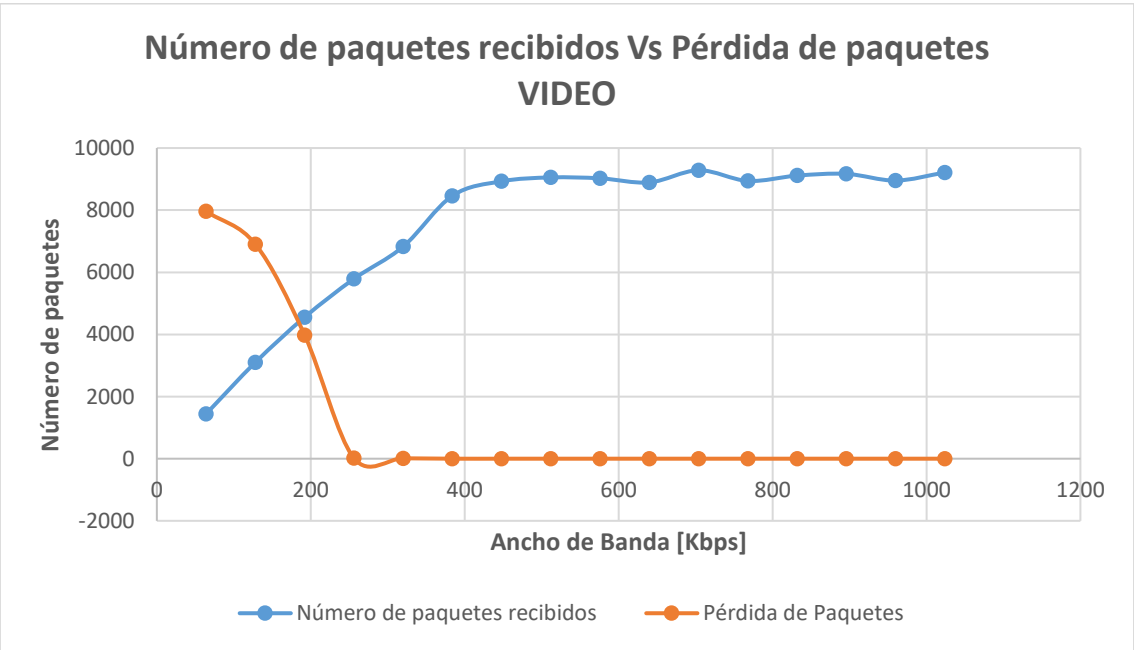

Figura 55. Perdida de paquetes de video. Fuente: Propia

La pérdida de paquetes del 1% puede generar congelamiento en el video, en este caso no se presentan perdida de paquetes a partir de un ancho de banda de 384 Kbps. Por esta razón la transmisión de video demanda un valor de ancho de banda más elevado para lograr una buena calidad de servicio.

Cabe anotar que, en los paquetes de audio, al estabilizarse la pérdida de paquetes también se mantiene el número de paquetes recibidos, pero los paquetes de video necesitan mayor ancho de banda, ya que al estabilizarse la perdida de paquetes, estos requieren de un mayor ancho de banda para transmitir más paquetes y mantener la calidad del servicio. En este caso el número de paquetes recibidos es constante a partir de 448 Kbps y la perdida de paquetes desaparece a partir de 384 Kbps.

4.2.3.3.3. "Jitter"

Para que la reproducción del video sea correcta, el tiempo entre el despliegue de los paquetes RTP debe ser adecuado, por lo que en esta sección se muestra cómo afecta al "jitter" la variación del ancho de banda. En primera instancia se analiza el "jitter" para los paquetes de audio. En la figura 56 se observa el valor medio que toma el "jitter" y el valor máximo alcanzado para este grupo de datos.

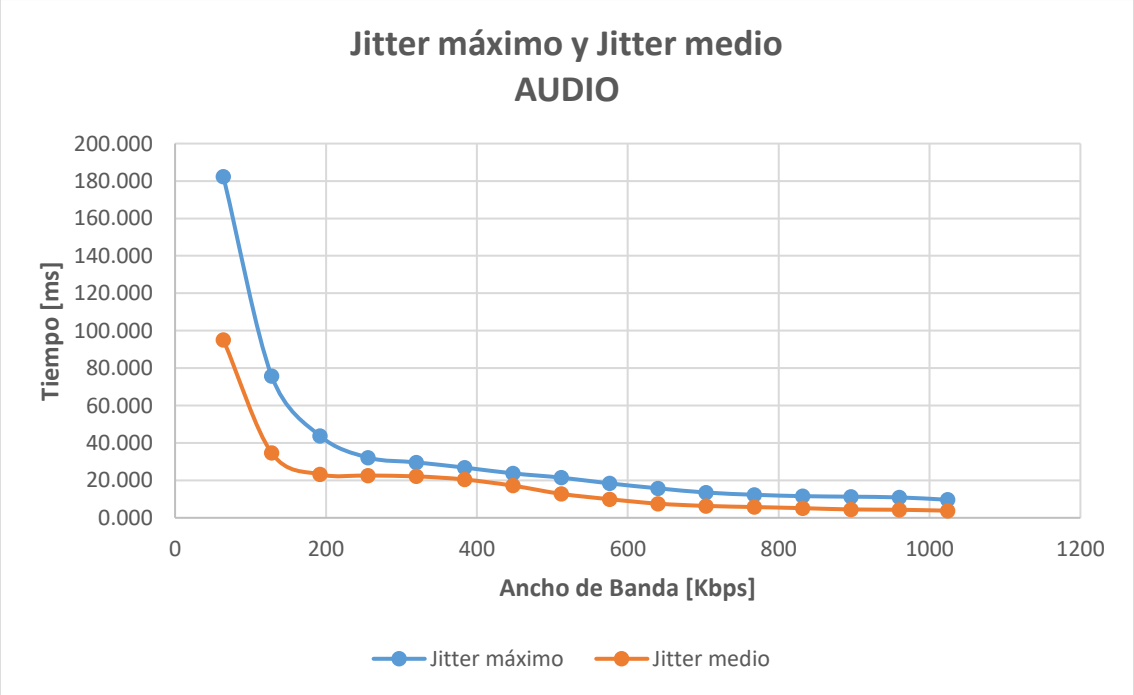

Figura 56. "Jitter" de audio. Fuente: Propia

Como es de esperarse para anchos de banda bajos el valor del "jitter" es mayor y va descendiendo a medida que aumenta el ancho de banda. Los valores promedios de "jitter" son tolerables a partir de 128 Kbps para que el audio en la

video llamada tenga buena calidad, ya que el valor del “jitter” debe ser menor a 100 ms para que no afecte su calidad.

El comportamiento del “jitter” de video con respecto al ancho de banda es similar al “jitter” de audio, pero estos valores son más elevados, ya que hay mayor cantidad de paquetes de video y demandan más ancho de banda. En la figura 57 se observa el comportamiento del “jitter” medio y del valor máximo de “jitter” alcanzado para este grupo de datos.

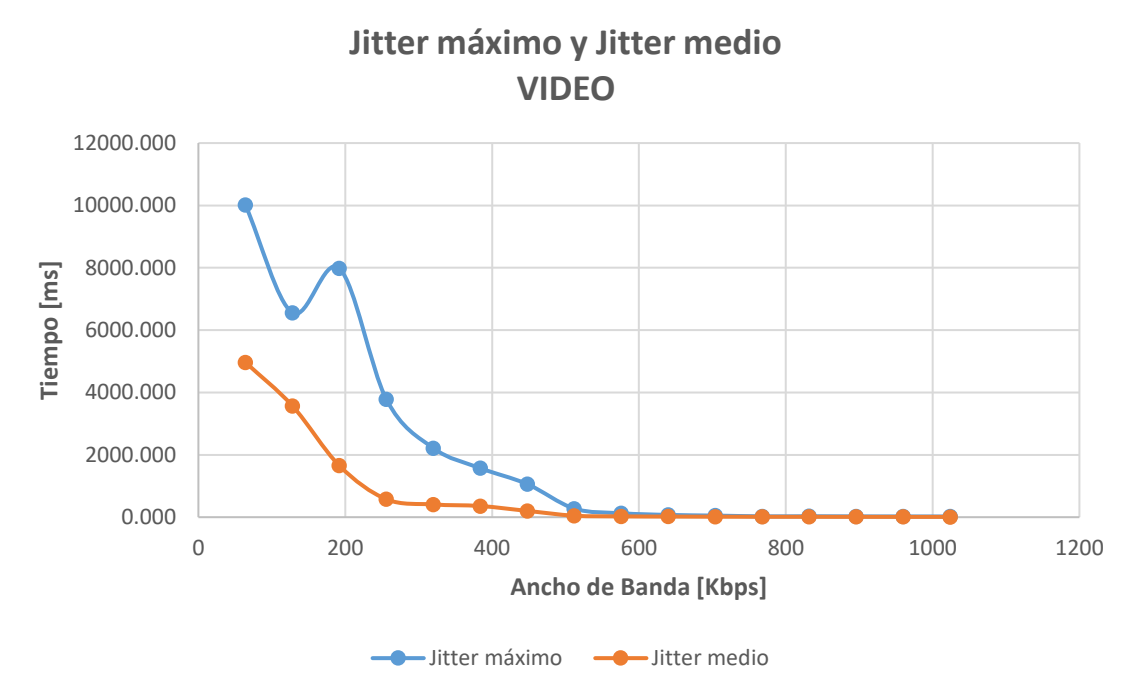

Figura 57. "Jitter" de video. Fuente: Propia

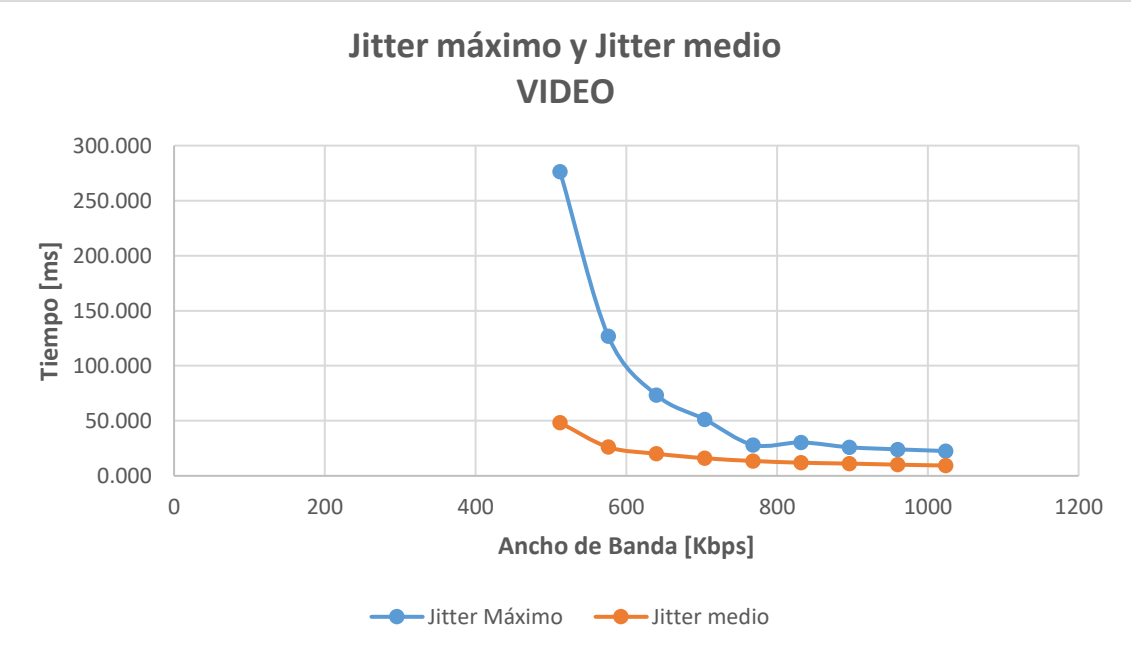

Figura 58. Jitter de video a partir de 512 Kbps. Fuente: Propia

Según la teoría, los valores de “jitter” deben ser menores a 100 ms para garantizar la calidad del servicio de video llamada. Para evidenciar este comportamiento, se grafican los datos a partir de un ancho de banda de 512Kbps, como se observa en la figura 58. En este caso los valores aceptables de “jitter” se pueden ver a partir de un ancho de banda de 640 Kbps.

4.2.3.4. Análisis de las pruebas

En la tabla 7 se muestran los datos de ancho de banda obtenidos en cada prueba, los cuales dependen de los valores aceptables de cada parámetro para brindar una buena calidad de servicio a los usuarios.

Parámetros de Calidad	**Audio (Kbps)**	**Video (Kbps)**
Retardo	64	
Perdida de Paquetes	256	384
"Jitter"	128	640

Tabla 7. Anchos de banda según los parámetros de calidad

Además de los datos analizados, se tiene en cuenta la calidad de la video llamada como tal, en donde la calidad de audio es aceptable para valores de ancho de banda relativamente bajos en comparación con la calidad de video, ya que esta mejora al aumentar el ancho de banda como se observa en la figura 59 y figura 60.

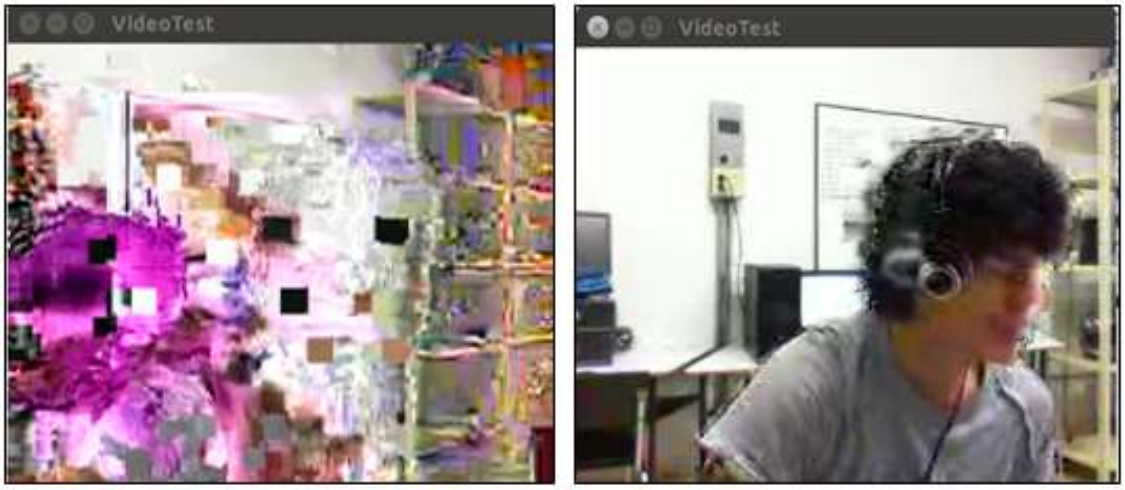

Figura 59. Calidad de video para un ancho de banda de 64 y 256 Kbps. Fuente: Propia

Figura 60. Calidad de video para un ancho de banda de 512 y 640 Kbps. Fuente: Propia

De acuerdo con estos datos, para tener una buena calidad de audio, el ancho de banda debe ser mayor a 256 Kbps y para video el ancho de banda necesario debe ser mayor a 640 Kbps. Por lo cual, el ancho de banda que cumple con las condiciones de los parámetros de calidad para brindar una buena calidad de servicio de video llamada es de 640 Kbps.

5. CONCLUSIONES Y TRABAJOS FUTUROS

El presente trabajo de grado enfoco sus esfuerzos en analizar la calidad de servicio (QoS) de una video llamada implementada sobre una red IMS virtualizada. Para ello se estableció un modelo en el cual se estudiaron los parámetros de calidad relacionados con el servicio de video llamada, determinando los parámetros más significativos que permiten medir la calidad de dicho servicio. Además, se describió la infraestructura general de la red especificando el hardware, el software y la topología utilizada. También, se realizó una caracterización de la red, analizando la escalabilidad tanto vertical como horizontal para verificar la eficiencia y el rendimiento que esta tiene al momento de implementar un servicio sobre la red IMS. Por último, se estructuraron los diferentes tipos de escenarios y casos de estudio de una forma eficiente y entendible, abordando los parámetros y características que componen la red.

En este capítulo se presentan las conclusiones, aportes adicionales y posibles trabajos futuros de investigación que pueden desprenderse del trabajo de grado realizado, con el fin de dar continuidad a esta línea de investigación.

5.1. CONCLUSIONES

A partir del trabajo desarrollado y de la experiencia adquirida, teniendo en cuenta que las pruebas se realizaron en un entorno ideal, se formulan las siguientes conclusiones aplicables al contexto planteado:

- El modelo propuesto permitió evaluar la calidad del servicio de video llamada sobre una red IMS virtualizada, al determinar qué tipo de escalabilidad es la apropiada para verificar la eficiencia y el rendimiento de Open Core IMS y qué parámetros de calidad son los adecuados para el análisis de dicho servicio.

- La variable que más afecta el rendimiento de Open IMS Core es el número de usuarios, ya que a medida que esta variable aumenta afecta el tiempo de respuesta y la perdida de información. De la misma forma se puede observar que el promedio de concurrencia de conexiones (parámetro lambda) también afecta el rendimiento ya que este se satura a medida que más usuarios desean conectarse en un determinado tiempo.

- La selección de los parámetros de calidad fue adecuada ya que por medio del análisis de cada uno de ellos en las pruebas realizadas permitieron determinar los valores que brindan una buena calidad de servicio en la red IMS. En este caso el valor conveniente de ancho de banda en donde se obtienen valores aceptables de “jitter”, perdida de

paquetes y retardo y que proporciona una buena calidad de video para el usuario es de 640 Kbps.

- El análisis de escalabilidad de la red permitió identificar la eficiencia en cuanto a tiempo de respuesta y pérdida de paquetes, además del comportamiento del Core IMS al aumentar el número de usuarios y el promedio de establecimiento de conexiones, analizando tanto el escalamiento vertical como el horizontal. En este caso el escalamiento horizontal es el más adecuado, ya que permite dividir las peticiones entrantes a la aplicación en diferentes Core IMS atendiendo una mayor cantidad de usuarios, brindando mejores tiempos de respuesta y disminuyendo perdidas de paquetes en comparación con el escalamiento vertical.

- Es indispensable contar con una red capaz de soportar una gran cantidad de usuarios, ya que una red de telecomunicaciones debe ser escalable, razón por la cual el análisis de escalabilidad horizontal permitió determinar que dicha escalabilidad soporta y mejora el rendimiento al incrementar el número de usuarios en la red IMS.

- En cuanto a la calidad de la video llamada, se observó que el ancho de banda es un parámetro indispensable para la calidad de dicho servicio, ya que a medida en que aumentaba el valor del ancho de banda, tanto el audio como el video mejoraba. En el caso del audio se necesitaban un ancho de banda relativamente bajo en comparación con el del video, ya que cuando la calidad del audio era estable, la calidad del video continuaba presentando algunas interferencias como distorsión, congelamiento del video y falta de sincronización con el audio.

- En los casos de estudio planteados se realizaron varias pruebas en diferentes condiciones y ambientes obteniendo resultados favorables y desfavorables, lo cual permitió validar el modelo de QoS propuesto para el servicio de video llamada.

5.2. APORTES ADICIONALES

A través del presente trabajo de grado se lograron aportes adicionales al logro de los objetivos definidos en el anteproyecto, al proveer una guía de instalación de los diferentes ambientes y herramientas utilizadas y de los componentes desarrollados para la ejecución de las pruebas.

Se documentó una guía de instalación clara y sencilla de seguir para la instalación de Open IMS Core, en donde se especifican todos los comandos y dependencias que se deben ejecutar para el correcto funcionamiento de la plataforma. Además se explica el proceso de instalación de Fokus Monster, el

cual es el cliente IMS utilizado para la ejecución de las pruebas, así como de Wondershaper, software utilizado para limitar el ancho de banda de la red.

Por otra parte, se desarrollaron varios componentes utilizados para la ejecución de las pruebas. Se desarrolló un script para la creación masiva de usuarios en Open IMS Core, un Simulador de Mensajes SIP utilizando el lenguaje de programación Python, capaz de enviar y responder peticiones SIP provenientes del servidor (Core IMS). Además, este simulador calcula los promedios de los tiempos de respuesta y el número de paquetes recibidos en cada conexión, almacenados en una base de datos. También se desarrolló un balanceador de carga, utilizando el mismo lenguaje de programación, el cual distribuye las peticiones del simulador de mensajes entre los diferentes Core IMS asociados al balanceador. La documentación de los componentes y la guía de instalación del Core de Open IMS se consignan en los anexos de este documento.

Además, con base en los resultados obtenidos en este trabajo de investigación se generó un artículo, el cual está en proceso de revisión en la revista Sistemas y Telemática (S&T) de la Universidad ICESI, indexada en Publindex en la categoría C.

5.3. TRABAJOS FUTUROS

A partir de los resultados obtenidos y las conclusiones formuladas, se observa que con el presente trabajo de grado se abren varias ramas de investigación haciendo uso de los equipos adquiridos por la plataforma Telco 2.0 de la Universidad del Cauca. A continuación, se plantean algunos de los posibles trabajos futuros:

- Implementar el servicio de video llamada sobre redes diferentes utilizando diversos clientes como X-Lite o Zoiper. Analizar su comportamiento, calidad del servicio y la escalabilidad de la red.

- Implementar un servicio diferente de telecomunicaciones sobre la misma red IMS, analizar su comportamiento, calidad del servicio y la escalabilidad de la red y comparar los datos con los resultados del estudio del servicio de video llamada, obtenidos en el presente trabajo de grado.

- Analizar el comportamiento de la red IMS al escalar el componente S-CSCF y el HSS. Establecer una video llamada y estudiar el efecto que tiene este componente en la QoS de dicho servicio.

- Realizar un estudio similar utilizando Clearwater, el cual es una implementación IMS diseñado para desplegarse masivamente sobre la nube, implementando virtualización de funciones de red (NFVs). Analizar su comportamiento, calidad del servicio y la escalabilidad de la red,

comparar los datos con los resultados del estudio del servicio de video llamada, obtenidos en el presente trabajo de grado y establecer qué tipo de entorno es el más óptimo para el servicio de video llamada.

REFERENCIAS BIBLIOGRAFICAS

[1] H. Nemati, A. Singhvi, N. Kara y M. El Barachi, «Adaptive SLA-based Elasticity Management Algorithms for a Virtualized IP Multimedia Subsystem,» *Cloud Computing Systems, Networks, and Applications,* p. 1, 2014.

[2] «Estudio Integral de Redes de Nueva Generación y Convergencia,» *Comisión de Regulación de Telecomunicaciones (CRT) - República de Colombia,* 2007.

[3] E. Villar y J. Gómez, «Virtualización de servidores de telefonía IP en GNU/Linux,» *Universidad de Alemania,* pp. 23, 37, 2010.

[4] M. Navarro, Y. Donoso y V. Rodríguez, «An IMS Architecture with QoS Parameters for Flexible Convergent Services,» *IEEE,* pp. 1-2, 2010.

[5] D. Blandón, Y. Díaz, F. G. Guerrero, J. C. Cuellar, A. Navarro C y C. Ochoa A, «Medición de la calidad del servicio en Redes de Próxima Generación en Colombia: Procedimientos de medida de calidad de servicio en redes de próxima generación,» *Centro de Investigación de las Telecomunicaciones - CINTEL,* pp. 55-57, 2010.

[6] «Recomendación UIT-T E. 800, Calidad de los servicios de telecomunicación: conceptos, modelos, objetivos, planificación de la seguridad de funcionamiento – Términos y definiciones relativos a la calidad de los servicios de telecomunicación,» *Unión Internacional de Telecomunicaciones,* p. 9, 2008.

[7] M. Liangli , C. Yanshen, S. Yufei y W. Qingyi , «Virtualization Maturity Reference Model for Green Software,» *International Conference on Control Engineering and Communication Technology,* pp. 1, 2, 2012.

[8] «Virtualización de servidores,» [En línea]. Available: http://www.integracanarias.com/blog/38-virtualizacion-servidores-caracteristicas-beneficios. [Último acceso: 18 Octubre 2016].

[9] «Maquina Virtual (VM),» [En línea]. Available: http://searchdatacenter.techtarget.com/es/definicion/Copy-of-virtual-machine-VM. [Último acceso: 18 Octubre 2016].

[10] J. V. Ros Solís , «Análisis de un plan de continuidad de servicios clave mediante infraestructuras virtualizadas privadas,» *Universidad Politécnica de Valencia ,* pp. 18-22, 2013.

[11] «Virtualización,» [En línea]. Available: http://www.vmware.com/latam/solutions/virtualization.html. [Último acceso: 18 Octubre 2016].

[12] «Recomendación UIT-T Y.2001, Redes de la próxima generación – Marcos y modelos arquitecturales funcionales,» *Unión Internacional de Telecomunicaciones,* p. 2, 2004.

[13] W. F. Sanchez Pacheco, «Diseño e implementación de una solución de interconexión de redes NGN mediante el protocolo SIP,» *Pontificia Universidad Javeriana,* pp. 16-18, 2013.

[14] P. Podhradský, J. Dúha, P. Trúchly y J. Blichár, «NGN (Next Generation Networks) – Selected Topics,» *Czech Technical University in Prague,* pp. 11-18.

[15] T. Russell, «The IP Multimedia Subsystem (IMS), Session Control and Other Network Operations,» *McGRAW - Hill Communications,* 2008.

[16] B. Morata Gonzáles, «Estudio de la tecnología IMS y diseño de una solución de telefonía multimedia,» *Universidad Politécnica de Madrid,* 2012.

[17] «Introduction to the IP Multimedia Subsystem (IMS),» *F5 Networks,* 2016.

[18] I. Vidal Fernández, «Configuración de transporte multicast IP con calidad de servicio en arquitecturas de red con plano de control IMS,» *Universidad Carlos III de Madrid,* 2008.

[19] L. A. Osorio Molina y L. E. Suárez de Aquiz, «Estado del arte de IP Multimedia Subsystem (IMS),» *Universidad Pontificia Bolivariana,* 2009.

[20] J. A. Hurtado, «Session Initiation Protocol SIP,» *Universidad del Cauca.*

[21] M. Ramos Benito, «Desarrollo de una aplicación de audioconferencia basada en Multicast para escenarios de red con soporte IMS,» *Universidad Carlos III de Madrid,* 2009.

[22] F. R. Monegui Rosas y I. Suarez Laguna, «Modelo de seguridad IMS para el aseguramiento de plataforma y servicios - Fase Conceptual,» *Universidad Católica Andrés Bello,* 2012.

[23] «Calidad de Servicio,» [En línea]. Available: https://www.ecured.cu/Calidad_de_servicio. [Último acceso: 18 Octubre 2016].

[24] S. García Subirón, «Calidad de servicio en redes NGN (Quality of service in next generation networks),» *Universitat Politécnica de Catalunya,* 2012.

[25] «Recommendation ITU-T F.700. Non-Telephone Telecommunication Services. Audiovisual services - Framework Recommendation for multimedia services,» *International Telecommunication Union,* 2000.

[26] M. F. B. Almeida y C. M. L. Cabrera, «Evaluación de los Parámetros que Afectan la Calidad de Servicio».

[27] J. F. Kurose y K. W. Ross, Computer Networking. A Top-Down Approach, Estados Unidos de America: Pearson Education, 2013.

[28] «Recomendación UIT-T G. 114, Conexiones y circuitos telefónicos internacionales – Recomendaciones generales sobre la calidad de transmisión para una conexión telefónica internacional completa. Tiempo de transmisión en un sentido,» *Unión Internacional de Telecomunicaciones,* 2003.

[29] «QoS Quality Of Sevice VoIP,» VoIPForo, [En línea]. Available: http://www.voipforo.com/QoS/QoS_Jitter.php. [Último acceso: 1 Noviembre 2016].

[30] S. C. Rodríguez, R. M. Gómez, E. . D. Laurencio y P. M. P. , «Quality of experience in the transmission of video over IP,» *7mo Congreso Internacional de Tecnologías y Contenidos Multimedia,* 2016.

[31] A. Cánovas Solbes, «Diseño y Desarrollo de un Sistema de Gestión Inteligente integrado de s ervicios de IPTV estándar, estereoscópico y HD basado en QoE,» *Universidad Politecnica de Valencia,* pp. 7,8, 2013.

[32] «Recomendación UIT-T G.1020 Definición de parámetros de calidad de funcionamiento para aplicaciones de voz y otras aplicaciones en la banda vocal que utilizan redes del protocolo Internet,» *Sector de Normalización de las Telecomunicaciones de la UIT,* pp. 7-10, 2006.

[33] «Recommendation ITU-T Y.1540, Internet protocol data communication service – IP packet transfer and availability performance parameters,» *International Telecommunication Union,* 2016.

[34] «Recommendation ITU-T Y.1541, Network performance objectives for IP-based services,» *International Telecommunication Union,* 2011.

[35] «Calculating Bandwidth for Video Calls,» [En línea]. Available: https://andrewjprokop.wordpress.com/2013/10/24/calculating-bandwidth-for-video-calls/. [Último acceso: 2016 Noviembre 15].

[36] Cisco, «Voz sobre IP - Consumo de ancho de banda por llamada,» 19 Mayo 2008. [En línea]. Available: http://www.cisco.com/cisco/web/support/LA/7/73/73295_bwidth_consume.html. [Último acceso: 30 Enero 2017].

[37] G. ACT, «Consideraciones para implementación de videoconferencia via IP».

[38] «The HP Shop,» [En línea]. Available: http://www.thehpshop.com/switches/24108-hp-n-hi-5500-24g-4sfp-w2-0886111896394.html. [Último acceso: 27 Febrero 2017].

[39] «Hewlett Packard Enterprise,» [En línea]. Available: https://www.hpe.com/us/en/product-catalog/servers/virtual-connect/pip.hp-virtual-connect-flex-10-10d-module-for-c-class-bladesystem.5288599.html. [Último acceso: 28 Febrero 2017].

[40] H. P. Enterprise, «Data Sheet - HPE ProLiant BL460c Gen9 Server Blade,» 2016.

[41] H. P. Enterprise, «Data Sheet - HPE MSA 2040 Storage,» 2016.

[42] «VMWare - vSphere 5 Documentation Center,» [En línea]. Available: https://pubs.vmware.com/vsphere-50/index.jsp?topic=%2Fcom.vmware.vsphere.solutions.doc_50%2FGUID-64C8FAD4-9B84-4492-B86A-9C3F4A058808.html. [Último acceso: 1 Marzo 2017].

[43] «Cloud Aruba,» [En línea]. Available: http://kb.arubacloud.com/en/computing/access-to-cloud-servers/connecting-to-the-cloud-server-service-via-vmware-vsphere-client.aspx. [Último acceso: 1 Marzo 2017].

[44] «Finalmente, lanzamiento de Ubuntu 14.04 (Trusty Tahr),» Asociación Educativa LIGNUX, [En línea]. Available: https://lignux.com/finalmente-lanzamiento-de-ubuntu-14-04-trusty-tahr/. [Último acceso: 1 Marzo 2017].

[45] MySQL, «IMS Adoption and Communities Fueled by the Open IMS Core Project, Usinng MySQL for the Home Subscriber Server,» *MySQL in Telecommunications,* 2009.

[46] «Wireshark User's Guide,» [En línea]. Available: https://www.wireshark.org/docs/wsug_html_chunked/index.html. [Último acceso: 2 Marzo 2017].

[47] CCITT, Comite Consultivo Internacional Telegrafico y Telefonico, «Recomendación UIT-T F.720, Servicios de Videotelefonia – Generalidades,» *Unión Iinternacional de Telecomunicaciones,* 1992.

[48] «Open IMS Core,» [En línea]. Available: http://www.openimscore.org/. [Último acceso: 7 Marzo 2017].

[49] H. A. R. Sandoval, «Balanceador de carga de llamadas para sistemas de Call Center multicola implementados sobre la plataforma de software libre Asterisk,» *Pontificia Universidad Javeriana,* 2015.

[50] «Anaconda Documentation,» [En línea]. Available:

https://docs.continuum.io/anaconda/. [Último acceso: 8 Diciembre 2016].

[51] «Anaconda,» [En línea]. Available: https://www.anaconda.com/download/. [Último acceso: 7 Dieciembre 2016].

[52] «PIP,» [En línea]. Available: https://pip.pypa.io/en/stable/ . [Último acceso: 7 Diciembre 2016].

[53] «Twisted Matrix Labs,» [En línea]. Available: https://twistedmatrix.com/trac/. [Último acceso: 9 Diciembre 2016].

[54] «Scapy,» [En línea]. Available: https://scapy.readthedocs.io/en/latest/index.html. [Último acceso: 10 Diciembre 2016].

[55] O. S. Developer, «Data Modeler User's Guide,» 2009.

[56] «Open Source IMS CORE - Guia de instalación,» [En línea]. Available: http://www.openimscore.org/documentation/guia-de-instalacion/. [Último acceso: 17 Septiembre 2016].

ANEXO A. INSTALACION Y CONFIGURACION DEL SOFTWARE

A.1. OPEN IMS CORE

La instalación de Open IMS Core [48], requiere de ciertas características en cuanto al software y hardware para su correcto funcionamiento. Se debe contar con un sistema operativo Linux, disponibilidad de varias Gigabytes de RAM, varios CPUs/Cores, y disponer de aproximadamente 100 MBytes de espacio en el disco.

Para este trabajo de grado se instaló el sistema operativo Ubuntu 14.04 en una máquina virtual, configurado y actualizado previamente para ejecutar las dependencias y requisitos que el Core IMS necesita.

1. Requisitos previos

Primero se debe actualizar la lista de paquetes disponibles y sus versiones de Ubuntu con el siguiente comando:

#sudo apt-get update

Los requerimientos o dependencias de software necesarios para instalar Open IMS Core son los siguientes:

- **Flex**
  # apt-get install flex

- **Bison**
  # apt-get install bison

- **GCC**
  # apt-get install gcc
  # gcc --version (para verificar la versión del gcc instalado)

- **Make**
  # apt-get install make

- **Ant**
  # apt-get install ant

- **JDK 1.5**
  Verificar que java no se ha instalado con el comando:
  # java -version

Si este comando regresa un mensaje como el siguiente "*The program java can be found in the following packages*", significa que java no está instalado, de modo que ejecutaremos la siguiente línea de código:

 # apt-get install default-jdk

- **MySQL**
  Se instala el módulo MySQL para la persistencia de los datos del Core.

  # apt-get update
  # apt-get upgrade
  # apt-get install mysql-server mysql-client

  Después de lanzar el anterior comando, aparece una ventana en la cual se pide ingresar una contraseña de administrador (root) para gestionar la base de datos.

  Para verificar que se instaló correctamente MySQL, se ingresa el siguiente comando para ingresar al gestor de base de datos desde consola.

  # mysql -u root –p

  Seguido de la contraseña que anteriormente se digito en la instalación, en este caso es "admin".

- **Libmysql**
  # apt-get install libmysqlclient-dev

- **libxml2**
  # apt-get install libxml2-dev

- **bin**
  # apt-get install bind9

- **Subversion**
  # apt-get install subversión

2. Obtener el código fuente

Para obtener el código fuente se debe tener instalada la anterior herramienta que es subversión. El código esta pre-configurado para trabajar desde una ruta de ficheros, para esto se deben crear los siguientes directorios:

 # mkdir /opt/OpenIMSCore/
 # mkdir /opt/OpenIMSCore/ser_ims

 # mkdir /opt/OpenIMSCore/FHoSS

Ahora se debe descargar el código fuente de Open IMS, para ello se ingresa a la carpeta *"/opt/OpenIMSCore"*, y se baja la última versión de Open IMS con la ayuda de subversión.

 # sudo svn checkout
https://svn.code.sf.net/p/openimscore/code/ser_ims/trunk ser_ims

 # sudo svn checkout
https://svn.code.sf.net/p/openimscore/code/FHoSS/trunk FHoSS

3. Compilar código fuente

Se debe ingresar al directorio *"/opt/OpenIMSCore/ser_ims*", y se ejecuta el siguiente comando:

 # make install-libs all

Si algo falla, probablemente no se cuenta con todos los requisitos previos. Para un correcto funcionamiento, fue pertinente instalar los siguientes componentes que la misma compilación exige.

 # apt-get install libxml2-dev
 # apt-get install libcurl4-openssl-dev
 # apt-get install libmysqlclient-dev

Después de instalar las anteriores dependencias ya se puede lanzar el comando para compilar los componentes de esta sección.

 # make install-libs all

Ahora, en la dirección *"/opt/OpenIMSCore/FHoSS"* se corren los siguientes comandos:

 # ant compile
 # ant deploy

Si no funcionan estos comandos se debe configurar el JAVA_HOME, como se observa a continuación.

3.1. Configurar el path de JAVA_HOME

Para encontrar la dirección de instalación de java ingresamos el comando:

 # whereis java

Aquí aparecen los diferentes directorios en donde podría estar instalado java. Se usa la primera opción.

java: ***/usr/bin/java*** */usr/bin/X11/java /usr/share/java /usr/share/man/man1/java.1.gz*

Para encontrar la verdadera dirección de java se ingresa el siguiente comando:

 # ls -l /usr/bin/java

Aquí se muestra una dirección alternativa del lugar donde java está instalado.

 # ls -l /etc/alternatives/java

Arrojando la verdadera dirección de java,

/usr/lib/jvm/java-7-openjdk-amd64/jre/bin/java

Ahora para configurar el path de JAVA_HOME de todos los usuarios de Ubuntu, se edita el script profile.

 # gedit /etc/profile

Se declara la variable de entorno JAVA_HOME al finalizar el script de la siguiente manera:

export JAVA_HOME=/usr/lib/jvm/java-7openjdk-amd64/jre
export PATH=$PATH:/usr/lib/jvm/ java-7openjdk-amd64/jre/bin

Para aplicar los cambios, se ejecuta el siguiente comando:

 # source /etc/profile

4. Copiar ficheros

Ahora se copian algunos archivos al directorio de trabajo *"/opt/OpenIMSCore/"* por cuestiones de comodidad más que por necesidad.

 # cp /opt/OpenIMSCore/ser_ims/cfg/ /opt/OpenIMSCore/*

5. Configurar base de datos

Se configura la base de datos donde está la persistencia y los datos necesarios para que el Core funcione correctamente.

 # mysql -u root -p < /opt/OpenIMSCore/ser_ims/cfg/icscf.sql
 # mysql -u root -p < /opt/OpenIMSCore/FHoSS/scripts/hss_db.sql

 # mysql -u root -p < /opt/OpenIMSCore/FHoSS/scripts/userdata.sql

6. Configurar dominio y dirección IP

En la dirección *"opt/OpenIMSCore/ser_ims/cfg"*, se ejecuta el siguiente comando.

 # bash configurator.sh

Aquí pide llenar los siguientes campos:

Domain Name: *open-ims.test*
IP Adress: *10.55.0.39* (La dirección IP que se desea cambiar para todos los scripts y es la dirección IP de la máquina virtual).

Se presenta el siguiente mensaje y se digita la palabra *"all"* para que la configuración se aplique en todos los archivos.

File to change ["all" for everything, "exit" to quit]: ***all***
changing: ecscf.cfg icscf.cfg icscf_pg.sql icscf.sql icscf.thig.cfg icscf.xml lrf.cfg mgcf.cfg pcscf.cfg pcscf.xml persist_my.sql persist_pg.sql scscf.cfg scscf.xml TGPPGq.xml TGPPRx.xml trcf.cfg.

Para encontrar direcciones IP o cualquier palabra dentro de un archivo se puede buscar ejecutando el siguiente comando:

 # grep -lir "palabra_a_buscar"

También se debe cambiar la dirección IP del fichero *"hss.properties",* el cual se encuentra en la ruta *"/opt/OpenIMSCore/FHoSS/deploy".*

7. Configuración del DNS-Zone

Para configurar el DNS-Zone se debe copiar el fichero *"open-ims.dnszone"* en la dirección *"/etc/bind"* con el siguiente comando:

 # cp /opt/OpenIMSCore/ser_ims/cfg/open-ims.dnszone /etc/bind

En este archivo se debe cambiar la dirección *127.0.0.1* por la dirección de la máquina virtual que se esté manejando y el dominio *localhost* por *open-ims.test*.

Además, en el archivo *"named.conf.local"* del mismo directorio *(/etc/bind*), se adiciona las siguientes líneas de código:

```
zone "open-ims.test" {
        type master;
```

```
        file "/etc/bind/open-ims.dnszone";
};
```

8. Configuración del script resolv.conf

Este archivo sirve para cambiar los servidores de DNS que nuestro sistema utiliza para resolver el nombre de dominio. Para ello se edita el archivo *"/etc/resolv.conf"* con las siguientes líneas de código:

nameserver ***dirección_IP del Core IMS***
search ***open-ims.test***
domain ***open-ims.test***

Este archivo se sobreescribe, es decir, siempre se modifica automáticamente cuando la máquina cambia de red, conexión o se inicia el sistema operativo. Razón por la cual es necesario editar este archivo.

Para no estar en esta tarea tan tediosa de modificar este script, se configura el archivo "base" ubicado en la ruta *"/etc/resolvconf/resolv.conf.d"*, con lo siguiente:

nameserver ***dirección_IP***
search ***open-ims.test***
domain ***open-ims.test***

Nota: A pesar de que este archivo ayuda a configurar el *"resolv.conf"* siempre se debe verificar y cambiar los parámetros, para evitar fallas al ejecutar los servicios IMS.

Se reinicia el servidor BIND con el siguiente comando, para que se guarde la configuración previamente realizada.

```
# /etc/init.d/bind9 restart
```

9. Configuración del Core IMS

Para configurar el Core IMS, se debe tener MySQL y el servidor DNS funcionando correctamente. Se copian los archivos que a continuación se describen con los siguientes comandos, desde el directorio *"/opt/ OpenIMSCore/"*.

```
# cp ser_ims/cfg/*.cfg .
# cp ser_ims/cfg/*.xml .
# cp ser_ims/cfg/*.sh .
```

10. Ejecución de los servicios IMS

Se lanzan los servicios con los siguientes comandos.

 # cd /opt/OpenIMSCore/
 # ./pcscf.sh &
 # ./icscf.sh &
 # ./scscf.sh &

 # cd FHoSS/deploy/
 # ./startup.sh &

Si el anterior paso falla, se debe revisar que la variable de entorno JAVA_HOME esté correctamente exportado y/o modificar el script que se acaba de intentar iniciar, es decir, cambiar en el archivo *"startup.sh",* ubicado en la carpeta *"FHoSS/deploy/"*, la variable JAVA_HOME por la dirección verdadera.

/usr/lib/jvm/java-7-openjdk-amd64/jre/bin/java

Se lanzan nuevamente los servicios y se verifican que estén funcionando correctamente. Para ello, en un navegador web, se ingresa a la URL *http://direccion_IP:8080*, en donde se digita el usuario y la contraseña de la interfaz FHoSS (figura 61). Por defecto los valores son:

User Name: ***hssAdmin***
Password: ***hss***

Si no se puede ingresar a la interfaz web es porque se presentó algún error en la ejecución de alguno de los pasos anteriores, por lo cual se debe verificar que todos los comandos fueron lanzados correctamente.

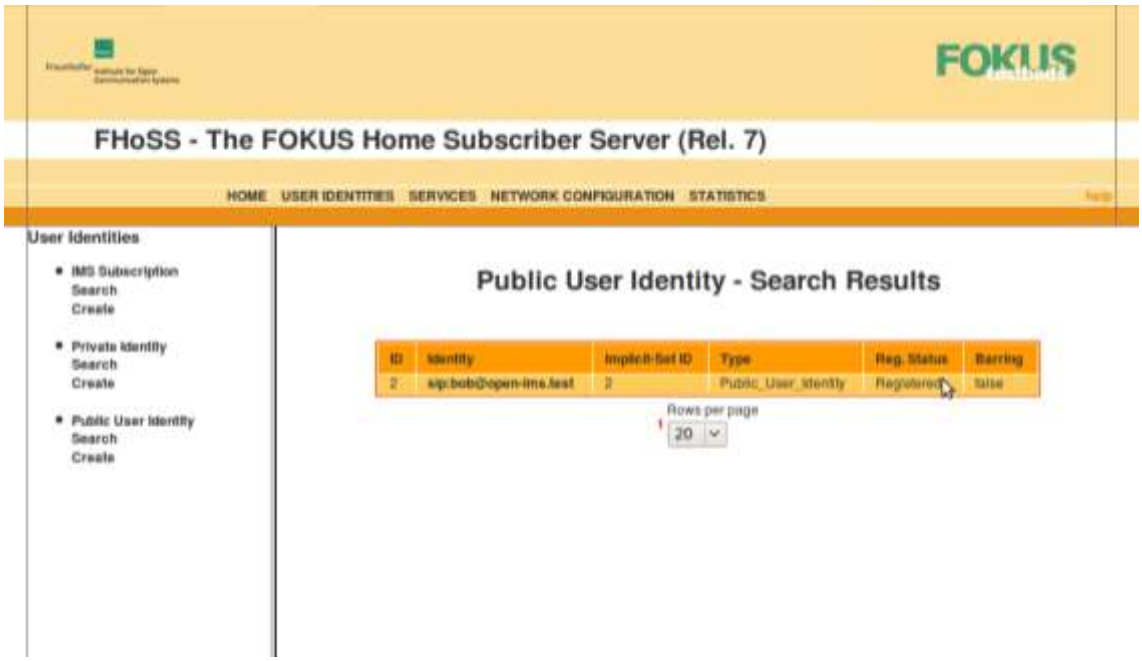

Figura 61. Interfaz web FHoSS. Fuente: Propia

A.2. INSTALACIÓN MONSTER FOKUS

Para la instalación de este cliente IMS, se debe copiar el archivo comprimido *"myMONSTER-TCS_Linux32_v0_9_25_tar.gz" en la carpeta "/usr/local/src/"*, con la siguiente línea de código:

 # cp /home/telco/Descarga/myMONSTER-TCS_Linux32_v0_9_25_tar.gz /usr/local/src

En la carpeta *"/usr/local/src"*, se descomprime el archivo con el siguiente comando:

 # tar -xzvf myMONSTER-TCS_Linux32_v0_9_25_tar.gz

Se deben instalar los siguientes plugins y dependencias, para el correcto funcionamiento del cliente IMS.

 # sudo add-apt-repository ppa:mc3man/trusty-media
 # sudo apt-get update
 # sudo apt-get install gstreamer0.10-ffmpeg
 # sudo apt-get install libexosip2-de
 # sudo apt-get install libgtk2.0-dev
 # sudo apt-get install libxml2-dev
 # sudo apt-get install libcurl4-openssl-dev
 # sudo apt-get install libgstreamer0.10-0

 # sudo apt-get install libgstreamer-plugins-base0.10-dev
 # sudo apt-get install gstreamer0.10-plugins-base
 # sudo apt-get install gstreamer0.10-plugins-good
 # sudo apt-get install gstreamer0.10-plugins-bad
 # sudo apt-get install gstreamer0.10-plugins-ugly
 # sudo apt-get install libav-tools
 # sudo apt-get install libvlc-dev
 # sudo apt-get install vlc
 # sudo apt-get update

Con el siguiente comando se ejecuta FOKUS MONSTER.

 # cd /usr/local/src/monster-0.9.25/
 # ./monster

Una vez instalada la aplicación, se debe configurar los clientes IMS como se observa en la figura 62.

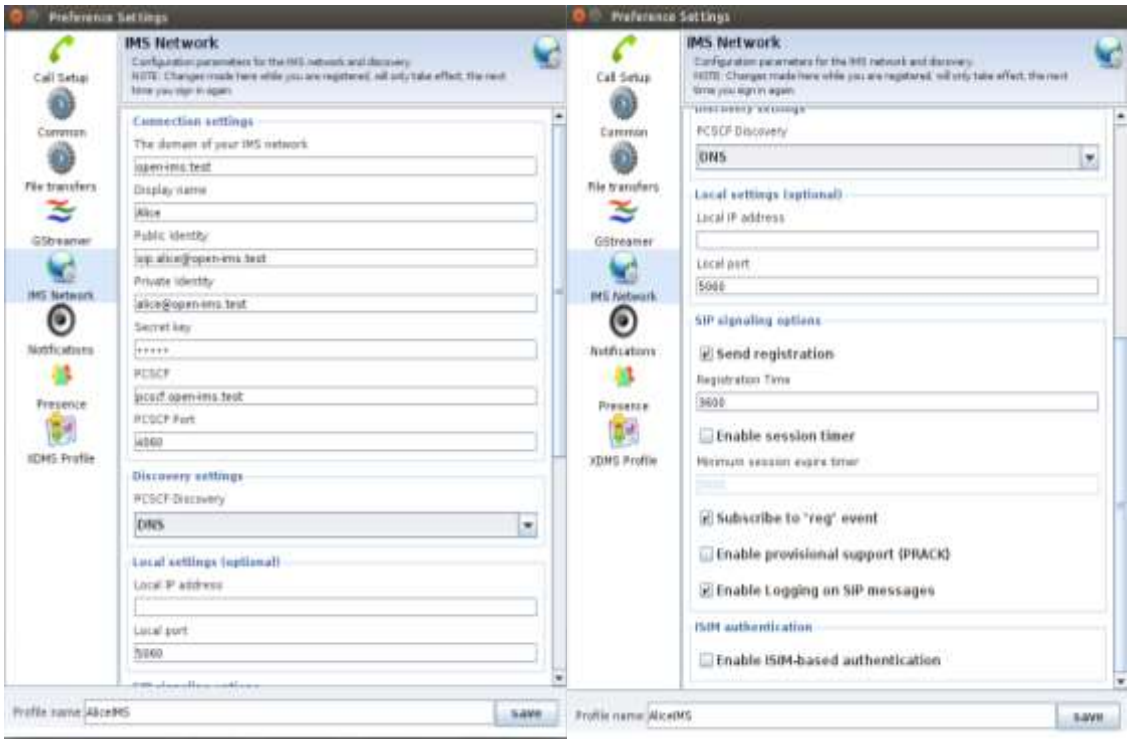

Figura 62. Configuración del cliente Fokus Monster. Fuente: Propia

A.3. INSTALACION WONDERSHAPER

Wondershaper es un limitador de ancho de banda para Linux, el cual funciona y se instala de una manera muy sencilla. Para su instalación, se abre el terminal, se ingresa como administrador (root) y se digita el siguiente comando:

$ sudo apt-get install wondershaper

Una vez instalado, se verifica la interfaz de red que se está utilizando (eth0, eth1, wlan0, etc.).

```
$ ifconfig
```

Con el siguiente comando se limita el ancho de banda tanto del enlace de bajada como el de subida. Es importante anotar que los valores que se ingresan deben estar en Kbps.

$ sudo wondershaper ***Intefaz_de_red Downlink Uplink***

Por ejemplo,

$ sudo wondershaper eth0 1500 1000

El anterior comando limita el ancho de banda de la interfaz de red eth0, en donde sus valores son 1.5 Mbps de Downlink y 1 Mbps de Uplink.

Para eliminar los ajustes previos, se digita el siguiente comando:

$ sudo wondershaper clear eth0

ANEXO B. COMPONENTES DE PRUEBA

B.1. TIEMPO DE CONCURRENCIA ENTRE PETICIONES CONSECUTIVAS SEGÚN LA DISTRIBUCION POISSON

En el presente anexo se explica el tiempo de concurrencia entre peticiones consecutivas que es necesario para el funcionamiento de uno de los módulos del simulador de mensajes SIP.

El tiempo de concurrencia entre peticiones consecutivas es modelado mediante la distribución de Poisson, ya que expresa la probabilidad de que ocurran un determinado número de eventos en un lapso de tiempo establecido.

Para generar los tiempos entre llamadas se utiliza la herramienta Matlab. Este software no tiene una función matemática que simule directamente la distribución Poisson, por lo cual se realizan una serie de comandos que permiten simular dicha distribución.

Primero se asignan los valores para el numero promedio de llamadas por minuto (Lambda, λ) y el número de llamadas a generar (N). Se utiliza la función *rand* para generar N valores aleatorios entre 0 y 1, simulando una distribución uniforme.

```
lambda=5/6;          %50 llamadas por minuto
X=rand(1, 5000);     %Distribución Uniforme entre 0 y 1
```

Se obtiene una distribución exponencial en función de λ de la siguiente manera:

```
Y= - log(X);         %Distribución exponencial con λ = 1
Z=(1/lambda)*Y;      %Distribución exponencial en función de λ
```

La distribución exponencial modela el tiempo entre 2 peticiones consecutivas, estos tiempos se reparten según la distribución Poisson, en este caso se distribuyen los tiempos para generan 50 peticiones en 1 minuto.

```
W=cumsum(Z);         %Distribución Poisson
```

Los valores de la matriz Z se exportan en un archivo .csv para que puedan ser usados por el simulador de mensajes SIP. Se utiliza esta matriz ya que se desea conocer los tiempos de ocurrencia entre peticiones consecutivas.

*csvwrite('**NombreDeArchivo.csv**', Z);*

Este archivo contiene 5000 tiempos, valores suficientes para el número de conexiones de usuarios a simular. Este proceso se repite para diferentes valores de lambda (50, 100, 150, 200, 250, 500, 1000) obteniendo para cada

uno de estos valores un archivos con extensión .csv, que son usados en el primer módulo del simulador de mensajes SIP.

B.2. CREACION DE USUARIOS EN OPEN IMS CORE

En este anexo se explica el script utilizado para la creación masiva de usuarios en Open IMS Core. Este archivo es una extensión .sh ubicado en el directorio *"/opt/OpenIMSCore".* En la figura 63 se observa el código utilizado, en donde se almacena el número de usuarios deseados, usando el script de Open IMS Core (./add-imscore-user_newdb.sh).

scriptCrearUsuarios.sh

```
#SCRIPT PARA LA CREACION MASIVA DE USUARIOS
scrip 200 240
numUsuarios=$1
timeCreation=$2

echo $numUsuarios
echo $timeCreation
value=$(($timeCreation/$numUsuarios))
residuo=$(echo "scale=2; $timeCreation/$numUsuarios" | bc)

if ! $(ping -c 1 10.55.0.36 > /dev/null); then
    echo La máquina 10.55.0.36 no responde
else
  i=1
  while [ $i -lt $(($numUsuarios+1)) ]; do
      nombreUsuario=$(($i+100))
      echo $nombreUsuario
      sleep $residuo
        ./add-imscore-user_newdb.sh -u "$nombreUsuario" -a
        echo "\n"
      i=$(($i+1))
  done
fi
```

Figura 63. Script para la creación masiva de usuarios. Fuente: Propia

Antes de ejecutar el script se debe modificar el archivo "./add-imscore-user_newdb.sh", para autorizar en la base de datos, la creación de usuarios sin necesidad de ingresar la contraseña cada vez que se desea registrar un usuario, ya que se registraran un número elevado de clientes en el Core IMS. En la figura 64 se observa el cambio que se debe realizar.

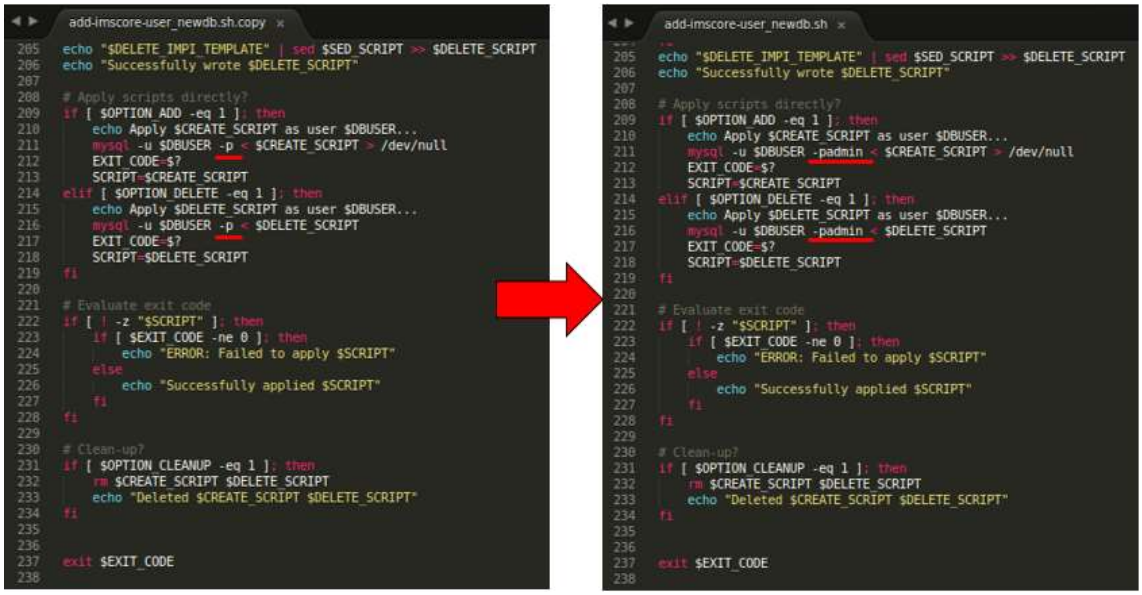

Figura 64. Modificación del script de Open IMS Core. Fuente: Propia

Una vez realizado este cambio, se procede a ejecutar el script desarrollado. Para ello se ingresa el número de usuarios que se desean crear y el tiempo que esta creación debe tomar. En este caso se desean crear 200 usuarios en 4 minutos (240 segundos) como se observa en la figura 65.

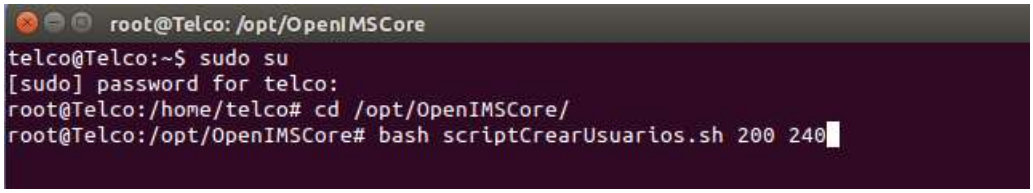

Figura 65. Comando para ejecutar el script de creación de usuarios. Fuente: Propia

Para verificar la creación de los usuarios, se ingresa a la interfaz web del FHoSS, como se observa en la figura 66. Para el desarrollo de las pruebas se crearon 2000 usuarios.

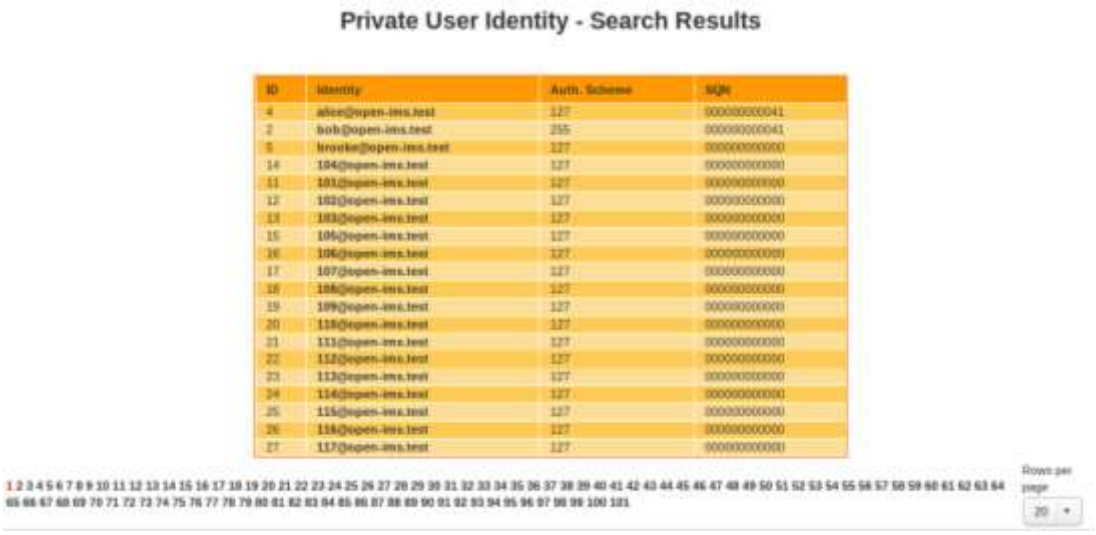

Figura 66. Lista de usuarios registrados en Open IMS Core. Fuente: Propia

B.3. SIMULADOR DE MENSAJES SIP

Para el estudio de los tiempos de respuesta y perdida de paquetes de Open IMS Core es preciso desarrollar un simulador de mensajes SIP capaz de enviar y recibir peticiones hacia el Core. Este simulador se adapta tanto a las pruebas hechas para el escalamiento vertical como para el horizontal acoplándose con el balanceador de carga, con el objetivo de analizar el desempeño del Core en cuanto a tiempos de respuesta y número de paquetes perdidos por falta de capacidad de procesamiento.

B.3.1. Software y lenguajes de programación

El simulador de llamadas se ejecuta sobre el sistema operativo de Ubuntu 14.04, en donde se configuran las siguientes dependencias.

- Python

El simulador de mensajes SIP se desarrolla usando el lenguaje de programación Python ya que existe una gran cantidad de librerías que pueden ser usadas para diferentes propósitos en múltiples plataformas. Además, es un lenguaje simple, flexible, multiparadigma que soporta orientación de objetos y utiliza una sintaxis muy limpia, lo que permite que el código sea legible.

- Anaconda

Anaconda es una distribución de Python, gestor de paquetes y de entornos que posee cerca 720 paquetes al alcance de los usuarios [50]. Anaconda permite una fácil gestión de paquetes y librerías con unos pocos pasos para su instalación, actualización y eliminación. Este software permite gestionar las librerías en el ordenador donde será instalado el simulador de paquetes SIP, así como el acceso a las librerías desde cualquier lugar dentro de la distribución de archivos y documentos del ordenador.

Para obtener Anaconda, se descarga de la página oficial el instalador del software [51]. En la carpeta donde se encuentra el archivo se ejecuta el siguiente comando.

#bash Anacondax-x.x.x-Linux-x86_64.sh

Al ejecutar el comando se despliegan algunas configuraciones para el proceso de instalación, en este caso se instala todo por defecto. En algunas ocasiones es necesario reiniciar el equipo para que los nuevos cambios fueran visibles. Para comprobar que se Anaconda está instalado correctamente se digita cualquiera de los siguientes comandos.

#conda list

#python

Con el primer comando se despliega la lista de paquetes instalados y sus versiones y el segundo muestra información sobre de la versión instalada.

Las librerías usadas para el simulador de mensajes SIP son gestionadas por Anaconda, pero en ocasiones anaconda no cuenta con ciertas librerías por lo cual es necesario instalarlas, para ello se utiliza el instalador de paquetes PIP. Este instalador es muy útil para descargar librerías de software escritos en Python. PIP ya se encuentra instalado por defecto en versiones de Python 2 mayores a 2.7.9 o Python 3 mayores a 3.4 [52]. Si PIP no se encuentra instalado, solo es necesario ejecutar los siguientes comandos.

#apt-get install Python-pip
#pip install -U pip

- Twited

Twisted Matrix es un framework de red para programación dirigida por eventos, escrito en Python y de código abierto, el cual soporta un gran número de arquitecturas y protocolos [53]. Esta librería es la base del simulador de mensajes SIP ya que permite escuchar los puertos por donde se establece la conexión del usuario (host del simulador) y Open IMS Core. Además, envía los paquetes de respuesta hacia el servidor (Open IMS Core).

Con ayuda de PIP se puede instalar la librería Twisted y los componentes necesarios para que esta funcione correctamente.

#pip install zope.interface
#pip install twisted
#pip install service_identity

- Scapy

Scapy es una librería para la manipulación interactiva de paquetes basado en Python que permite enviar, detectar y analizar paquetes de red para diferentes tipos de protocolos, además de otras funciones como graficas en 2D y 3D. Esta librería permite la construcción de herramientas o aplicaciones capaces explorar, escanear o atacar redes. Entre sus funciones más importantes están el análisis, decodificación y envío de paquetes de un gran número de protocolos [54]. Para la instalación de Scapy se ejecuta el siguiente comando.

 # pip install scapy

- MySQL

El simulador requiere almacenar los datos obtenidos, para ello es necesario instalar el sistema de administración de base de datos MySQL, ejecutando los siguientes comandos:

 # sudo apt-get update
 # sudo apt-get upgrade
 # sudo apt-get install mysql-server mysql-client
 # sudo mysql_secure_installation

- Connector MySQL-python

Para usar MySQL desde un proyecto de Python es necesario instalar MySQL Connector/Python, el cual es el conector de base de datos estandarizado para Python. Para ello se ejecutan los siguientes comandos.

#apt-get install python-dev libmysqlclient-dev
sudo apt-get install python-pip python-dev libmysqlclient-dev
pip install MySQL-python

B.3.2. Desarrollo del simulador de mensajes SIP

Para el desarrollo del simulador de mensajes SIP se tienen en cuenta las necesidades del simulador, las condiciones de red y su relación con el Core. Además, de antemano se sabe cómo es la comunicación que existe entre el Core IMS y un usuario, los tipos de mensajes SIP necesarios para el registro, conexión, desconexión de una llamada y diferentes eventos presentes durante dichos procesos. Se estudió el funcionamiento de Open IMS Core al interactuar con el cliente IMS (Fokus Monster), capturando y analizando los paquetes de dicha conexión en Wireshark. Este software admite visualizar detalladamente la estructura de cada uno de los paquetes intercambiados, lo que le permitió crear un mensaje apropiado en respuesta a los mensajes enviados por el Core.

Para la implementación del simulador se definen los siguientes componentes: (i) Diseño e implementación de la base de datos, (ii) Envío de paquetes, (iii) Escuchar, procesar y responder los mensajes SIP provenientes de Open IMS Core, (iv) Calculo de tiempos de respuesta y paquetes perdidos.

i. Diseño e implementación de la base de datos

Para realizar el diseño de la base de datos del simulador de mensajes SIP se tuvieron en cuenta los criterios y condiciones que demanda Open IMS Core para la comunicación con un cliente IMS y la estructura de cada mensaje involucrado, ya que entre ellos se comparten secuencias de mensajes, identificadores únicos para cada llamada requeridos para la estructura de un mensaje de respuesta, entre otros. Cabe notar que la base de datos es

utilizado por el simulador de mensajes SIP y el balanceador de carga, pero cada uno usando las tablas necesarias para su correcto funcionamiento.

Se tienen en cuenta 3 modelos que permiten representar el diseño de la base de datos: Modelo Conceptual, Modelo Lógico y Modelo físico [55]. En el modelo conceptual se describe el contenido de la base de datos. En el modelo lógico se explica en detalle la estructura de la base de datos según el SGBD (sistema gestor de base de datos) y por último el modelo físico interpreta la descripción de la creación de la base de datos. Este modelo físico se adapta al SGBD escogido.

- Modelo Conceptual

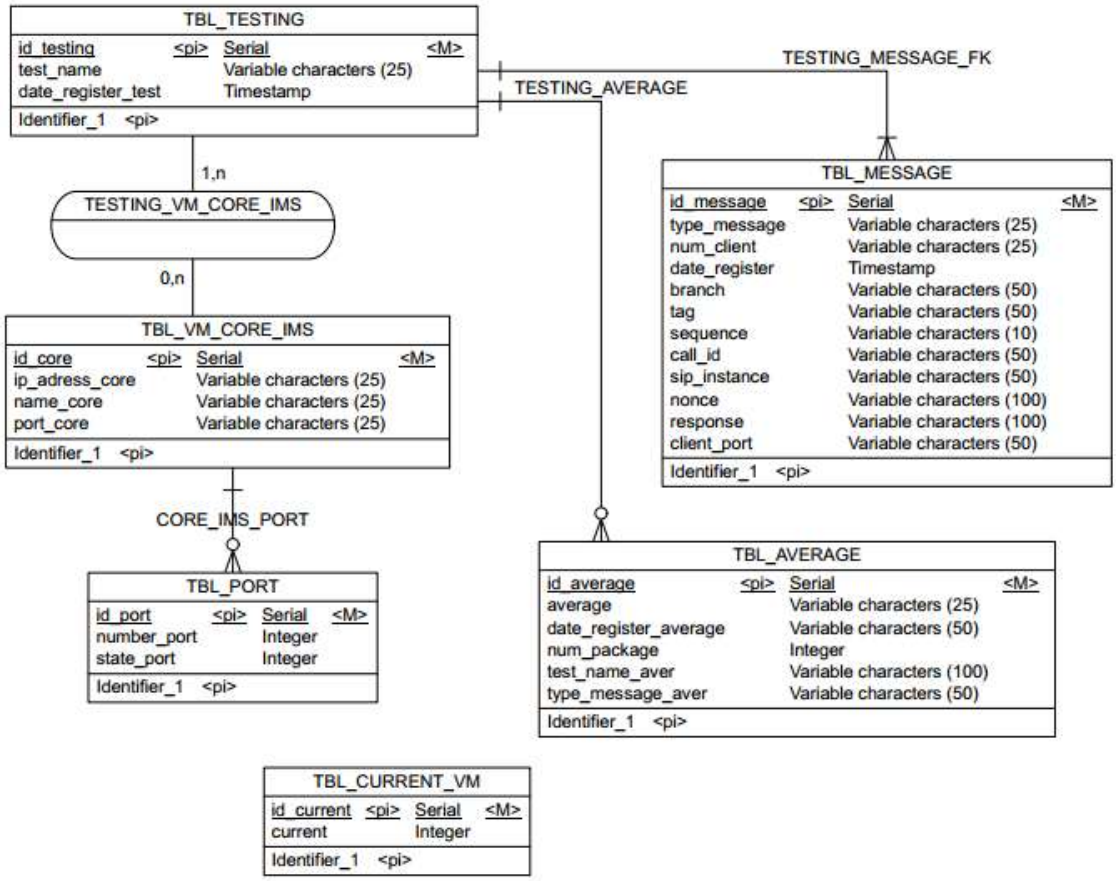

Figura 67. Modelo conceptual de la base de datos. Fuente: Propia

- Modelo Lógico

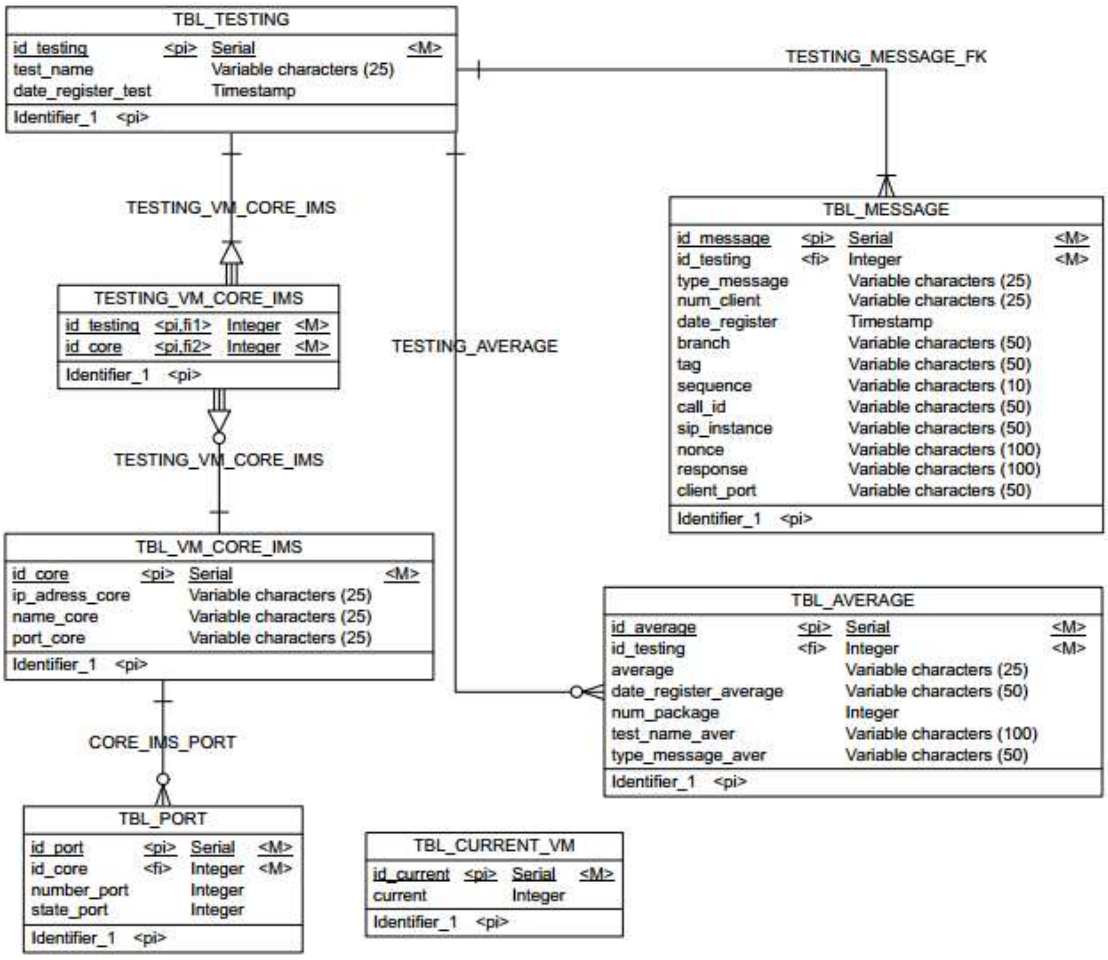

Figura 68. Modelo lógico de la base de datos. Fuente: Propia

- Modelo Físico

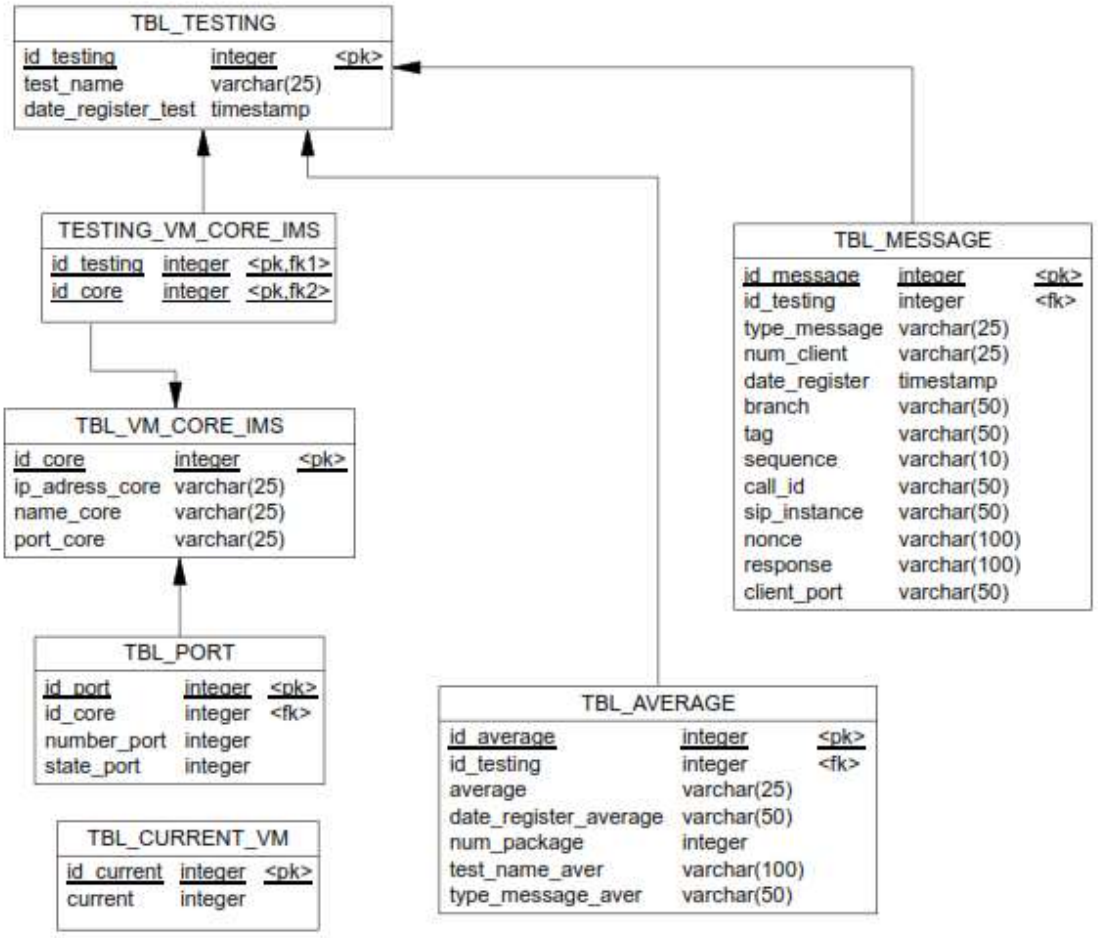

Figura 69. Modelo físico de la base de datos. Fuente: Propia

ii. Envío de paquetes

El principal objetivo es construir paquetes SIP con información de registro para un determinado número de usuarios hacia Open IMS Core de acuerdo a los tiempos registrados en el archivo .csv. Este componente es el encargado de enviar el primer paquete para el registro de un usuario (“Register”) hacia Open IMS Core, el cual tiene unos parámetros de entrada específicos para su correcto funcionamiento. Primero se especifica un rango de usuarios, digitando el número de inicio y el número final, cabe notar que los usuarios que están en este rango deben estar creados en Open IMS Core. Después, se especifica el nombre del archivo con extensión .csv de los tiempos de establecimiento de conexiones, este archivo debe estar en el mismo directorio del proyecto. Por último, se digita el nombre deseado para la prueba, este registro se guarda en la base de datos junto con la fecha de creación.

Cada prueba está asociada a una o varias máquinas virtuales, para el caso del simulador solamente se asocia a una máquina virtual con una dirección IP, este registro se guarda en la tabla TBL_VM_CORE_IMS de la base de datos, así

que cada prueba tiene asociado la dirección IP a donde se va a enviar el mensaje SIP.

Este componente se ejecuta por consola, como se observa en la figura 70.

```
root@Telco: /home/telco/pythonProjects
telco@Telco:~$ sudo su
[sudo] password for telco:
root@Telco:/home/telco# cd pythonProjects/
root@Telco:/home/telco/pythonProjects# python attackSip.py -f 101 -t 601 -p "lambda50" -n "Prueba1"
```

Figura 70. Comandos para el envío de paquetes. Fuente: Propia

En el siguiente diagrama de flujo se puede explicar el funcionamiento básico del primer componente del simulador de mensajes SIP.

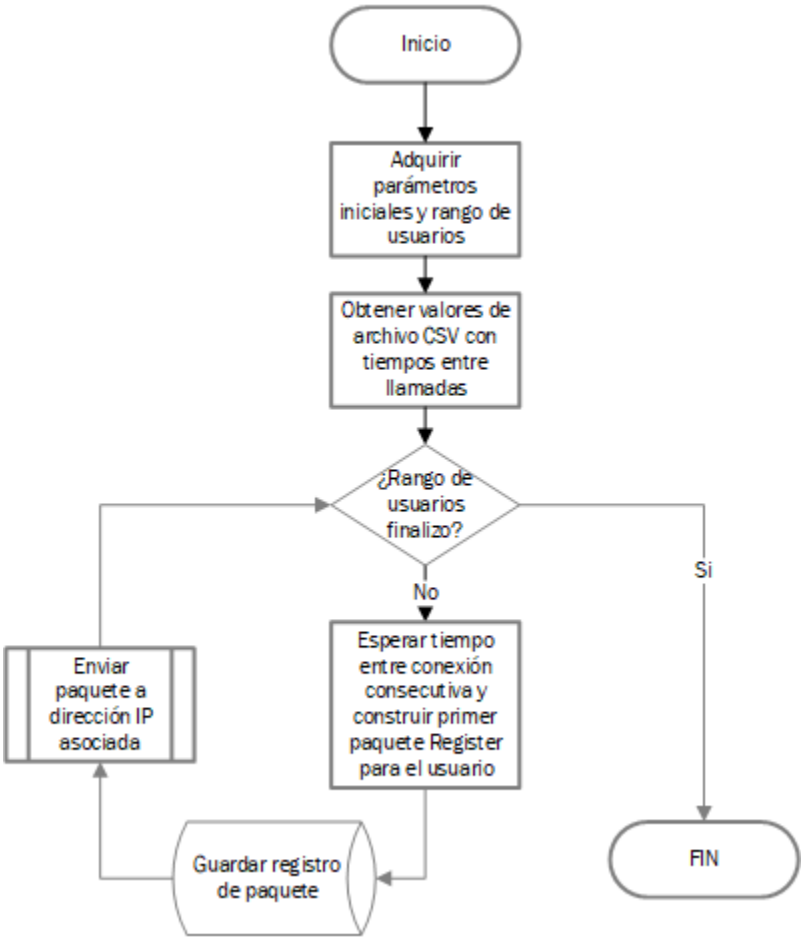

Figura 71. Diagrama de flujo del envío de peticiones. Fuente: Propia

iii. Escuchar, procesar y enviar mensajes SIP provenientes de Open IMS Core

Este es el componente más complejo del simulador de mensajes SIP, ya que es el encargado de escuchar los mensajes provenientes de Open IMS Core, clasificarlos y según el mensaje de entrada construir un mensaje de respuesta y enviarlo de nuevo a Open IMS Core según los parámetros guardados en el primer componente del simulador y de las respuestas enviadas por el Core. Los

parámetros de entrada de este componente son: la cantidad de conexiones a realizar y el nombre de la prueba. Estos datos deben ser iguales a los parámetros de entrada configurados en el primer componente, donde el número de conexiones debe ser igual al número de usuarios.

Este componente se ejecuta por consola, como se observa en la figura 72.

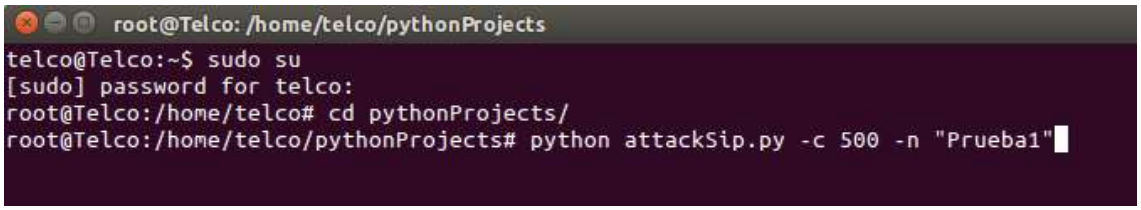

Figura 72. Comandos para escuchar, procesar y enviar mensajes SIP. Fuente: Propia

Con este comando el componente escucha por los puertos designados en el primer componente y según la cantidad de conexiones a realizar.

La librería Twisted tiene una limitación en cuanto al número de puertos habilitados para escuchar los mensajes de Open IMS Core, por esta razón al momento de enviar el último mensaje de desconexión el puerto se libera y se registra en la base de datos el estado del mismo. Cada máquina virtual tiene asociada la lista de puertos con su respectivo estado, libre o en uso, para utilizar los puertos libres en futuras conexiones cuando se sobrepase el límite que permite la librería.

En el diagrama de flujos de la figura 73 , se explica el funcionamiento básico de este componente.

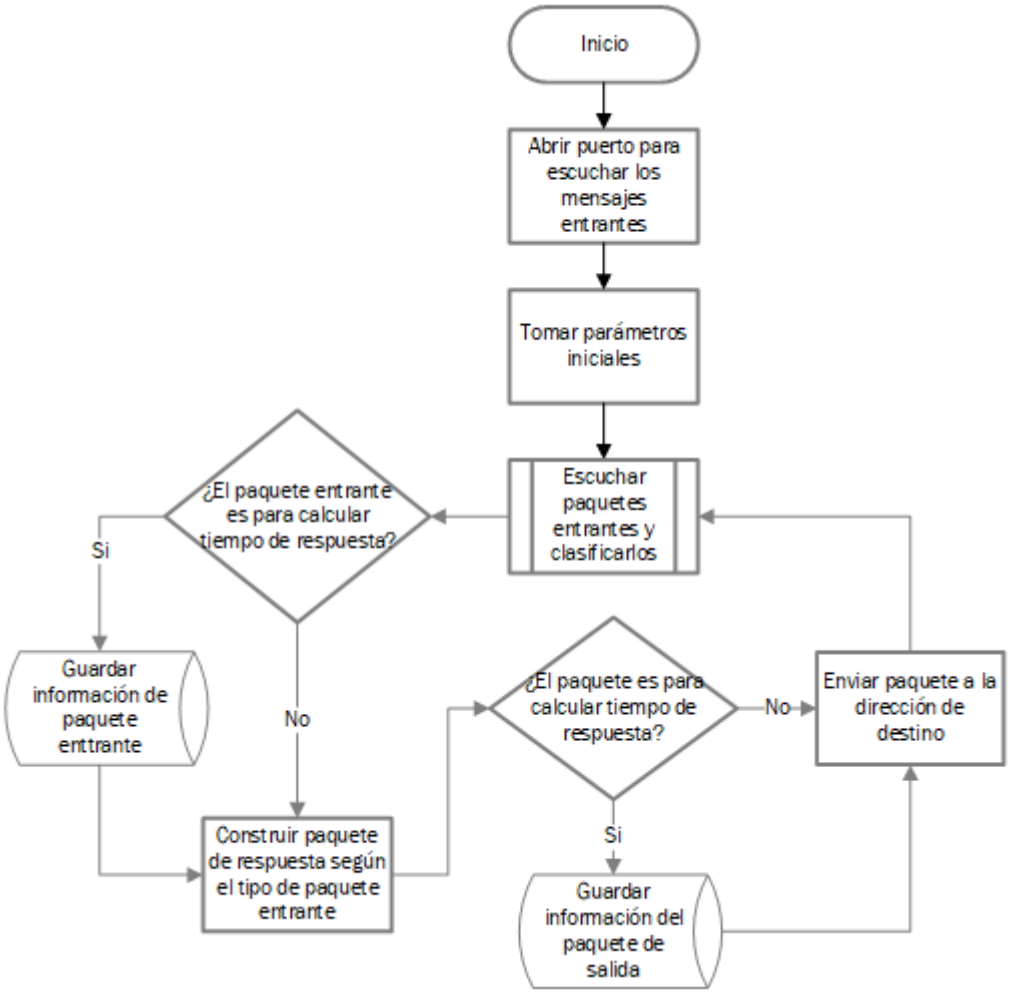

Figura 73. Diagrama de flujo para escuchar, procesar y enviar mensajes SIP. Fuente: Propia

iv. Calculo de tiempos de respuesta y perdida de paquetes

De los anteriores componentes se almacena un registro de cada paquete enviado o recibido, con el propósito de calcular el tiempo de respuesta de Open IMS Core. En este cálculo se ven involucrados 10 mensajes, 5 enviados y 5 recibidos, tomando 5 tiempos de respuesta según los procesos descritos en la sección 3.4.2 del presente documento, para los procesos de registro, suscripción y desconexión.

Este componente toma el tiempo de llegada del mensaje de respuesta del Core y le resta el tiempo del mensaje enviado que produce tal respuesta. Este proceso lo repite para cada uno de los 5 mensajes de respuesta del Core de cada usuario. Una vez se tiene los tiempos de respuesta de todos los usuarios involucrados, se calcula el tiempo promedio de todos los mensajes. En cuanto a la perdida de paquetes, se tiene un registro de todos los paquetes enviados y recibidos, evidenciando así la ausencia de algún mensaje.

Este componente se ejecuta después de que los anteriores módulos ya han terminado todos los procesos para cada usuario (figura 74).

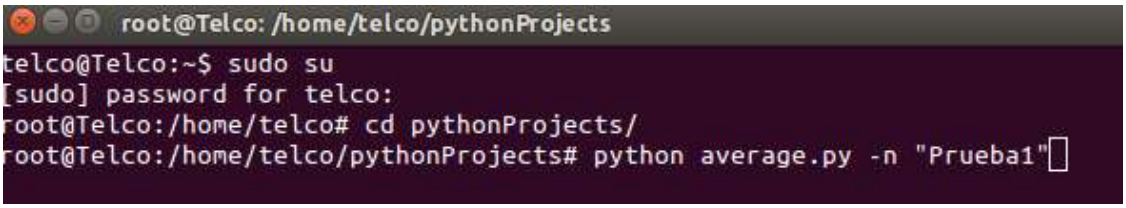

Figura 74. Comando para el cálculo de tiempos de respuesta y pérdida de paquetes. Fuente: Propia

Estos promedios de tiempo de respuesta y número de paquetes perdidos son almacenados en la base de datos para después ser analizados por el usuario.

B.4. BALANCEADOR DE CARGA

Para el desarrollo del balanceador de carga se utilizan las mismas herramientas software, lenguaje de programación y base de datos usados en el simulador de mensajes SIP. A diferencia del simulador, el balanceador solo posee un componente que se encarga de hacer todo el proceso de balanceo de carga.

Este componente tiene dos parámetros de entrada, uno es la cantidad de usuarios, lo que permite habilitar los puertos y escuchar las peticiones de los usuarios asociados a dichos puertos. El otro parámetro de entrada es la dirección IP del simulador, de esta forma el balanceador de carga identifica de donde son enviadas las peticiones. Previamente se deben haber ingresado los datos de las máquinas virtuales donde están alojados los diferentes Open IMS Core asociados al balanceador de carga, de esta forma el balanceador tiene presente la información de las diferentes máquinas virtuales para enviar las solicitudes según el algoritmo de balanceo de carga escogido (Round Robin).

El funcionamiento del balanceador de carga es sencillo y eficaz, detecta el primer paquete enviado por el simulador (“Register”) de cada usuario y lo almacena en la base de datos, enviando equitativamente estos mensajes a los Open IMS Core asociados. De esta forma cada vez que llega un mensaje proveniente del simulador este ya conoce a que máquina virtual debe ir este mensaje. Cuando el balanceador detecta un mensaje proveniente de las máquinas virtuales asociadas, este es enviado al único destino, el simulador de llamadas.

El tiempo que adiciona el balanceador de carga al tiempo de respuesta es despreciable, ya que el balanceador envía los mensajes de manera inmediata. En el diagrama de flujos de la figura 75, se explica el funcionamiento básico de este componente.

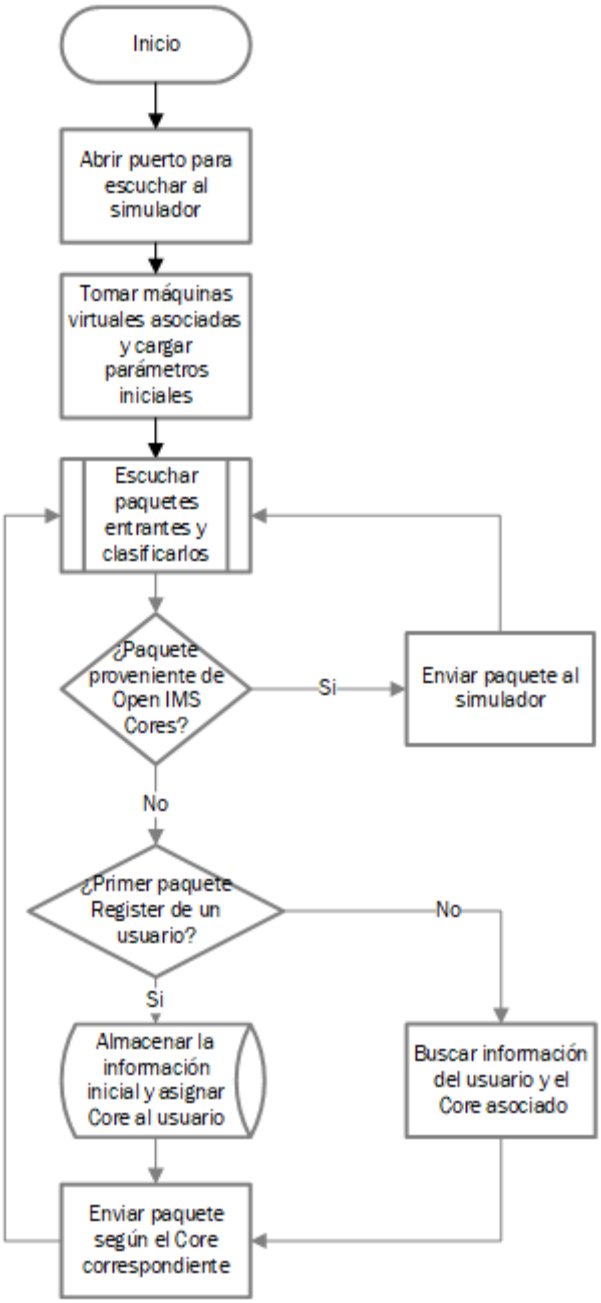

Figura 75. Diagrama de flujo del balanceador de carga. Fuente: Propia