Universidad de Mendoza

Facultad de Ingeniería

Tesis de Maestría en Teleinformática

“Análisis y Estudio Experimental del Comportamiento de Métricas de QoS y QoE de Streamings de Video Multicast IPTV”

Esp. Lic. Varela Pablo Nicolás

Directores de Tesis:

Mg. Ing. Higinio Alberto Facchini

Dr. Ing. Santiago Cristóbal Pérez

Mendoza, Mayo 2022

AGRADECIMIENTOS

Quiero agradecer especialmente a Santiago Pérez e Higinio Facchini por ser mis tutores, darme su apoyo y su tiempo para poder realizar este trabajo. Además, por la cooperación de este trabajo de tesis agradezco a la Universidad Tecnológica Nacional que me ha brindado las herramientas necesarias para la experimentación.

Quiero agradecer a mi familia que es la que me dio el fortalecimiento necesario para crecer como persona y las pautas necesarias para ser un buen profesional.

Agradezco, también, el apoyo de Dios por brindarme la capacidad y la autocrítica necesaria para poder llevar a cabo la carrera y lograr la maestría.

Y por útlimo agradezco agradezco a Miguel Mendez, a Osvaldo Marianetti, y al cuerpo de profesores de la Universidad de Mendoza, por haber sido parte del conocimiento brindado en varios temas durante el transcurso de la maestría.

ÍNDICE

i.VISION GENERAL 1
i.1. INTRODUCCIÓN 1
i.2. OBJETIVOS 3
i.2.1. Objetivo General 3
i.2.2. Objetivos Específicos 3
i.3. PUBLICACIONES REALIZADAS 4
i.4. CONTRIBUCIONES PRINCIPALES 5
i.5. MARCO TEÓRICO 5
i.6. HIPÓTESIS 7
I.INTRODUCCIÓN A LA TECNOLOGIA IPTV 9
I.1.INTRODUCCIÓN 9
I.2. PANORAMA GENERAL IPTV 9
I.2.1. Características IPTV 10
I.2.2. Reseña del proceso de señales de video para el manejo sobre IPTV 11
I.3. MODELO DE COMUNICACIONES IPTV 19
I.3.1. Gráfica y Conceptos sobre el Modelo de Comunicaciones IPTV 20
II.CASO DE ESTUDIO DE UNA RED IPTV 25
II.1. DISEÑO DE LA RED IPTV 25
II.1.1. Cálculo de implementación de Servicio por Abonado 28
II.1.2. Orígenes de las Señales IPTV Sobre IRD o Cabezales Externos 29
II.1.3. Funcionalidades del Switch IPTV 40
II.1.4. Servidor Middleware 41
II.1.5. Servidor DHCP 42
II.1.6. Dispositivos finales STB 44
II.1.7. Controles Necesarios en el Servicio IPTV 45
III.DESCRIPCIÓN DE LOS RECURSOS Y HERRAMIENTAS 47
PARA LA EXPERIMENTACIÓN IPTV 47
III.1. SOFTWARE FFMPEG SERVER 47
III.1.1. Descripción General del framework 48
III.1.2. Caso Práctico de Funcionamiento 49
III.2. SOFTWARE WIRESHARK DESCRIPCIÓN 56
III.2.1. Ejemplo de captura de Paquetes de Datos IPTV 56
III.3. ENTORNO DE DESARROLLO INTEGRADO 59
III.3.1. Librería Panda bajo Python 59
III.3.2. Librería Mathplotlib bajo Python 60

III.3.3. Librería openpyxl bajo Python 61
IV.ANALISIS EXPERIMENTAL DE LAS CARACTERISTICAS DE QoS DEL TRÁFICO IPTV EN LA RED DE ESTUDIO 63
IV.1. CALIDAD DE SERVICIO QoS 63
IV.2. VISIÓN GENERAL DE CÓDECS DE VIDEO 65
IV.3. METODOLOGÍA DE ANÁLISIS DE LAS MEDICIONES OBTENIDAS EN LOS ENSAYOS EXPERIMENTALES 68
IV.3.1. Procesamiento de los archivos Wireshark de captura del tráfico de video IPTV 68
IV.3.2. Análisis de la métrica Retardo (Delay) 70
IV.3.3. Análisis de la métrica Diferencia de Retardo (Jitter) 71
IV.4. ANÁLISIS DEL TRÁFICO IPTV CON EL CÓDEC H.264 73
IV.4.1. Análisis de la métrica Retardo 73
IV.4.2. Análisis de la métrica Jitter 75
IV.5. ANÁLISIS DEL TRÁFICO IPTV CON EL CÓDEC H.265 78
IV.5.1. Análisis de la métrica Retardo 78
IV.5.2. Análisis de la métrica Jitter 80
IV.6. ANÁLISIS DEL TRÁFICO IPTV CON EL CÓDEC THEORA 83
IV.6.1. Análisis de la métrica Retardo 83
IV.6.2. Análisis de la métrica Jitter 85
IV.7. ANÁLISIS DEL TRÁFICO IPTV CON EL CÓDEC VP8 88
IV.7.1. Análisis de la métrica Retardo 88
IV.7.2. Análisis de la métrica Jitter 90
IV.8. ANÁLISIS COMPARATIVO DEL COMPORTAMIENTO DE MÉTRICAS DE QoS DE LOS CÓDECS EN LA RED EXPERIMENTAL 93
IV.8.1. Análisis de la métrica Retardo 93
IV.8.2. Análisis de la métrica Jitter 94
V.ANALISIS EXPERIMENTAL DE LAS CARACTERISTICAS DE QoE DEL TRÁFICO IPTV EN LA RED DE ESTUDIO 95
V.1. CALIDAD DE EXPERIENCIA QoE 95
V.2 MODELOS DE MAPEO DE QoS/QoE en IPTV 100
V.3. METODOLOGÍA DE ANÁLISIS DE LAS QoE 101
V.3.1. Introducción 101
V.3.2. Determinación de la QoE 102
V.3.3. Aplicación en los escenarios de experimentación 103
V.4. ANÁLISIS DE LA QoE PARA EL CÓDEC H.264 105
V.5. ANÁLISIS DE LA QoE PARA EL CÓDEC H.265 109

V.6. ANÁLISIS DE LA QoE PARA EL CÓDEC THEORA 113
V.7. ANÁLISIS DE LA QoE PARA EL CÓDEC VP8 117
V.8. ANÁLISIS COMPARATIVO DEL COMPORTAMIENTO DE MÉTRICAS DE QoE DE LOS CÓDECS EN LA RED EXPERIMENTAL 121
V.8.1. Análisis de la métrica Retardo 121
V.8.2. Análisis de la métrica Jitter 122
VI.CONCLUSIONES y TRABAJOS FUTUROS 125
VI.1. INTRODUCCIÓN 125
VI.2. CONTRIBUCIONES PRINCIPALES 126
VI.3. CONCLUSIÓN 126
VI.4. TRABAJOS PRESENTES Y FUTUROS 128
VII.BIBLIOGRAFIA 131
VII.A. ESTÁNDARES DE IMPORTANCIA 134
APÉNDICE 135
ANEXO A: CÁLCULO DE EXCEL A PYTHON QoS 136
ANEXO B: CÁLCULO DE EXCEL A PYTHON QoE 141
ANEXO C: CÁLCULO CONCLUSIÓN QoS 145
ANEXO D: CÁLCULO CONCLUSIÓN QoE 147

ÍNDICE DE FIGURAS

I.3. Modelo de comunicaciones IPTV 20
II. Topología IPTV 25
II.1.2. Dispositivo IRD 29
II.1.2.1. Control IRD 30
II.1.2.1. Tuner IRD 31
II.1.2.1. Stps IRD 32
II.1.2.1. Output IRD 33
II.1.2.3. Router IPTV 35
II.1.3. Switch IPTV 40
II.1.5. Configuración VLANs 43
II.1.5. VLANs archivo de peticiones 44
III.1.2. Configuración FFM 50
III.1.2. Control VLC 50
Red y Software FFMPEG 51
III.2.1. Topología Experimental IPTV 57
III.2.1. Cliente 58
III.2.1. Captura desde el Servidor 58
III.3. IDE Spyder 59
IV.3.1. Tráfico de captura IPTV 68
IV.3.1. Archivo Formato .csv 69
IV.3.2. Resultado Delay 70
IV.3.2. Cálculos del Plot 71
IV.3.3. Valores Estadísticos por Consola 72
IV.3.3. Valores de Jitter 73
IV.4.1. Repeticiones por Retardo 74
IV.4.1. Resultados en Frecuencia 75
IV.4.2. Repeticiones del Jitter 76
IV.4.2. Jitter valores de Frecuencia 77
IV.5.1. Repeticiones por Retardo 79
IV.5.1. Resultados del delay en Frecuancia 79
IV.5.2. Repeticiones del Jitter 81
IV.5.2. Jitter H.265 81
IV.6.1. Cantidad de Repeticones por Retardo 84
IV.6.1. Resultados en Frecuencia 84

IV.6.2. Cantidad de Repeticiones en del parámetro jitter 86
IV.6.2. Resultados en Frecuencia 86
IV.7.1. Cantidad de Repeticiones por Retardo 89
IV.7.1. Resultados en Frecuencia 89
IV.7.2. Repeticiones por Jitter 91
IV.7.2. Resultados en Frecuencia 91
IV.8.1. Curvas solapadas con el Delay 93
IV.8.2. Métricas solapadas del Jitter 94
V.1. Dominios QoS y QoE 96
V.3.2. QoE en función del Delay 103
V.4. QoE Delay/Jitter P H.264 106
V.4. QoE Delay/Jitter M H.264 106
V.4. QoE Jitter/Delay P. H.264 107
V.4. QoE Jitter/Delay M H264 108
V.4. QoE figura única H.264 108
V.5. QoE Delay/Jitter P. H.265 109
V.5. QoE Delay/Jitter M. H.265 110
V.5. QoE Jitter/Delay P H.265 111
V.5. QoE Jitter/Delay M H.265 111
V.5. QoE única figura H.265 112
V.6. QoE Delay/Jitter P Theora 113
V.6. QoE Delay/Jitter M Theora 114
V.6. QoE Jitter/Delay P Theora 115
V.6. QoE Jitter/Delay M Theora 115
V.6. QoE única figura Theora 116
V.7. QoE Delay/Jitter P VP8 117
V.7. QoE Delay/Jitter M VP8 118
V.7. QoE Jitter/Delay P VP8 119
V.7. QoE Jitter/Delay M VP8 119
V.7. QoE única figura VP8 120
V.8.1. QoE para valores de Delay 121
V.8.2. QoE para valores de Jitter 122

ÍNDICE DE TABLAS

IV.1. Tabla UIT 64
Tabla de Tramas 69
Tabla Tiempo Delay 74
Repetición valores de Jitter 76
Métricas códec H.264 77
Tiempo de Retado 78
Repeticiones de los valores del Jitter 80
Métricas códec H.265 82
Repeticiones del Delay 83
Repetición de los valores de Jitter 85
Métricas códec Theora 87
Repeticiones en el tiempo por retardo 88
Repetición valores de Jitter 90
Métricas códec VP8 92
VI.4. Perspectiva desde MOS 129

RESUMEN

La televisión digital es el avance más importante en la tecnología de televisión desde que se inventó el medio hace más de 100 años. La televisión digital brinda a los usuarios más opciones, y permite una experiencia visual más interactiva. El sistema tradicional analógico de televisión ha estado en funcionamiento durante más de setenta años. Durante este tiempo, los espectadores experimentaron la transición del blanco y negro a televisores en color.

Durante los últimos años, la industria ha estado atravesando una profunda transición, migrando de la televisión convencional a una nueva era de la tecnología digital. Mientras, la mayoría de los operadores de TV han actualizado sus redes existentes y han implementado plataformas digitales avanzadas, en un esfuerzo por migrar a sus suscriptores de los servicios analógicos tradicionales, a los servicios digitales más sofisticados. Una nueva tecnología llamada televisión basada en el protocolo de Internet (IPTV), ha comenzado a encabezar titulares alrededor del mundo con historias sobre compañías de telecomunicaciones, cable, satélite, y una gran cantidad de empresas emergentes de Internet que entregan video a través de un servicio basado en IP[1]. Como sugiere el nombre, IPTV describe un mecanismo para transportar un flujo de contenido de video a través de una red que utiliza el protocolo de red IP.

Es decir, la Televisión por Protocolo de Internet (IPTV), es el resultado de la unión de dos servicios de Telecomunicaciones; la del Internet y la Televisión, posibilitando y dando lugar a nuevas formas de entretenimiento para los usuarios, y la posibilidad de mayores ingresos para los proveedores de dichos servicios. La Televisión IP debe ser una experiencia totalmente personalizada que debe garantizar la Calidad de Servicio (QoS)[2] (Calidad de servicio) y la QoE[3] (Calidad de Experiencia) dentro de la organización.

La plataforma IPTV no tiene límites, incluyendo las Redes LAN[4] de un Canal de Televisión, o las Redes LAN tradicionales, que nos permiten obtener las

velocidades necesarias de datos, para el manejo interno de la red. Sin embargo, hay que considerar que el servicio audio visual que se brinda debe dar soporte a varios usuarios conectados, en forma simultánea, en sub redes de tipo inalámbricas o cableadas, por lo cual deben considerarse varios factores, para dar un servicio correcto.

En este contexto, la más importante motivación para el presente trabajo de investigación de esta tesis fue analizar el comportamiento de tráfico IPTV, en una red LAN experimental con tráfico controlado. Se utilizaron diferentes codecs para contraste, y se establecieron resultados cuantitativos detallados de las métricas de QoS y, a partir de ellas, valores indicativos de QoE, que orientan sobre su conducta.

Se utilizó una topología de red de experimentación, un servidor y clientes de video IPTV, un trailer de Star Trek, con 4 subescenarios o casos particulares, por cada uno de 4 codecs: H.264, H.265, Theora y VP8. El tráfico fue capturado para análisis, en todos los casos, usando un sniffer. Y procesado usando rutinas en lenguaje Python.

Se demostró experimentalmente que:

- La métrica de retardo para tráfico IPTV, en la red experimental, presenta un comportamiento claramente diferenciado, a excepción de los codecs H.264 y H.265. Estos últimos codecs concentran sus retardos en el orden de los 2 mseg, mientras que el códec Theora lo hace en el orden de los 1,5 mseg. Finalmente, el códec VP8 concentra sus retardos en el orden de 1 mseg.

---

[1]IP Internet Protocol, este protocolo permite la identificación de un dispositivo en la red de redes. Para tener una visión más amplia consulte https://www.itu.int/osg/csd/mina/2001/ip-itu-2001-es.html

[2]LAN es una definicón que comprende parte del desarrollo de las redes. Una red que abarca distancias acotadas.

[3]QoS o Calidad de Servicio es un término que define aquellos parámetros técnicos que dan resultado de la eficiencia cuando se porcede a transmitir datos de video, audio y datos por la misma.

[4]QoE o Calidad de Experiencia es una definición que mide los resultados de la calidad de reproducción desde el punto visual del usuario final.

- La métrica jitter para tráfico IPTV, en la red experimental, presenta un comportamiento claramente diferenciado. Cada códec tiene una respuesta en la forma de triángulo, cuando se representa la cantidad de valores, en repetición, en función del jitter. Los códec tienen bases y alturas distintas. El codec H.264 (Theora) presenta la peor respuesta general.
- La respuesta de QoE, en base a la expresión matemática utiliza, usando el retardo (jitter) como variable independiente, y el jitter (retardo) como parámetro, muestra que los H.264 y H.265 tienen el peor comportamiento. El códec VP8 presenta el mejor comportamiento, mientras que el codec Theora se ubica en una respuesta intermedia.

CAPITULO i

i.VISION GENERAL

Esta tesis aspira a ser una contribución para el avance del estado del arte de las redes LAN y, específicamente, para el manejo de datos audiovisuales usando el servicio IPTV. En la tematica, se efectuaron ensayos experimentales a través de distintos subescenarios, dónde cada subescenario sirvió para la prueba y la evaluación de distintos algoritmos de codificación procesados bajo un mismo lenguaje de programación. El procesamiento cuantitativo de los algoritmos permitió determinar que valores de QoS (Calidad de Servicio) y de QoE (Calidad de Experiencia) proporcionaron la mejor experiencia audiovisual. Por lo tanto, en base a estos, y otros detalles que posteriormente se mencionarán, se da a conocer el alcance de la investigación de la tesis.

i.1. INTRODUCCIÓN

Con el constante avance y desarrollo tecnológico de las redes LAN, éstas se han consolidado en Redes de Datos de áreas, edificios o campus locales. En ellas, se ve la necesidad de las distintas instituciones existentes, para obtener el máximo nivel de utilización significativa de los datos, ahora multimediales, para fortalecer la comunicación institucional.

La comunicación es el elemento fundamental primario en el que se rigen las distintas instituciones, como empresas, compañías, universidades u otras organizaciones similares. Por lo tanto, debe de comprenderse que este proceso comunicacional está plenamente ligado con la tecnología subyacente, implementada por la institución, para la transmisión de manera correcta de los datos hacia sus usuarios. La forma más habitual de comunicación, debido a la tecnología actual, es a través de las redes, y con una tendencia hacia el formato audiovisual que transmiten.

Para la transmisión del formato audiovisual debe entenderse que los receptores serán dispositivos tales como tablets, notebooks, smartphones, computadoras, etc., es decir, una diversidad de dispositivos que pueden ser parte de la red

LAN. Y para que los datos lleguen a cada dispositivo en dicha red, cada uno de estos dispositivos tendrá que estar configurado apropiadamente través de la pila TCP/IP.

El uso de la red por estos dispositivos procede a establecer un consumo de tráfico, seguramente variable, que abarca parte del ancho de banda que aquella puede proporcionar. Lógicamente, que se toma como un hecho que habrá flujos de datos de distintas características, inclusive multimediales. Por lo tanto, se debe analizar cada situación particular, del total de datos, con respecto al uso de la capacidad de la red, y ver de qué forma establecer los parámetros de configuración de los dispositivos de red (servidores y dispositivos de comunicaciones), para brindar un servicio de transmisión de calidad.

Hay que tener en cuenta que la conducción de tráfico audio/visual está destinado a una gran cantidad de usuarios, debido a que este se reproduce a través de un mismo canal, hacia un grupo particular mediante tecnología multicast. Además, de esta misma forma, se pueden reproducir varios videos hacia distintos grupos, por distintos canales. Esto es lo que se denomina streaming por canal, hacia un grupo de usuarios, y es la lógica de funcionamiento que maneja IPTV[5]. Es necesario saber, que a partir de lo comentado hay que diferenciar este servicio de otros en línea, tales como canales de internet gratuitos o de tipo "Youtube", en los que los videos se pueden recargar y mirar sin garantías de calidad.

Dentro de las especificaciones para IPTV, se garantiza que estos servicios (audio/visuales) de televisión están preparados para ser seguros y rentables, lo que promete garantías de calidad visual y de sincronismo constante.

[5]IPTV este servicio de televisión que se maneja por protocolo IP. El servicio proporciona una arquitectura con capas bien diferenciadas. Si se desea puede encontrar información detallada del mismo siguiendo el siguiente link: https://www.itu.int/itunews/manager/display.asp?lang=es&year=2008&issue=08&ipage=28&ext=html

Para prometer un servicio fiable en IPTV entran dos conceptos claves que tratan de definir el nivel de calidad de servicio obtenido: la Calidad de Servicio (QoS) y, por otro lado, la Calidad de Experiencia (QoE). En este trabajo de tesis se plantearon diferentes situaciones, considerando parámetros, tales como el uso del ancho de banda, jitter, retardos, entre otros, para un entorno IPTV.

De esta manera, y a partir de una elección de codecs preliminar, se crearon distintos subescenarios de experimentación, para determinar mediante análisis cuantitativo, el nivel de la calidad obtenida para cada caso.

i.2. OBJETIVOS

i.2.1. Objetivo General

Obtener la evaluación de parámetros de Calidad de Servicio y de Calidad de Experiencia usando diversos codecs, a través de un estudio y análisis experimental del servicio streaming multicast en formato IPTV, aplicado a la red de datos de la Universidad Tecnológica Nacional.

i.2.2. Objetivos Específicos

- Llevar a cabo un estudio sobre el comportamiento de sistema de Televisión por protocolo IPTV [1], para determinar cuáles son los datos y métricas relevantes en función de los protocolos IPTV para IPv4, cuando el tráfico se origina, viaja y llega a un destino en la red de campus de la UTN FRM.
- Definir un caso de estudio real general, y subescenarios, sobre el paradigma IPTV, que permita realizar mediciones por captura de tráfico, y evaluar el comportamiento del tráfico para cada caso.
- Conocer y evaluar específicamente los parámetros y métricas de calidad de servicio QoS y de experiencia QoE, para cada subescenario, usando un

servicio de streaming multicast [2] en formato IPTV, con destino a redes que puedan incorporar variados dispositivos inalámbricos y/o cableados.

- Implementar frameworks específicos, para hacer pruebas de streaming de video de tipo "ffmpeg", desde el servidor hacia dispositivos finales.
- Estudiar y comparar diferentes softwares clientes para la recepción de IPTV. El trabajo consistirá en búsquedas y pruebas con los diferentes modelos de softwares, que se evaluarán para determinar entre ellos cual maneja mejor el formato IPTV según el códec utilizado.
- Construir tablas comparativas, haciendo uso de recursos de modelado, que permitan denotar, medir y diferenciar los comportamientos de QoS y QoE, para los diferentes subescenarios de experimentación del servicio IPTV.
- Enunciar y establecer conclusiones acerca de los resultados más relevantes, para cada subescenario, del servicio IPTV, para video con óptima calidad, considerando, las opciones propuestas por el autor Facchini [3]:

1. La cantidad de bytes y de paquetes por códec.
2. La tasa de bits.
3. El tamaño de los paquetes promedio.
4. El espacio intertrama.
5. La distribución estadística de paquetes por orden de llegada, y por espacio intertrama, que los mismos presenten.

i.3. PUBLICACIONES REALIZADAS

Este proyecto ha sido presentado en los siguiente Congreso, Workshop y Encuentro de divulgación de las ciencias informáticas:

- XXII Workshop de Investigadores en Ciencias de la Computación - (WICC), 7 y 8 de Mayo de 2020, UNPA, Santa Cruz, Argentina.[4]

- CACIC 2022, Congreso Argentina de Ciencias de la Computación, 3 al 6 de octubre 2022, UNLaR, La Rioja, Argentina, Aprobado para publicación. [5]

i.4. CONTRIBUCIONES PRINCIPALES

- Los resultados cuantitativos de los estudios por experimentación sobre una red LAN de laboratorio real, propuesto al efecto, para evaluar el comportamiento de QoS y de QoE, para tráfico IPTV y diferentes codecs, en un todo de acuerdo a la hipótesis.
- Las conclusiones de dicho estudio podrán ser utilizadas como referencias a contextos similares, y por analistas de simulación para la parametrización de los simuladores de tráfico IPTV.
- La propuesta topológica y metodológica de la red LAN de laboratorio puede usarse por otros investigadores a los efectos de contraste, usando otros videos y/o codecs, u otros programas de gestión de videos, cuyo objetivo sea el análisis de QoS o QoE, o ambos de un streaming de video IPTV.

i.5. MARCO TEÓRICO

Hay un esfuerzo constante en los grupos de investigación para entender el comportamiento del tráfico en las redes de datos, y la conducta del tráfico generado o recibido por los nodos de red (servidores, puestos de trabajo final, y los dispositivos activos), para mejorar el rendimiento, y la productividad de los mismas.

Un escaso entendimiento puede conducir a conclusiones o interpretaciones simplificadas, planificaciones erradas, simulaciones deficientes, con la consiguiente falta de estabilidad en la infraestructura.

El trabajo de tesis es parte de la Teoría de Ingeniería de Tráfico de Redes de Datos. La Teoría de Tráfico de Redes es la aplicación de la teoría de la probabilidad a la solución de los problemas relacionados con: la planificación, la evaluación de prestaciones, la operación y el mantenimiento de los sistemas de comunicaciones.

El objetivo de la teoría es hacer medible el tráfico usando modelos matemáticos, derivar relaciones entre el grado de servicio y la capacidad del sistema, y convertir la teoría en una herramienta para la planificación de la inversión. Y el objetivo final de la Teoría de Redes de Datos es diseñar sistemas a un costo tan efectivo como sea posible, con un grado predefinido de servicio, cuando se conoce la demanda de tráfico futuro y la capacidad de los elementos del sistema.

La tarea del Ingeniero de Tráfico de Redes es especificar métodos para controlar que el grado de servicio real satisfaga los requerimientos, y especificar acciones de emergencia cuando los sistemas estén sobrecargados u ocurran fallas técnicas. Para esta tarea se requieren métodos para predecir la demanda (medidas de tráfico), métodos para calcular la capacidad de los sistemas y especificación de medidas cuantitativas para el grado de servicio.

El análisis de un sistema de comunicaciones requiere de un modelo del sistema. El modelo requiere conocimiento del sistema técnico, conocimiento de herramientas matemáticas disponibles y la implementación del modelo en una computadora.

El modelo contiene 3 componentes principales: la estructura del sistema, la estrategia operacional y las propiedades estadísticas del tráfico. La estructura del sistema está determinada técnicamente, y se puede obtener con cualquier nivel de detalle. La estrategia operacional se refiere a que un sistema físico dado puede usarse de diferentes formas, para adaptar el sistema a la demanda de tráfico. Por ejemplo, en un sistema de comunicaciones se aplican estrategias para dar prioridades y encaminar el tráfico al destino, o en los sistemas telefónicos clásicos se usó lógica cableada para las estrategias, mientras en los sistemas modernos se hace por software, con mayor flexibilidad y adaptación. Finalmente, las propiedades estadísticas del tráfico son parte del modelo, dado que las demandas del usuario, o de las aplicaciones, se modelan con propiedades estadísticas del tráfico.

Los modelos pueden clasificarse en tres clases:

- Modelos matemáticos o analíticos: son generales y frecuentemente aproximados. Optimizan los parámetros analíticamente o numéricamente. Son rápidos en los análisis.

- Modelos de simulación: dónde pueden usarse datos medidos o datos artificiales desde distribuciones estadísticas. Son más demandantes de recursos y tiempo dado que no son generales. Cada caso individual debe simularse.

- Modelos físicos o prototipos: son mucho más demandantes de recursos y tiempo que un modelo de simulación.

Este trabajo de tesis se ha realizado en el marco de la Teoría de la Ingeniería de Redes de Datos, utilizando un modelo físico real de experimentación, para la evaluación de las prestaciones y la operación de las Redes Multicast para IPTV, desde la perspectiva de QoS y QoE.

i.6. HIPÓTESIS

El trafico IPTV multicast aplicado a las redes LAN produce un nivel ajustado a los requerimientos de calidad de servicio para este tipo de redes, desde la pespectiva de QoS y QoE, independientemente de la topología de red, y de la infraestructura LAN subyacente. Especialmente, cuando se cuentan con los recursos disponibles suficientes para dar dicho servicio. Esto es cierto aun cuando se compartan las comunicaciones con diferentes tipos de tráficos de menor prioridad, y se presenten diferencias en el comportamiento de los codecs entre sí.

CAPITULO I

I.INTRODUCCIÓN A LA TECNOLOGIA IPTV

El presente capitulo presenta una revisión general de las técnicas, tecnologías y conceptos que abarca todo lo relacionado al servicio IPTV. El objetivo es plantear los conceptos principales, que serán utilizados en los capítulos subsiguientes, para comprender cuáles serán los métodos de estudio que se aplicaron en la investigación y el desarrollo de este trabajo de tesis.

I.1.INTRODUCCIÓN

La televisión digital es el avance más significativo en tecnología de televisión desde que se creó el medio hace más de un siglo, según Partison [6]. La televisión digital ofrece a los consumidores más opciones y hace que la experiencia visual sea más interactiva. La mayoría de los operadores de TV han actualizado sus redes existentes, y han implementado plataformas digitales avanzadas en un esfuerzo por migrar a sus suscriptores de los servicios analógicos tradicionales a los servicios digitales más sofisticados. Una nueva tecnología llamada Televisión basada en el Protocolo de Internet (IPTV) describe un mecanismo para transportar un flujo de contenido de video a través de una red que utiliza el protocolo de red IP. Los beneficios de este mecanismo de transmisión de señales de TV varían desde mayor soporte para la interactividad a tiempos de cambio de canal más rápidos, y mejorada interoperabilidad con las redes domésticas existentes.

I.2. PANORAMA GENERAL IPTV

En términos generales, IPTV es una definición que aplica a la entrega de canales de televisión tradicional, entre otras cosas películas y contenido de video a demanda a través de una red de tipo privada. Desde la perspectiva del usuario final, IPTV se ve y funciona como un servicio estándar de televisión pago.

La deficinión oficial aprobada por la Unión Internacional de Telecomunicaciones sobre IPTV (ITU-T FG IPTV) es la siguiente: IPTV se define como servicios multimedia como televisión / video / audio / texto / gráficos / datos entregados a través de redes basadas en IP, gestionados para proporcionar el nivel de calidad requerido de servicio y experiencia, seguridad, interactividad y confiabilidad.

Desde la perspectiva de un proveedor de servicio, IPTV abarca la adquisición, procesamiento y entrega segura de contenido de video, a través de una infraestructura de red basada en el protocolo IP.

I.2.1. Características IPTV

La plataforma IPTV posee características que permiten la mejor interactividad entre el servicio brindado, y el propósito que tiene el mismo con el usuario final. A continuación se mencionan cuales son ellas y cuales son sus funciones.

- Porporciona soporte para TV interactiva: Las capacidades bidireccionales de los sistemas IPTV permiten a los proveedores de servicio ofrecer todo una serie de aplicaciones de televisión interactiva. Los tipos de servicios prestados a través de un proveedor pueden incluir desde TV en vivo, TV en alta definición (HDTV)[6], juegos interactivos de alta velocidad.
- Time shifting: Es el caso similar al de una combinación de una grabadora de video digital. La cual permite un mecanismo para la grabación, y el contenido almacenado IPTV, para su posterior visualización.
- Personalización: Un sitema IPTV de extremo a extremo admite comunicaciones bidireccionales, y permite a los usuarios finales personalizar sus hábitos de visualización de televisión, al permitirles decidir que quieren ver y cuando quieren mirarlo.

[6]HDTV propiamente indica señales de TV de alta definición de entre 720p, 1080p y 1440p, como así también permite el trabajo del protocolo en conjunto con un sistema de audio Surround Dolby Digital.

- Requisitos de ancho de banda: En el lugar de entregar todos los canales a todos los usuarios finales, la tecnología IPTV permite a los proveedores de servicio transmitir un solo canal, que ha solicitado el usuario final. Esta característica permite a la red de operadores conservar ancho de banda en el manejo del servicio.

I.2.2. Reseña del proceso de señales de video para el manejo sobre IPTV

Debido a la convergencia de la televisión, la telefonía y el acceso a internet, varias tecnologías de compresión están desempeñando un papel fundamental en el crecimiento de nuevos productos y servicios, para operadores y proveedores de equipos.

Antes de definir codificación en si, es necesario comprender el proceso que ocurre antes de que el video llegue a esta misma etapa.

El proceso es relativamemente sencillo e implica el uso de la cámara para tomar y capturar el contendio del video. Una vez capturado el video, normalmente en formato analógico, el contendio del video debe pasarse a otro proceso llamado digitalización, para convertir la señal analógica continua a una serie de bits digitales. Para establecer el proceso de conversión digital se utiliza un dispositivo de hardware llamado Conversor Analógico a Digital (A/D).

El proceso de conversión analógico digital implica técnicas de muestreo y cuantificación, que se aplican al procesamiento de la señal. Como su nombre lo indica, el muestreo se refiere a un número de muestras tomadas de la señal analógica entrante, donde actúa la frecuencia de muestreo, que se mide típicamente en segundos.

La cuantificación es la parte del proceso de conversión, e implica la asignación de un número de bits a cada muestra tomada de la señal. Una vez en formato digital, el flujo de video sin comprimir esta listo para la codificación.

I.2.2.1. Proceso de Codificación IPTV

La codificación utiliza una serie de dispositivos especializados llamados codificadores. El proceso de codificación en un centro IPTV es algo desarrollado y generalmente lleva tres etapas separadas:

- Se recibe una trasnmisión de video de una fuente en particular. El formato de esta fuente puede variar, y puede ir desde señales analógicas de baja calidad, hasta transmisiones digitales de alta calidad.
- Una vez recibida la señal, el codificador aplica un esquema de compresión particular al contenido. Hay ciertos esquemas de compresión utilizados por los codificadores.
- Una vez comprimido el video, esta listo para su transmisión. La preparación implica que el contenido se inserta en paquetes de datos

I.2.2.1.1. Ventajas de la Codificación IPTV

- Una gran reducción en la cantidad de espacio en el disco duro para almacenar el archivo de video.

- El procesamiento del video sin comprimir requiere de una cantidad significativa de poder computacional. De forma alterna existen aplicaciones para IPTV que permiten la aplicación de técnicas de compresión específicamente sobre fragmentos de contenido de video.

- Debido al hecho de que el tamaño del archivo de video comprimido sea más pequeño que el original, reduce la cantidad de tiempo necesaria para enviarlo a través de la red.

- Se pueden utilizar conexiones de banda ancha con capacidades relativamente bajas, para la entrega de contenido IPTV. Por ejemplo, un flujo codificado de un canal SDTV[7] ocupará aproximadamente 1,5 Mbps, mientras que un canal HDTV ocupará aproximadamente 8 Mbps. En comparación con los estándares de compresión más antiguos, que requieren 3,5 Mbps[8] y entre

20 y 25 Mbps para HDTV, los beneficios de las técnicas de compresión modernas son evidentes.

I.2.2.1.2. Desventajas de la Codificación IPTV

- El proceso de compresión y descompresión de una señal introduce retrasos en el procesamiento.
- Parte de la información se descarta durante el proceso de compresión, por lo que la calidad de la imagen será menor que la de la señal sin comprimir.
- La transcodificación de señales entrantes y el cambio de un formato de compresión a otro pueden afectar la calidad de la señal.

I.2.2.2. Codificación IPTV en Tiempo Real

La codificación en tiempo real prevee la existencia de compensaciones con respecto a las desventajas anteriormente mencionadas.

La compresión en tiempo real permite a los proveedores de servicios IPTV transmitir varios canales de audio y video de alta calidad, a través de una red de banda ancha IP.

Entonces, a partir de la compresión, surgen algunas definiciones como: la compresión reduce el tamaño original y elimina secciones de la imagen. Sin embargo, esta característica no influye en la calidad de reproducción final, debido a que el ojo humano no puede detectar todos los patrones de la imagen,

[7]SDTV pertenece a una señal de TV de definición estándar que tiene sus primeros antecedentes en señales analógicas de 480 líneas. Actualmente esta puede transportar señales de tipo analógica o digital de hasta 576 líneas y con una relación aspecto de entre 4:3 a 16:9.
[8]Mbps es la métrica utilizada para medir cuantos datos en Mega bits por segundo viajan en la red desde un lugar origen a un destino.

cuando estos están representados a través de un video. Por ejemplo, una compresión de 100:1 significa que el tamaño del contenido original se ha reducido en un factor de 100.

Desde la utilización de los métodos de compresión sin pérdida es que permite al IPTVCD[9] (por ejemplo, un STB[10], consumer device) del cliente recrear la imagen original perfectamente en una pantalla.

Pero, sin embargo, esto no es cierto, porque esto sería visto como una ocurrencia rara, debido a que en las redes IPTV prácticamente todas las técnicas de compresión introducen una cierta cantidad de pérdida durante el proceso de codificación. Por eso, es importante considerar que los algoritmos de compresión sin pérdidas son principalmente usados para codificar imágenes fijas y no videos en vivo.

Hay que considerar que durante el proceso de codificación con pérdida, cierta información de la imagen se destruye. Por lo tanto, el decodificador IPTVCD no puede recrear completamente la imagen original, que se generó durante el proceso de digitalización. Asimismo, hoy en día, los algoritmos de compresión con pérdida estan diseñados para garantizar que solo se destruyan cantidades limitadas de datos durante el proceso de codificación.

I.2.2.3. Compresión del video mediante IPTV

El video en su forma más básica es una secuencia de imágenes, que se muestran en orden secuencial. El término técnico para una de estas imágenes de video es "un cuadro" (fotograma). Asimismo, estos en las tramas pueden identificarse dentro de flujo de bits mediante un encabezado. El ojo humano tiene una capacidad de percepción de unos 25 fotogramas por segundo (fps[11]).

Por lo tanto, no tiene sentido transmitir a una tasa más alta, porque los televidentes no notarían la diferencia las técnicas de compresión reducen el

[9]IPTVCD es la terminología usada para referirse en parte al la central de datos que maneja el servicio de IPTV.
[10]STB esta terminología es explicada con mas detalle en la sección III.1.5.
[11]FPS son las primeras letras de las palabras en ingles de "frame for second" y se refiere a los cuadros por segundos de las imágenes.

tamaño del video original, y lo hacen mediante dispositivos del video original, y llamados codificadores de video, que reducen el contenido en cada uno de sus cuadros, mientras mantienen un alto nivel en la calidad de imagen.

Desde la primera etapa de compresión que implica un proceso de sub muestreo, que básicamente consiste en reducir el tamaño de cada fotograma. La reducción del tamaño de la trama elimina una cierta cantidad de bits, lo cual disminuye el ancho de banda para transportar la señal.

Sin embargo, este proceso no esta exento de inconvenientes. Por ejemplo, la reducción del tamaño del fotograma, a menudo, puede producir problemas de tipo relación – aspecto, cuando esta se muestra en un televisor de baja resolución.

A partir de la segunda etapa de compresión, donde en una señal de video produce divisiones de sus cuadros de imagen en bloques de 8x8 píxeles, siendo estos la unidad mas pequeña en el algoritmo MPEG[12].

Siendo, entonces, que los bloques pueden dividirse en tres tipos diferentes:

- ✓ Luminancia en escala de grises (Y).
- ✓ Cromancia roja (Cr).
- ✓ Cromancia azul (Cb)

Los tipos de cromancia contienen información sobre los diferentes colores de la imagen, mientras que la luminancia lleva la parte blanco y negro de la imagen.

I.2.2.3.1. Compresión del Video Mediante MPEG en IPTV

La tecnología MPEG es un estándar de compresión utilizada en los sistemas de televisión por satélite, cable y terrestre. MPEG es un acrónimo de Moving Pictures Experts Group, y representa una asociación industrial, que se formó para ayudar a desarrollar técnicas de compresión adecuadas para la transmisión de video. Desde su fundación, se han propuesta una familia de estándares de compresión de video y audio. Por ejemplo, desde los más importantes: MPEG-1, MPEG-2, MPEG-4, MPEG-7, MPEG-21. Además, de otras especificaciones que describen estándares de contenido, entrega y

consumo. Para los servicios IPTV, de los estándares propuestos anteriormente, los más utilizados son MPEG-2, que incorpora las categorías de audio y video, mientras que también utiliza MPEG-4 parte 10, que es el que utiliza el código H.264[13] que se estudiará mas adelante.

La lógica de MPEG es la aplicación de los siguientes mecanismos de resolución: Primero se aplican dos pasos preliminares en el proceso, el primero es llamado de sub muestreo y el segundo consiste en la compresión de la señal en bloque de 8x8 píxeles.

Una vez que se completa los procesos anteriormente mencionados, MPEG realiza una función matemática llamada de transformada de coseno discreta (DCT), para cada bloque de video de forma independiente. El proceso de DCT consiste en separar el bloque del video en partes de diferentes importancias. Las partes consideradas importantes se conservan, para su posterior procesamiento, mientras que las partes restantes se desechan. Este enfoque asegura que el ojo humano no note la eliminación de las partes menos importantes del bloque del video, mientras limita la velocidad de bits en forma general.

La segunda parte del proceso de compresión MPEG se llama cuantificación. Este proceso consiste en la reducción de bits, que se requieren para representar los diversos bloques contenidos en el cuadro imagen. Dependiendo del nivel de cuantificación aplicado por MPEG, este puede ser mejor o peor. Por ejemplo, si el nivel es alto es por que se ha eliminado una buena cantidad de píxeles. Sin embargo, el resultado del mismo puede ser una degradación de la calidad de la imagen final. El caso contrario es el opuesto debido a que si no existe un nivel alto de cuantificación es por que no se ha eliminado una cantidad significativa de píxeles. Por lo tanto, la calidad puede ser algo mejor.

[12]MPEG es un estándar de codificación de audio y video.
[13]H.264 es sinónimo de MPEG-4 parte 10 es un estándar de compresión de alta definición desarrollado por la ITU-T

I.2.2.4. Velocidades por Métodos de Compresión

Las velocidades de bits emitidas por un codificador se dividen en dos categorías: velocidad de bits constante (CBR) y velocidad de bits variable (VBR).

Como su nombre lo sugiere, los flujos CBR MPEG-2 operan a un rendimiento de datos de ancho de banda constante, independientemente de la complejidad del contenido del video. Esta velocidad de bits es ideal para redes DSL[14], que normalmente proporcionan conexiones de ancho de banda fija.

La generación de un flujo CBR se logra ajustando los parámetros de cuantificación del codificador. Asi mismo, la incapacidad de los flujos CBR, para ajustarse de acuerdo con la complejidad de la imagen, es un inconveniente que los proveedores de servicios IPTV deben tener en cuenta.

Sin embargo, las tramas VBR MPEG-2 se codifican utilizando diferentes velocidades de bits. Por lo tanto, una trama compleja requiere de una gran cantidad de bits, mientras que una menos compleja utiliza una cantidad de bits menor, para representar dicha imagen. El sector de la industria del cable a menudo usa VBR en combinación con un proceso llamado multiplexación estadística, para "ajustar" múltiples servicios de televisión digital en el mismo rango de frecuencia, en el que alguna vez estuvo ocupado por un solo canal analógico. La multiplexación estadística funciona según el principio de que las tasas de bits de entre varios canales, que se ejecutan en una misma conexión, en una instancia en particular, en el tiempo, fluctuarán y diferirán.

Por lo tanto, si existiera un canal que ejecutase una serie de cuadros, que representara una escena de deportes de alta acción, requiere de una alta tasa de bits, para transmitir ese detalle por la red. Mediante el uso de multiplexación estadística, este canal codificado en VBR tomaría prestada capacidad de ancho de banda adicional, requerida por otras tramas de otro canal RF

[14]DSL (Digital Suscriber line), línea de abonado digital para acceso a internet mediante par trenzado de hilos de cobre de la red telefónica conmutada.

compartido, que esta transmitiendo una escena simple, con tipos de tramas de baja capacidad en esta instancia particular de tiempo.

Asimismo, cuando finaliza la escena de acción, la velocidad VBR de ese flujo caerá a tasas normales, o por debajo del promedio, y cualquier exceso de bits disponible de ese flujo puede ser utilizado por otros flujos dentro del canal RF compartido. Por lo tanto, vale aclarar que los picos de un programa pueden corresponder con los valles de otros programas. En conclusión la multiplexación estadística es una técnica que se utiliza cuando la tasa de bits general, para los distintos flujos del canal, permanecen dentro de una tasa de bits constante, o cuando se presenta dicha situación, por otro canal particular del medio RF compartido, que requiera ancho de banda extra en un momento determinado.

Es necesario tener en cuenta que muchas de las arquitecturas de red implementadas, para transportar servicios de IPTV, utilizan el modo de operación de video conmutado. Para este tipo de arquitectura, cada flujo opera de forma independiente, y no se combinan en un canal compartido con otros flujos. Por lo tanto, la técnica CBR es el modo de operación predeterminado para el transporte de video, a tráves de redes basadas en DSL. Aún así, durante los últimos años se han desarrollado nuevos esquemas de compresión con mejores capacidades, con el fin de comprometerse en la entrega de video, a través de redes con ancho de banda limitado. Justamente uno de esos estándares es MPEG-4 parte 10, o más conocido como H.264.

I.2.2.5. MPEG-4 para IPTV

El despliegue de MPEG-4 requiere de equipo especializado en el centro de datos IPTV, y la introducción de nueva tecnología de decodificación en el IPTVCD. Hasta los últimos años, uno de los principales inconvenientes de MPEG-4 ha sido el hecho de que se requiere de potencia para el procesamiento, y de memoria adicional, para decodificar contenido de video basado en este formato. Sin embargo, con el avance de la tecnología basada en semiconductores, MPEG-4 se ha convertido en la tecnología de compresión de punta para los sistemas de Centros de Datos IPTV.

I.3. MODELO DE COMUNICACIONES IPTV

El modelo de comunicaciones IPTV es un marco de trabajo de red compuesto por siete capas conceptuales (siendo una de esta opcional), que se apilan una encima de la otra, según Arellano [7].

Los datos de video se transmiten, por el modelo de una capa a la siguiente, en el dispositivo de envio, hasta que se transmite a través de la red de banda ancha por los protocolos de la capa física. Los datos llegan a la capa inferior del IPTVCM[15] residente en el dispositivo de destino, y viajan por su IPTVCM. Por lo tanto, si un codificador tiene contenido de video, para transferir a un dispositivo de consumo IPTV, debe de pasar este material a través de la estructura en capas IPTVCM en ambos dispositivos.

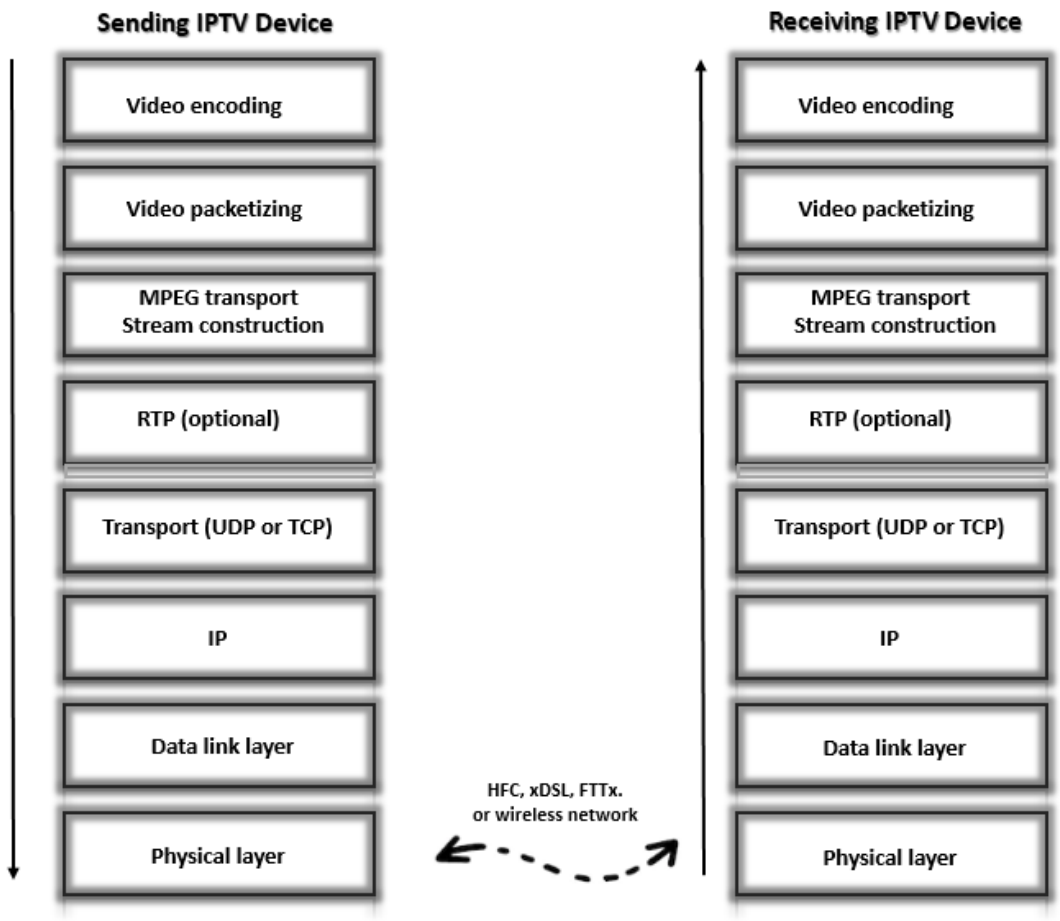
Cada capa del modelo de comunicaciones es generalmente autónoma y tiene resposabilidades específicas. Una vez que se ha cumplido cierta responsabilidad en una de las capas, los datos de video pasan a la siguiente capa del IPTVCM. Cada capa agrega o encapsula alguna información de control adicional a los paquetes de video durante el procesamiento. Esta información de control incluye instrucciones específicas y, normalmente, se formatea como encabezados. Asi es que en el extremo remoto, que los datos pasan por el modelo de comunicaciones a la aplicación receptora.

Bajo este enfoque, se establece una conexión de comunicaciones lógica entre las capas correspondientes o pares. Las siete, y en algunas implementaciones, las ocho capas de IPTVCM se pueden, por lo tanto, clasificar en dos categorías: como capas superiores e inferiores. Las capas superiores se ocupan de tratar con las aplicaciones y formatos de archivos específicos IPTV, mientras que los niveles inferiores del modelo se ocupan principalmente del transporte real del contenido.

[15]IPTVCM, referente al modelo de comunicación de la plataforma IPTV.

I.3.1. Gráfica y Conceptos sobre el Modelo de Comunicaciones IPTV

A continuación, se presenta en forma gráfica los detalles de las principales capas y, posteriormente, los conceptos que definen cada una de las mismas:



La Figura Nº 1: Presenta el modelo de comunicaciones IPTV
Fuente: https://docplayer.es/23341563-A-componentes-de-un-sistema-iptv.html [8]

I.3.1.1. Capas del Modelo de Comunicación IPTV

Capa de codificación de video: El proceso de comunicación comienza en la capa de codificación, donde se comprime la señal analógica o digital sin comprimir, y se emite un flujo elemental MPEG desde el codificador. Un flujo elemental se lo puede definir como una señal digital continua en tiempo real.

Así mismo, existen diferentes flujos de corrientes elementales. Por ejemplo, el audio codificado con MPEG se denomina "flujo elemental de audio". Entonces, si se definiera de forma general un flujo elemental, este se lo declararia como cualquier flujo de tipo digital, que se obtiene de la salida del codificador en un estado de "sin procesado".

La transmisión, generalmente, se organiza en cuadros de video, para esta capa del modelo IPTVCM. Los tipos de información que puede incluir un flujo elemental pueden ir desde:

- Tipo de trama y tasa.
- Posicionamiento de bloques de datos en pantalla.
- Relación aspecto.

Los flujos elementales forman la base para la creación de flujos MPEG. Es importante tener en cuenta que esta capa está, efectivamente, dividida en dos subcapas por la especificación H.264/AVC: dividiéndose, entonces en la capa de codificación de video (VCL) y la capa de abstracción de red (NAL). Asimismo, la capa VCL es la encargada de comprimir el video, con la cual la salida de la misma es definida como una serie de cortes de imágenes. Mientras que la capa NAL esta organizada en varios paquetes discretos llamados "unidades de tipo NAL".

Capa de empaquetado de video: Una vez obtenido los flujos elementales de audio, video, y datos, para que estos se transmitan a través de una red digital, es necesario que cada flujo elemental se convierta en un flujo intercalado de paquetes (Packetized Elementary Stream) PES, con una cierta marca de

tiempo distintiva. Una secuencia de tipo PES contiene un solo tipo de dato de una fuente.

Un paquete PES puede ser un bloque de tamaño fijo o variable, con hasta 65536 bytes por paquete. Esto incluye una asignación de 6 bytes para el encabezado con respecto al resto del paquete, y es utilizado para transportar contenido.

Debido a la naturaleza de las conexiones en red, el orden o la secuencia de los fotogramas de vídeo, en el que se envían desde el centro de datos IPTV, puede diferir en orden en los que estos se reciben en el destino.Por lo tanto, para ayudar a la sincronización, los sistemas basados en MPEG, a menudo, marcan el tiempo de los diversos paquetes PES, que forman parte de un flujo de video particular.

Existen dos tipos de sellos de tiempo que se pueden aplicar a cada paquete PES: Sello de Tiempo de Presentación (PTS) y el Sello de Tiempo de Decodificación (DTS).

- PTS: Un PTS es un valor de tiempo de 33 bits, que se establece en el campo de encabezado PES. El propósito de aplicar un PTS a cada paquete es definir cuando, y en que orden, se debe presentar el video al espectador.

- DTS: El propósito de aplicar un DTS a cada paquete es de indicar al decodificador IPTVCD cuando procesar los paquetes.

Capa de construcción de flujo de transporte: La siguiente capa del modelo de comunicación IPTVCM se ocupa de la construcción de un flujo de transporte, que consiste en un flujo continuo de paquetes. Estos paquetes, comúnmente llamados paquetes TS, se forman al dividir los paquetes PES en paquetes TS de tamaño fijo de 188 bytes, y que estan referenciados en base a tiempos independientes. El uso de bases de tiempos independientes, frente a bases de tiempos idénticas, ayuda a reducir el potencial de pérdida de paquetes, o por paquetes corruptos por ruido.

Cada paquete TS contiene uno de los tres formatos de medios: de video, audio o datos. Por lo tanto, los paquetes de transporte no admiten una combinación de medios. Cada paquete TS comprende 184 bytes de carga útil, y un encabezado de 4 bytes.

Los siguientes tipos de datos pueden encapsularse directamente en el campo de carga útil de un paquete de flujo de transporte MEPG.

- Video MPEG-1 (ISO / IEC 11172-2)
- Video MPEG-2 (ISO / IEC 13818-2)
- Audio MPEG-1 (ISO / IEC 11172-3)
- Audio MPEG-2 (ISO / IEC 13818-3)

Asimismo, es necesario entender que esta capa también proporciona una funcionalidad, la cual permite la creación de secuencias de programas. Un flujo de programa es un múltiplex de paquetes PES, que transporta varios flujos elementales, que fueron codificados usando el mismo reloj maestro, o el reloj de tiempo del sistema. Estos tipos de transmisiones se desarrollan para aplicaciones sin errores, como el contenido de video en almacenamiento de discos duros u ópticos. Por lo tanto, rara vez o nunca las transmisiones de programas se utilizan en implementaciones IPTV de un extremo a otro.

Capa RTP (opcional): Esta capa opcional es utilizada por una amplia variedad de aplicaciones IPTV. Actúa como un intermediario entre el contenido codificado en H.264/AVC, MPEG-2 o VC-1, en las capas superiores, y las secciones inferiores de IPTVCM. El protocolo RTP (Real time Protocol) representa el núcleo de esta capa y, a menudo, es el bloque base que admite la transmisión en tiempo real del contenido multimedia, a través de una red IP. Por lo tanto, RTP entrega flujos de audio y video de un extremo a otro, al encapsular el contenido en un formato particular llamado paquete. Cada paquete consta de un encabezado, y los datos IPTV de carga útil. Para mejorar la eficiencia del ancho de banda, la carga útil, generalmente, incluye más de un paquete MPEG-TS.

Capa de transporte: En general, los paquetes RTP forman parte de la entrada de una capa de transporte IPTVCM. También, es importante señalar que es posible mapear paquetes MPEGTS directamente en la carga útil del protocolo de la capa de transporte. Esto evita de forma efectiva la utilización de RTP por completo.

La capa de transporte IPTV, ha sido diseñada para ocultar la complejidad de la estructura de red IP de los procesos de las capas superiores. Los estándares de esta capa garantizan confiabilidad e integridad sobre el enlace de comunicación de un extremo a otro. Si los datos de video no se entregan al IPTVCD correctamente, entonces, la capa de transporte puede iniciar la retrasmisión. Alternativamente, puede informar a las capas superiores para que puedan tomar la acción correctiva necesaria.

Los protocolos que esta capa dispone, para establecer las funciones anteriormente nombradas son TCP[17] y UDP[18], siendo los dos protocolos con mayor importancia empleados en la capa de pila de comunicaciones IPTV. Aún teniendo en cuenta las diferencias que existen entre ambos, para el manejo de los paquetes de datos IPTV sobre la red.

[16]TCP este protocolo de capa de transporte puede trabajar sobre el protocolo IPTV y entre otras cosas permite proporcionar integridad y disponibilidad (ya que es un protocolo orientado a conexión), sin embargo, este agrega tiempo extra en el manejo de datos en la red cuando se trata de retransmitir datos en una reproducción continua de un video en stream.

[17]UDP El protocolo de transporte de tipo no orientado a conexión, cuando se trata de enviar los datos a un grupo de máquinas individuales este puede ser mas eficiente que TCP debido a que meneja mejor el protocolo multicast que es con lo que se facilita el servicio.

CAPITULO II

II.CASO DE ESTUDIO DE UNA RED IPTV

En el presente capitulo se utilizará un Caso de Estudio para avanzar con más detalle sobre las definiciones, las topologías, las configuraciones, y en las características de los dispositivos de una arquitectura IPTV típica.

II.1. DISEÑO DE LA RED IPTV

Para dar comienzo al trabajo de implementación de un sistema IPTV, se debe realizar un análisis, e ir enumerando los componentes de configuración de red implementados en la operadora.

Como se puede observar en la figura N°2, se muestra la red y la concentración del tráfico que se realiza a través de un switch principal. De acá la definición que lo denomina "Switch IPTV", que cumple con la función de encaminar el tráfico por las distintas VLANs[18] a los distintos puntos de la red.

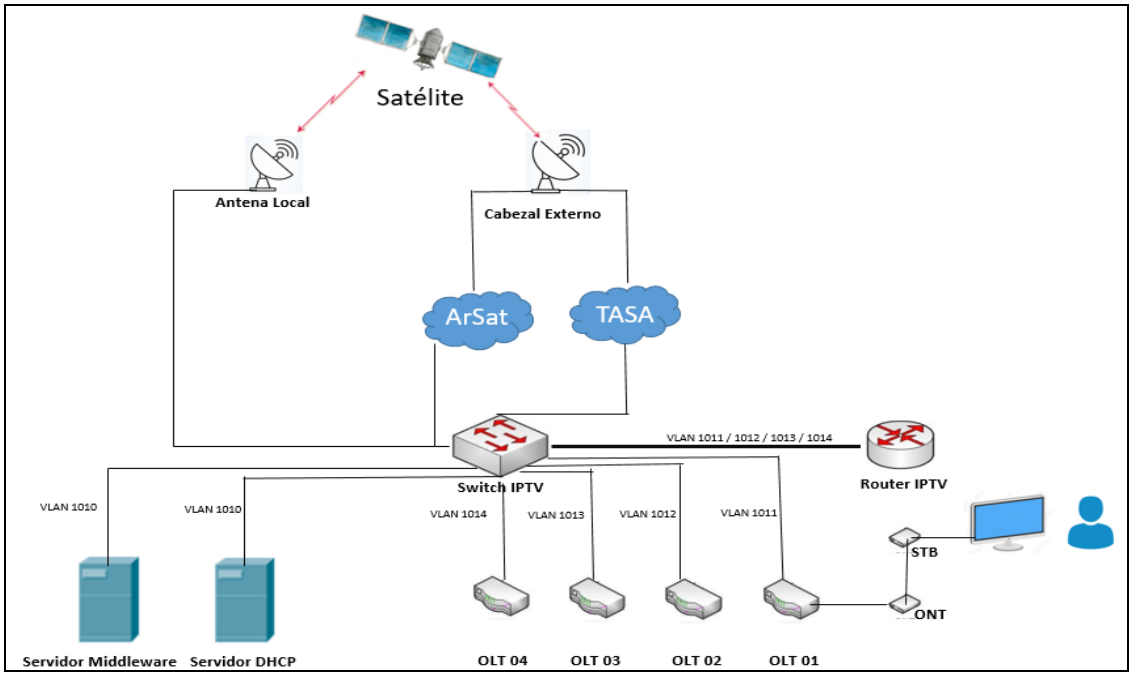


Figura N° 2: Muestra la topología de la red IPTV.

Si se lee esta topología desde arriba hacia abajo se encontrarán distintos tipos de dispositivos:

En la parte superior vemos aquellos dispositivos originadores de señales: las antenas locales o externas, que toman las señales satelitales. Es a través de un proceso de codificación que estas antenas locales o externas envían la señal al switch de forma directa o indirecta con formato IP multicast. Cuando nos referimos a un envío de señal de forma indirecta es porque las señales pueden bajarse o procesarse en un punto remoto, y enviarse a un destino a través de internet, o de la misma manera por un enlace punto a punto. En este ejemplo, las señales recibidas desde el "cabezal externo", llegan a destino a través de enlaces punto a punto por los proveedores ARSAT[19] y TASA[20].

Las señales son recibidas en el switch IPTV, y re-enviadas al router presente en la red (router IPTV). Dicho dispositivo será el encargado de efectuar el ruteo multicast de dichas señales, según sean solicitadas por los abonados finales.

Siguiendo con la lista de dispositivos que se presentan en la red, se pueden observar dos servidores: el "Servidor Middleware" y el "Servidor DHCP". El primero es el que contiene el software que realiza la presentación al usuario final; el mismo presenta la grilla de canales, y administra y gestiona los abonados y dispositivos Set-top-box, conocidos por la sigla STB.

El servidor DHCP se encarga de asignar direcciones IP de red, que tendrán los dispositivos STB, así como de pasar los parámetros de conexión, tales como la dirección a la cual tienen que conectarse, y otros parámetros de seguridad.

Por último, se encuentran los equipos distribuidores del servicio de abonado final.

Para el caso de distribución del servicio se observa que por un tema de calidad

[18]VLANs Una red LAN puede tener un tamaño considerable. Sin embargo, la misma puede dividirse en LANs lógicas creadas a partir de un subnetting de red y manejándose a través de un switch administrable por capa 2.
[19]ARSAT (Empresa Argentina de Soluciones Satelitales Sociedad Anónima) https://www.arsat.com.ar/
[20]TASA (Telefónica de Argentina Sociedad Anónima) https://www.telefonica.com.ar/

IPTV se brinda mediante tecnología FTTH[21] bajo la norma GPON[22] (Gigabit Passive Optical Network).

Los distribuidores de servicio son llamados OLT[23] (Optical Line Termination). Estos elementos son aquellos situados en el sitio de la central de equipamiento. De el parten las fibras ópticas hacia los usuarios (cada OLT suele tener capacidad para dar servicio a varios miles de usuarios).

[21]FTTH Fiber to the home, es una nomenclatura que especifica fibra hasta la casa del abonado.
[22]GPON es una tecnología de red óptica con capacidad de gigabit que puede llegar, al igual que el anterior, a la casa de los abonados.
[23]OLT es el dispositivo de red colocado por el operador de manera de alcanzar la última milla y que permite la retransmisión de la señal a multiples clientes.

II.1.1. Cálculo de implementación de Servicio por Abonado

Para la implementación del servicio por abonado puede considerarse un despliegue de máximo 3 STB por equipo terminal de FTTH (ONT – Optical Node Terminal). Este cálculo se desprende debido a que cada puerto GPON puede disponer de 2,5 Gbps[24] de bajada, y al tener en cuenta como una posibilidad 1 nivel de splitteo por 64 (subdivisiones del pelo principal de fibra). Esto quiere decir que pueden llegar como máximo por abonado (64 abonados por puerto GPON) 40 Mbit/s aproximadamente.

Teniendo en cuenta que 20 Mbps se utilizan para el servicio de internet, quedarían otros 20 Mbps para el servicio IPTV.

Actualmente cada señal HD (720P) ocupa un ancho de banda de 6,5 a 7 Mbps (al tratarse de un bit rate variable), por ende de 20 Mbps se pueden tener un máximo de 3 señales diferentes (lo cual hacen un total de aproximadamente 20 Mbps).

Esta consideración es importante ya que establece un límite del servicio para un supuesto escenario. Dicho límite podría cambiarse, si se reajustan los planes de ancho de banda de internet, o se disminuye el nivel de splitteo; también si se implementa la norma xGPON, la cual permite direccionamiento de hasta 1 Gbps por puerto.

[24]Gbit/s es una medida que sirve para determinar el caudal del transporte de datos desde un origen a un destino, y que suele expresarse con la unidad mínima de datos de 10^9 bits.

II.1.2. Orígenes de las Señales IPTV Sobre IRD o Cabezales Externos

Las fuentes de las señales de origen multicast pueden ser diversas: IRD locales, encoders locales o remotos, o señales bajadas de internet. También se podría tener el caso puntual, donde la señal se genera localmente, ya sea porque existe un canal de televisión, o algún tipo de generador de contenido, que entrega la señal cruda o de multicast generada.

En este caso de estudio, como se observa en la topología de la Figura Nº 2, el origen de las señales son los satélites. Por lo tanto, las señales se bajan desde estos dispositivos, y son recibidas por medio de parabólicas situadas de manera local o remota. Estas a su vez estan conectados a un dispositivo que recibe la señal. Tal dispositivo es conocido profesionalmente como IRD (Integrated Receiver Decoder) que permite procesar la señal.

Para un caso de estudio como este, el dispositivo IRD podría ser de marca Dexing modelo NDS3508TX SERIE N, y la parabólica un plato de 3 metros de diámetro, LNB banda C.

Figura N° 3: Muestra el dispositivo IRD marca Dexing.
Fuente: http://english.dsdvb.com/

II.1.2.1. Funciones Partes del IRD

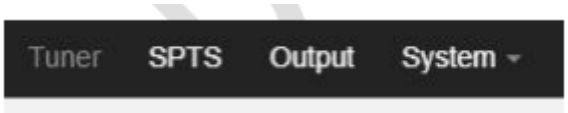


Figura N°4: Indica el panel de control del IRD.
Fuente: http://english.dsdvb.com/

Seteo Tuner: Este setup permite la visualización de los distintos canales (hasta 8 para este modelo), y con los datos en frecuencia, tanto del satélite como del LNB, observar la calidad y la fuerza de la señal. También puede notarse el Symbol Rate, el cual indica la cantidad de información digital por segundo que emiten los datos en un canal, expresado en Ksps[25].

[25]Ksps (Kilo Samples per second), permite a una señal/les analógica/s ser medida a través de un flujo de números, por ejemplo un/os flujo/s analógicos se pueden representar mediante la amplitud de su señal, por lo tanto, el Ksps seria la medida tomada en números de la cantidad de muestras que existen en la amplitud de una determinada señal. De manera simplificada el primer término "k" proviene de kilo y hace referencia a las miles posibles muestras que existen en estos periodos de amplitud de señal.

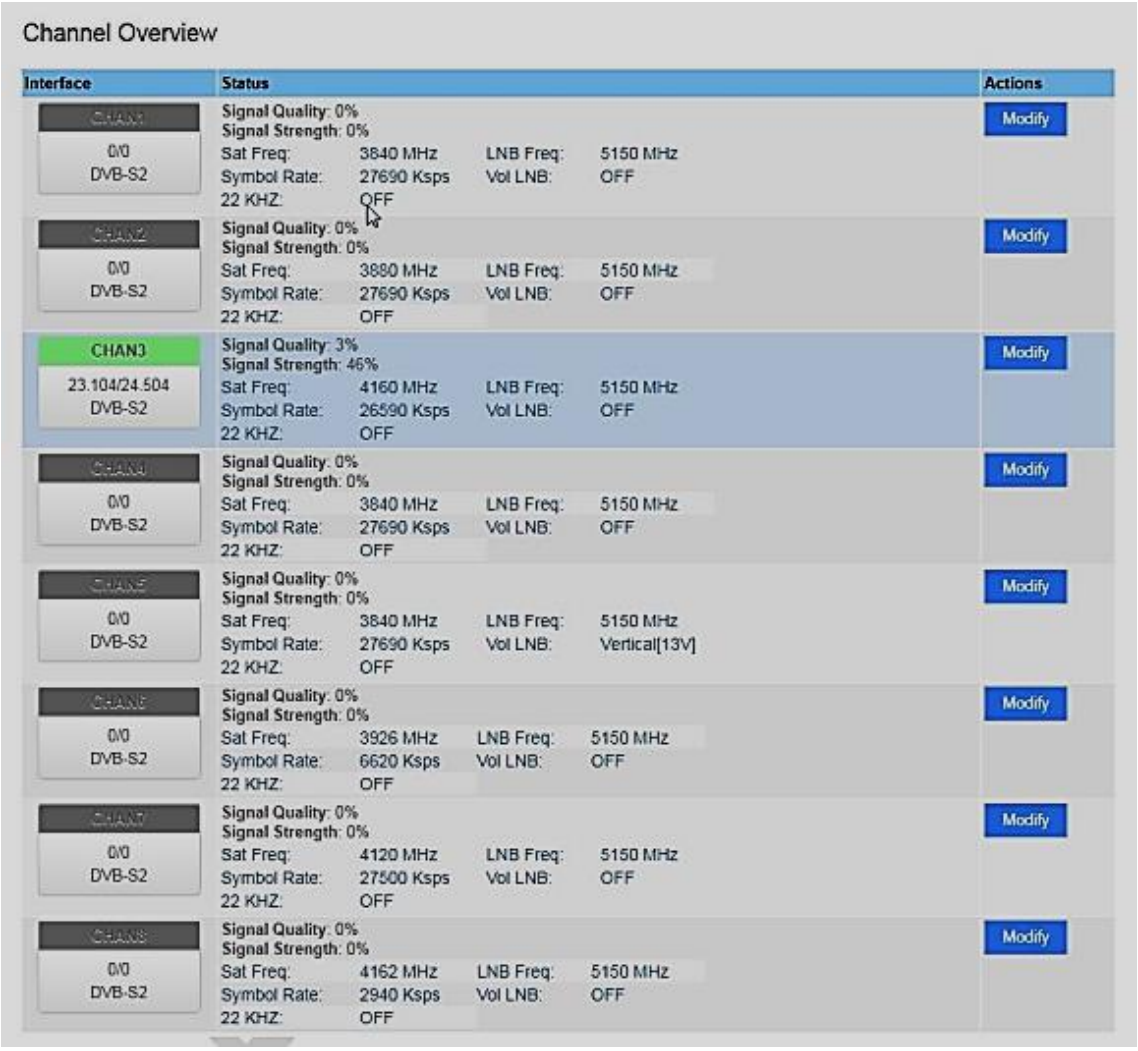


Figura N° 5. Se presenta la configuración en Tuner del IRD.
Fuente: https://petriko.kolev.org/

Seteo SPTS: El Transport Stream es otra de las funciones del setup del IRD. Básicamente es un protocolo para audio, video y datos. Así mismo, los flujos binarios de video y audio de cada programa se comprimen de tal manera que cada uno de ellos forma una "corriente elemental". A su vez, estas corrientes elementales se estructuran en forma de paquetes llamados PES (Packetized Elementary Stream).

A continuación, estos paquetes pasan por un multiplexor, donde el resultado puede diferir entre dos posibles salidas: la primera de tipo "corriente de programa" (PS, Program Stream), o puede haber otra denominada "corriente de transporte" (TS, Transport Stream).

De esta manera, con las opciones del IRD se pueden ver las distintas señales dentro del canal.

PMT: Asocia cada programa actual en el transport Stream. Y cada información necesaria correspondiente a los elementary stream.

- PID: Asociado al trama fundamental.
- El tipo de trama fundamental en si (video, audio y datos)
- Y los descriptores asociados a la trama fundamental.

PCR (Program Clock Reference): Este es un valor que da la información de sincronización respecto del receptor, para establecer la decodificación con el video, audio y los datos bajo un esquema de reloj de 27 MHz (Mega Hertz).

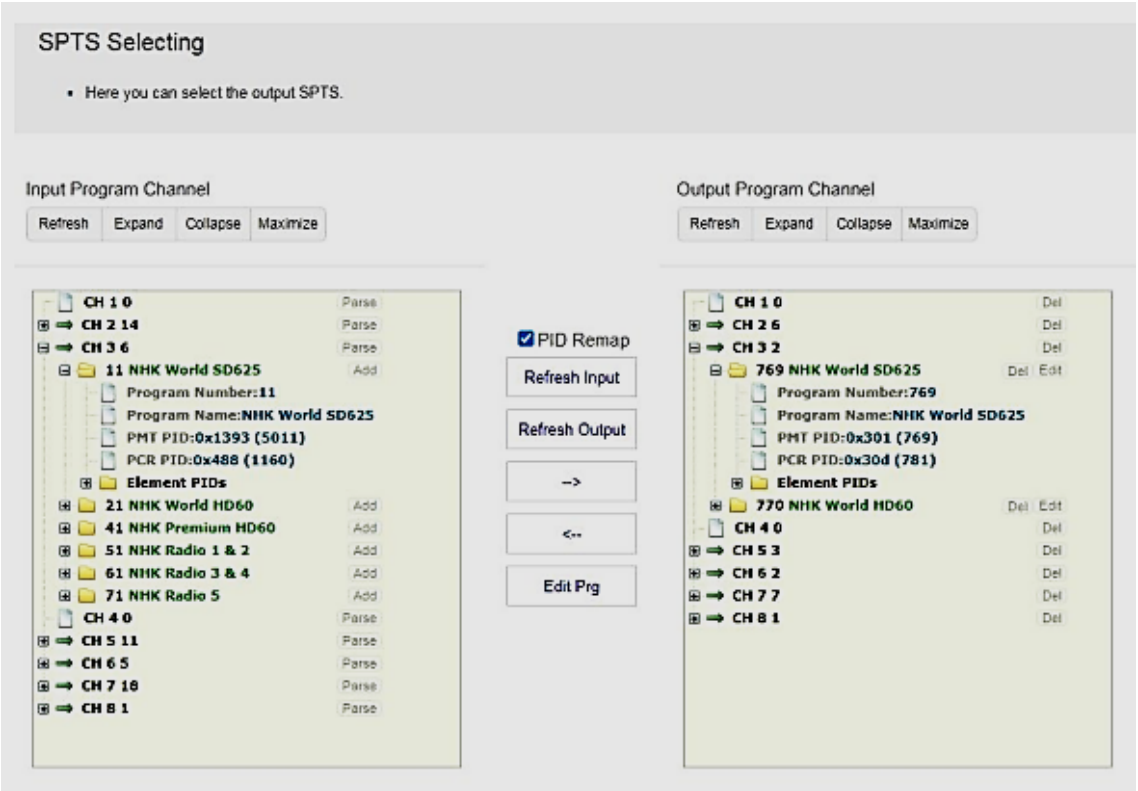


Figura N° 6: Especifica la configuración del STPS en el IRD.
Fuente: http://english.dsdvb.com/

Seteo Output: La pestaña permite la visualización de los datos en la señal de salida. Este display muestra entre otras cosas la dirección de red, el número de subred, y si es una señal de tipo multicast. Además se puede mostrar el número de puerto y la dirección MAC[26] de la interfaz de salida.

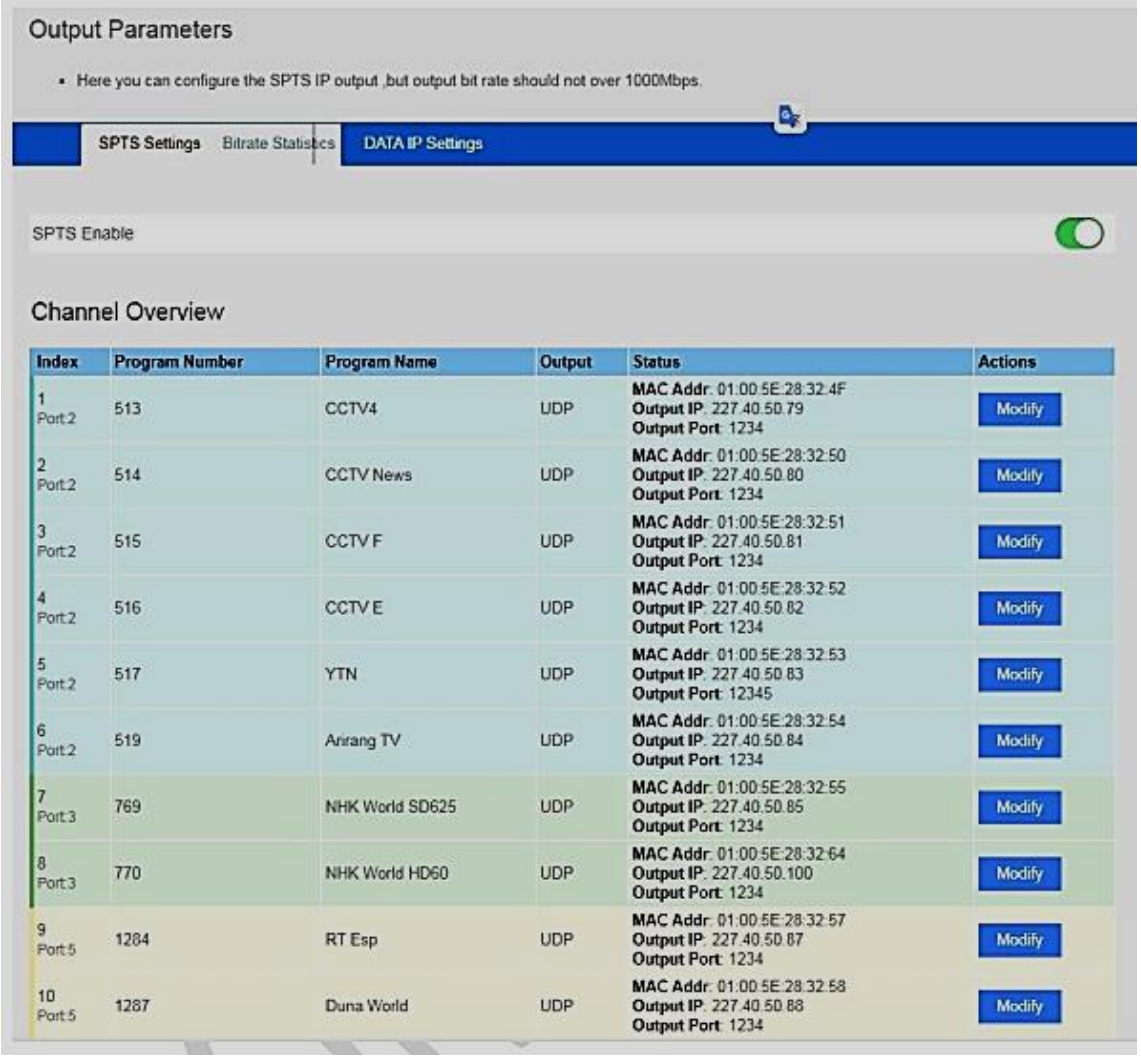


Figura N° 7. Se presenta la configuración Output en el IRD.
http://english.dsdvb.com/

[26]MAC es un término que se refiere a la dirección física de la tarjeta de red de un dispositivo de red. Es una dirección haxadecimal que difiere de la dirección lógica IP de capa 3. Esta se encarga de manejar datos a nivel de capa dos del modelo OSI.

II.1.2.2. IRD sobre Cabezales Externos

El sistema IRD de cabezales externos proporciona servicio de señales de TV brindados, generalmente, por una empresa externa. Dicho de otra manera, esto se puede deber a que los costos de montar un cabezal propio son muy altos, si la cantidad de clientes es baja.

Para un caso de estudio real se podría hablar de un máximo de 2000 clientes de IPTV, según un despliegue de sistema de red por FTTH. Este es un valor relativamente bajo, si se tiene en cuenta que para un solo equipo IRD los canales están contados, una grilla entera puede disponer de 150 canales, y se necesitarían varias antenas parabólicas y, de la misma manera, varios dispositivos IRD. Esto es debido a que cada generadora de contendio (productora /canal) trabaja bajo sus propios seteos de compresión y calidad de imagen. Por lo tanto, la compra de equipamiento podría rondar unos cientos de miles de dólares americanos. Sin embargo, el ISP podría ser miembro de una cooperativa de ISPs, que proporcionaría su propio cabezal, y todo el equipamiento, para realizar el transcoding de las señales, para, de esta manera, enviar las señales, a través de transporte terrestre, entre otros enlaces de tipo punto a punto, a cualquier parte de un amplio territorio.

En la topología del caso de estudio se observan los enlaces preparados que llegan a los destinos de los proveedores ARSAT y TASA. Y que a su vez realizan las entregas de la señal a la localidad que estas manejan.

La entrega de contenido se realiza por duplicado para tener disponibilidad y redundancia, ante cualquier posible corte, o algún tipo de contigencia en el trazado de las redes.

En este esquema, las señales pueden venir tagueadas desde el origen, para que sean recibidas bajo una hipotética VLAN 1010, y así permanecen en el switch IPTV, hasta que son enrutadas al router IPTV y, correspondientemente, tagueadas a la VLAN que corresponde, para ser enviada al destino.

II.1.2.3. Ruteo IPTV y Protocolo Multicast

La función básica del router es la de encaminar las señales multicast de una manera inteligente, mediante la ayuda del protocolo PIM. Es decir, el router solo debe hacer la entrega de las señales a las OLTs que se les solicitan.

Un tipo de router que podría utilizarse en la instalación es un Cisco 7201 de la serie 7200 (es un equipo que prácticamente se encuentra en EOL[27]), alguna versión más actual.

Sin embargo, este modelo de ejemplo, cumple con creces las necesidades requeridas para el ruteo y se mantiene bajo las siguientes especificaciones:

- Posee 4 puertos Gigabit Ethernet SFP, 2 puros de FO y 2 combo.
- También tiene 1 Gb de memoria RAM.
- Una performance o throughput de 2.000.000 pps.
- El procesador es de 1.67 –Ghz Motorola Freescale 7448 processor

Figura N° 8: Muestra el dispositivo de routeo IPTV.
Fuente: https://www.cisco.com/

Este router permite los protocolos IGMP y PIM, para el manejo del enrutamiento multicast bajo el protocolo IP.

[27]EOL. Son los términos en ingles de "End of life", y es una manera de expresar el fin de vida útil de un dispositivo de red.

II.1.2.3.1 MULTICAST

Básicamente, lo que debe conocerse entre un tipo de tráfico unicast y un tráfico de tipo multicast es su funcionamiento [9], y a quien va dirigido. El tráfico de tipo unicast funciona entre un emisor y un único receptor, de manera de establecer un único canal entre ellos. La transmisión de tipo unicast se efectúa desde el servidor a una dirección de red de un usuario en particular, y este hará uso del servicio teniendo un efecto de consumo acumulativo en la red. Por ejemplo, este usuario podrá consumir tanto ancho de banda en Kbps, en un periodo de tiempo, dependiendo de cuanto requiera la codificación del mensaje.

Sin embargo, multicast es un protocolo mucho más eficiente, debido a que con un único paquete, y con una única dirección multicast, son suficientes para que con información de ruteo dada por el protocolo IGMP, sirva como dato para el envío por la red hacia los destinos. Para este caso, la red tiene suficiente inteligencia para replicar los paquetes hacia los host destinos. En cambio sí se tratare de hacer lo mismo con tráfico de tipo unicast el servidor debería enviar paquetes hacia cada destino, con las direcciones de cada host, con lo cual se estaría haciendo mal uso de los recursos de la red, debido a la duplicidad del tráfico desde la red origen.

II.1.2.3.2. Funcionamiento multicast

IGMP es el protocolo que tiene la responsabilidad de anunciar que un nuevo dispositivo de capa 2 desea unirse a un grupo de difusión. Por lo tanto, el dispositivo que tiene la intención de unirse, le envía datos al switch (o conmutador), o a un grupo de switches, o hacia un enrutador o enrutadores de tipo multidifusión, para que los dispositivos de red aprendan que grupos difusión están existiendo en la red.

Para aplicar el concepto de funcionamiento de comunicación debe seguirse la siguiente secuencia: cuando un host está ejecutando una aplicación envía un mensaje con dirección IP: 224.0.0.2, y difunde este paquete a todos los

dispositivos multidifusión cercanos. Básicamente, lo que está reclamando el dispositivo es que lo unan a un grupo de difusión y, por lo tanto, le dice a los dispositivos conmutadores que agreguen este a su número de puerto (ya que tal aplicación requiere unirse a tal grupo difusión).

Los conmutadores que entienden acerca del protocolo y del funcionamiento sobre capa dos, agregan al host en caso del que el protocolo se los pida o, en caso contrario, si requiere que dicho host sea eliminado del grupo difusión, los dispositivos de red lo eliminarán con sus puertos de sus tablas MAC. Dicho de otra manera, la función de los dispositivos de red es la de relacionar el número de puerto de la máquina que lo proporciona, y unirlo con su respectiva dirección MAC a su grupo difusión.

Posteriormente, si los dispositivos de la red local requieren de envío de tráfico de servicios externos, como por ejemplo, de video por demanda, lo que debe de hacer el switch es enviar un informe al enrutador (o Gateway multidifusión de su red local), para que este grupo se difunda a través de IGMP, y se conozca por internet.

Como consecuencia, la función del enrutador multidifusión es de hacer conocer públicamente su grupo difusión a la red WAN, justamente porque este no entiende de direcciones de capa de nivel 2, pero si entiende de direcciones de capa 3. Su función entonces será de mapear este grupo multidifusión a un número IP válido (para que internet lo reconozca), y para que a través del protocolo IGMP pueda comunicar estos equipos enmascarados al exterior.

Para que IGMP haga esta publicación debe de tener en cuenta la existencia de rutas efectivas, por lo tanto el protocolo debe configurarse para que utilizando algún método de enrutamiento, en lo posible, pueda distribuir de manera correcta los datos del grupo y anunciarlos por internet.

Entonces, para que el protocolo distribuya esta información de manera correcta debe ir armando un esquema de distribución de árbol, que indique como se distribuyen los grupos difusión en los routers existentes, para crear una base de datos en cada router de difusión, y que estos se vayan actualizando en la

medida que tengan nuevos integrantes o, en todo caso, para crear la existencia de un nuevo grupo de difusión.

Y por último debe repetirse nuevamente el ciclo. Por lo tanto, el switch y el enrutador multidifusión envían mensajes periódicos dentro de la red local, para determinar si un dispositivo desistió de su grupo difusión o, en caso contrario, si desea unirse a uno nuevo.

II.1.2.3.3. Tipos de Protocolos Multicast

A nivel de Campus:

- **IGMP**: (entre host y routers) protocolo de información de grupos multicast hacia otros routers del dominio [10].

- **IGMP Snooping**: Consiste en que los conmutadores de capa 2 escuchen el tráfico producido por el protocolo IGMP, que viajan de entre los host y los routers. Y gracias al producto de la comunicación, los conmutadores puedan mantener el mapa de tipo multicast y, de esta forma, se pueda dar acceso a los puertos que realmente lo necesiten.

- **PIM Sparse**: Este es un protocolo de ruteo, y es el que permite la construcción del árbol de información de grupos entre un emisor y el receptor del grupo multicast.

Multicast Inter dominio:

- **MBGP:** Es una extensión del protocolo BGP, que admite que el tráfico de diferentes direcciones se distribuyen en paralelo, mientras que el protocolo BGP solo admite trafico unidifusión IPv4. Este en cambio no solo admite tráfico unidifusión y multidifusión IPv4, sino también de IPv6 con unidifusión y multidifusión.

- **MSDP:** Es un protocolo que permite el descubrimiento de otros grupos multicast, que se encuentran en otros routers de otros dominios, o de otros proveedores. Y su finalidad es la de obtener suficiente información de ruteo inter dominio.

II.1.2.3.4. Configuración multicast para el Router IPTV

Para el funcionamiento del enrutamiento multidifusión IP se debe configurar la versión PIM y el modo PIM. De esta manera el software puede reenviar paquetes de tipo multidifusión, y el router puede llenar su tabla de enrutamiento multidifusión.

Se debe configurar una interfaz para que esté en modo denso PIM, modo disperso o modo denso disperso. Una vez preparado el proceso, el router llena su tabla de enrutamiento multidifusión, para poder reenviar los paquetes de tipo multidifusión a sus LANs, que están conectadas directamente, de acuerdo a la configuración del modo preestablecido. Se debe habilitar PIM en una de las interfaces, para que este modo realice el enrutamiento multidifusión. Por lo tanto, una vez habilitado PIM por defecto también se habilita la operación IGMP de esa interfaz.

Es necesario tener en cuenta que si se habilita PIM en varias interfaces y, a su vez, la mayoría de estas no forman parte de la lista de interfaces salientes, cuando la indagación de IGMP es deshabilitada, es muy posible que la interfaz saliente no pueda mantener la velocidad de línea, para el tráfico multidifusión, debido a una replicación adicional innecesaria.

Entonces, una vez que se complete la tabla de enrutamiento multidifusión, al reenviar la señal de pedido de tráfico, desde una LAN, se produce una operación de tipo modo disperso, si existe un RP conocido para el grupo. Si esto es cierto, los paquetes se encapsulan y se envian al RP. Cuando no se conoce ningún RP, el paquete se inunda en modo denso. En caso de que el tráfico multidifusión de una fuente sea suficiente, el enrutador del primer salto puede enviar mensajes hacia el router fuente, para construir un árbol de distribución multicast.

II.1.3. Funcionalidades del Switch IPTV

El switch IPTV utilizado para el caso de estudio es uno de los protagonistas, y que cumple las funciones más importantes para la red, ya que este es el encargado de interconectar todas las VLANs, y el equipamiento de la red. Por lo tanto, es necesario que este dispositivo proporcione alta capacidad de switching. Por defecto, este cuenta con interfaces de fibra de al menos 1 Gbps, para establecer conexiones con las OLTs, y los proveedores de señal.

Para el caso de estudio se seleccionó un swicth ZTE ZXR10 8902E, el cual posee una capacidad de switching de 960 Gbps, y una capacidad de total de (Backplane Capacity) de 3.2 Tbps[28].

En cuanto a las interfaces del switch, este cuenta con dos módulos. Uno con 48 interfaces Ethernet Gigabit, mientras que también posee un módulo de 24 interfaces ópticas SFP Gigabit.

Figura N° 9: Se presenta el dispositivo switch IPTV.
Fuente: https://www.zte.com.cn/

[28]Tbps, Es una unidad de tasa de transferencia de red con equivalencia a 1000 gigabits o lo que es lo mismo 1000000 megabits/segundo.

II.1.4. Servidor Middleware

El servidor middleware es el software encargado de la administración de las señales multicast y OTT, así como también del contenido OnDemand. También es el encargado de gestionar que los usuarios puedan conectarse a la plataforma y a los dispositivos terminales Set-top-box.

Dentro de la variedad de funciones, también permite la configuración de las grillas EPG (Guías electrónicas de Programación), y de los servicios de DRM (Gestión de derechos digitales), así como también de la gestión de cobro.

A continuación se procederá a la muestra de un ejemplo de configuración tipica (sobre todo para aquellos paquetes incluidos en la instalación y los programas que brindan el servicio) siendo, entonces este la plataforma middleware de la compañía Beenius, el middleware en su versión 4.2.5.5-GA.

Configuraciones:

Sistema Operativo:

- ✓ Distribución de Linux: CentOS 6.7/RHEL 6.5 o superior.

El paquete de instalación incluye los siguientes componentes:

- ✓ Java VM.
- ✓ Servidor de aplicaciones JBoss 7.2.0.
- ✓ Base de datos Oracle Express Edition 11g.
- ✓ Web server Nginx 1.8.0.
- ✓ Aplicación Beenius.

Beesmart incluye en su totalidad los siguientes servicios:

- ✓ TV en vivo.
- ✓ EPG.
- ✓ PVR.
- ✓ VOD.
- ✓ Cobranzas.
- ✓ PPV.

- ✓ Aplicaciones.
- ✓ Estadísticas de usuario.
- ✓ Recomendaciones.
- ✓ VoIP.
- ✓ Chat.
- ✓ Pre pago.
- ✓ Ayuda.

II.1.5. Servidor DHCP

El servidor DHCP proporciona el servicio de asignación de direcciones IP a las máquinas de la red. Este puede ser un servidor, una estación de trabajo, o incluso un PC. Como se había mencionado su función es la asignación de direcciones IP en una LAN, o entre varias VLANs.

Entre la información de configuración IP los datos necesarios son: máscara, gateway, DNS, etc. Un servidor DHCP (DHCP Server) es un equipo de red que esta corriendo un servicio DHCP, como se diría en Linux un demonio escuchando peticiones de tipo broadcast. Cuando estas peticiones son oídas, el servidor proporciona una dirección IP con información adicional, si es necesaria.

El tipo de servidor DHCP utilizado es ISC, con un sistema operativo Linux Debian 9.

Como argumento fundamental para asegurar la calidad del servicio, la instalación solo ofrece el servicio FTTH, siendo por lo tanto, que la red de tipo FTTH tiene 4 OLTs activas y, por ende, el bloque esta dividido en 4/16. Esto da la posibilidad de obtener hasta 65534 direcciones por cada segmento de red.

A partir de dicha información, los rangos previamente configurados quedarían en:

15.201.0.0/16 ➡ OLT1

15.202.0.0/16 ➡ OLT2

15.203.0.0/16 ➡OLT3

15.200.0.0/16 ➡ OLT4

A continuación, en la figura Nº 10, se muestran las 4 VLANs, las cuales están divididas una por cada OLT. Por ejemplo, la VLAN 1011 para el OLT1, la VLAN 1012 para el OLT2, la VLAN 1013 para el OLT3 y para la VLAN 1014 el OLT4. A continuación se procede a la muestra detallada del archivo de configuración en /etc/network/interfaces.

```
auto eth1.1011
iface eth1.1011 inet static
        address 15.201.0.1
        netmask 255.255.0.0
        vlan-raw-device eth1

auto eth1.1012
iface eth1.1012 inet static
        address 15.202.0.1
        netmask 255.255.0.0
        vlan-raw-device eth1

auto eth1.1013
iface eth1.1013 inet static
        address 15.203.0.1
        netmask 255.255.0.0
        vlan-raw-device eth1

auto eth1.1014
iface eth1.1014 inet static
        address 15.200.0.1
        netmask 255.255.0.0
        vlan-raw-device eth1
```

Figura N° 10: Muestra la configuración de las VLANs en el directorio /etc/network/interfaces.

En la Figura Nº 11, se procede a mostrar el archivo en el directorio /etc/default/isc-dhcp-server, que proporciona las configuraciones de las interfaces desde las cuales el servidor escuchará las peticiones:

```
  GNU nano 2.7.4              Fichero: isc-dhcp-server

# Defaults for isc-dhcp-server initscript
# sourced by /etc/init.d/isc-dhcp-server
# installed at /etc/default/isc-dhcp-server by the maintainer scripts

#
# This is a POSIX shell fragment
#

# Path to dhcpd's config file (default: /etc/dhcp/dhcpd.conf).
#DHCPD_CONF=/etc/dhcp/dhcpd.conf

# Path to dhcpd's PID file (default: /var/run/dhcpd.pid).
#DHCPD_PID=/var/run/dhcpd.pid

# Additional options to start dhcpd with.
#       Don't use options -cf or -pf here; use DHCPD_CONF/ DHCPD_PID instead
#OPTIONS=""

# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
#       Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACES="eth1.1011 eth1.1012 eth1.1013 eth1.1014"
```

Figura N° 11: Presenta la configuración de las VLANs

II.1.6. Dispositivos finales STB

El set-top-box es el dispositivo final instalado en el domicilio del abonado, que proporciona la señal al computador que posea el abonado. De la misma forma que otros dispositivos poseen un firmware, implementa un Sistema Operativo, pudiendo ser el mismo de tipo Linux o Android. Además, el mismo puede poseer otras funciones, como por ejemplo la posibilidad de permiterle al usuario instalar aplicaciones que desee con contenido: música, juegos, u otros que implementen interactividad.

A continuación, se muestra en la Figura Nº 12 un dispositivo STB, para dar soporte final del servicio IPTV en el abonado. Sus características se nombran a continuación: Sistema Operativo Linux, marca Amino modelo a139, kernel versión 2.6.23.17 desarrollado por STMicroelectronics.

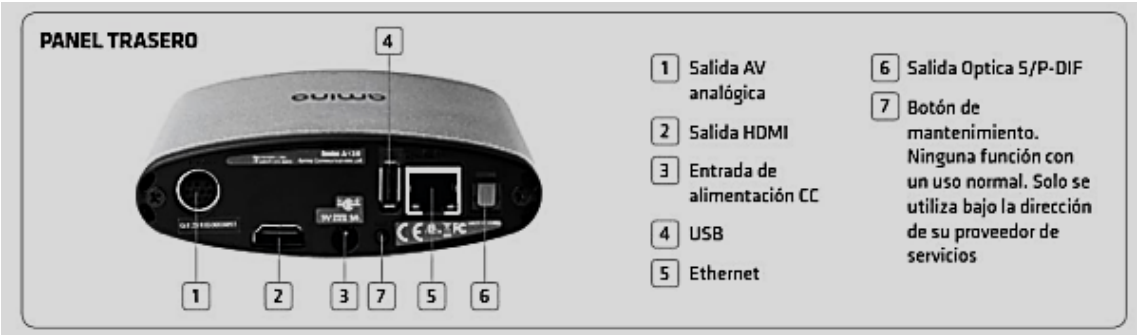


Figura N° 12: Muestra un dispositivo STB y sus propiedades. [11]
Fuente: https://www.gtd.cl/

II.1.7. Controles Necesarios en el Servicio IPTV

Monitoreo de Señales: Cuando se brinda el servicio de televisión, se obliga a proporcionar una alta disponibilidad. Por eso, a diferencia del tráfico de internet, donde este puede ofrecerse sin demanda, el tráfico IPTV debe llevar un control, para evitar que variables externas el usuario final le impidan obtener un sistema de reproducción de video nítido. Por lo tanto, una caída del tráfico multicast produce una incorrecta visualización de la señal, lo cual conlleva multiples reportes y quejas por parte de los abonados.

Una de las soluciones brindadas, es el establecimiento de un segundo enlace, a través de otro proveedor de transporte de señales, lo que permite tener redundancia de las mismas ante alguna incidencia.

Además, se realiza un monitoreo sobre la interfaz primaria y, si es necesario, se procede a realizar un cambio de interfaz, ante la caída de la misma. Este cambio se aplica cuando al monitorearse bajo la plataforma Zabbix[29] logra detectar una falla. El "cambio" se aplica por software, mediante un script bash, utilizando librería especifica.

[29]Zabbix es un aplicativo que permite el control y el monitoreo de redes. Tanto que puede monitorear el estado de los servicios de red, de los servidores y el hardware de red.

CAPÍTULO III

III.DESCRIPCIÓN DE LOS RECURSOS Y HERRAMIENTAS PARA LA EXPERIMENTACIÓN IPTV

En este capítulo se presentan las principales herramientas utilizadas para la experimentación, estudio, y análisis del trafico IPTV, en un ambiente LAN de laboratorio controlado, que es parte de la red de la UTN Mendoza. Se trata del software ffmpeg, de Wireshark, y del IDE Anaconda Spyder. El software ffmpeg es la aplicación que proporcionó las funciones de transmisión de video en vivo y bajo demanda, en distintas codificaciones. Wireshark es el software utilizado para la captura de tráfico IPTV. Y Anaconda Spyder es el IDE[30] que permitió, usando Python, la programación de utilitarios, que facilitaron el análisis del tráfico capturado.

III.1. SOFTWARE FFMPEG SERVER

Es una herramienta de software, o más específicamente un framework, que proporciona una interfaz de comandos, para la transmisión de video bajo streaming, como muestra en la especificación del sitio oficial [12]. Entre otras funciones permite grabar, o convertir audio y video, para ser enviado a través de un host/servidor por una red de tipo TCP/IP[31]. Este framework está formado por varios componentes, que facilitan al mismo tiempo trabajar como servidor de streaming, en conjunto con bibliotecas que apoyan la transmisión del video con su codificación correspondiente, entre otros recursos. Su código fuente esta armado en GNU Linux.

[30]IDE Entorno de Desarrollo Integrado: presenta un entorno de trabajo con soluciones prácticas que resuelve los problemas de la codificación de algún lenguaje de programación para que luego el programa en su versión final pueda trabajar correctamente en el framework que propiamente maneje dicho lenguaje.
[31]TCP/IP TCP e IP son dos protocolos que pueden coexistir para hacer sus operaciones en la red de datos. Uno le presenta sus ventajas y el otro la suya y de esa manera interactúan.

III.1.1. Descripción General del framework

El framework[32] consta de una base de componentes, que interactúan con la aplicación a través de comandos ffmpeg, para completar los procesos de streaming de manera correcta. Algunos de estos componentes son:

- El ffserver: El framework permite la administración de streaming mediante la función servidor. Básicamente ffserver es el responsable de responder a la solicitud de transmisión de medios del cliente, y de enviar datos de transmisión de medios al abonado. A su vez, ffserver posee un archivo de pre configuración, donde se configuran los datos de los archivos de video, que van a proceder a llegar al reproductor del usuario.

- EL formato de empaquetado permite la conformación de archivos, para que cuando el cliente reciba los mismos codificados en HTML[33], le permita la decodificación del mismo en su forma más óptima.

En la actualidad HTML5 maneja los siguientes empaquetados

Primero, MP4, H264 + AAC.
Segundo, OGV, Ogg Theora + Ogg Vorbis.
En tercer lugar, WEBM, VP8.

- VLC Active X: Este es un reproductor para HTML basado en controles active X, que obtiene la reproducción bajo distintos formatos, basándose en el uso de diferentes protocolos, para el manejo de video en la red. Entre aquellos formatos compatibles para la reproducción se encuentran: MMS[34], RTSP[35] y UDP.

[32]Framework permite brindar a una aplicación predefinida funciones que ayudan al usuario a trabajar de manera ágil y completa.
[33]HTML este es el lenguaje de marcado que permite al desarrollador la elaboración de páginas web.
[34]MMS este estándar permite a los teléfonos móviles el envio y recepción de contenido multimedia.
[35]RTSP este protocolo sincroniza los datos de audio y video en tiempo real para el envío de los mismos por la red.

III.1.2. Caso Práctico de Funcionamiento

Consideremos un caso práctico que utilice el códec H.264. En este caso deberán tenerse en cuenta los siguientes aspectos:

a. El formato de empaquetado y codificación de video. Para el caso de estudio el códec a utilizar será el H.264:

El formato de empaquetado tendrá la siguiente orden:

Primero, MP4, H264 + AAC

b. El archivo de configuración con extensión .ffm servirá para la composición del envío del video. A continuación, en la Figura N° 13, se muestra el archivo .ffm con su programación.

```
Port 8090
BindAddress 0.0.0.0
MaxHTTPConnections 2000
MaxClients 1000
MaxBandwidth 30000
CustomLog -
NoDaemon
#NoDefaults

<Feed feed1.ffm>
File /tmp/feed1.ffm
FileMaxSize 1M
ACL allow 127.0.0.1
</Feed>

<Stream test1.ts>
Feed feed1.ffm
#File btv2.ts
Format mpegts
#BitExact
#DctFastint
#IdctSimple
VideoFrameRate 25
#VideoBufferSize 4096
#VideoSize 720x576
VideoSize 1920x1080
VideoBitRate 9000
NoAudio
</Stream>

<Stream stat.html>
Format status
ACL allow localhost
ACL allow 192.168.0.0 192.168.255.255
</Stream>
```

Figura N° 13: Ilustra el archivo de configuración .ffm.

c. Control VLC ActiveX: El control VLC active X debe estar iniciado (Figura Nº 14).

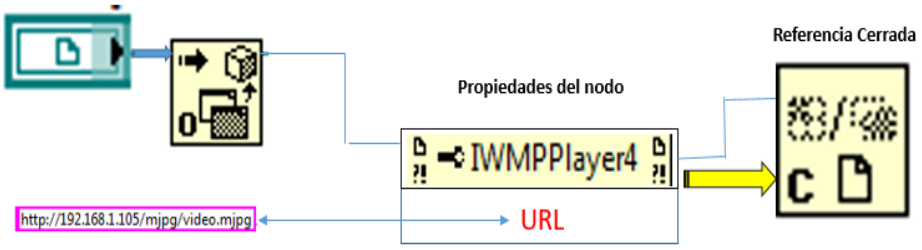


Figura N° 14: Se especifica el esquema de Control VLC Active X.
Fuente: https://wiki.videolan.org/

d. A continuación, se muestra el código de inicio y configuración de la aplicación en modo servidor, y la topología de red experimiental, de tráfico controlado, y configurada al efecto, para ejecutar ffmpeg [13]:

```
"Para iniciar el servidor:"

ffserver -f ffserver.conf

"Para iniciar el envío a través del códec H.264"

ffmpeg -i 'Stark_Trek.avi'-ab 128k -vcodec libx264 'Stark_Trek.avi'
```

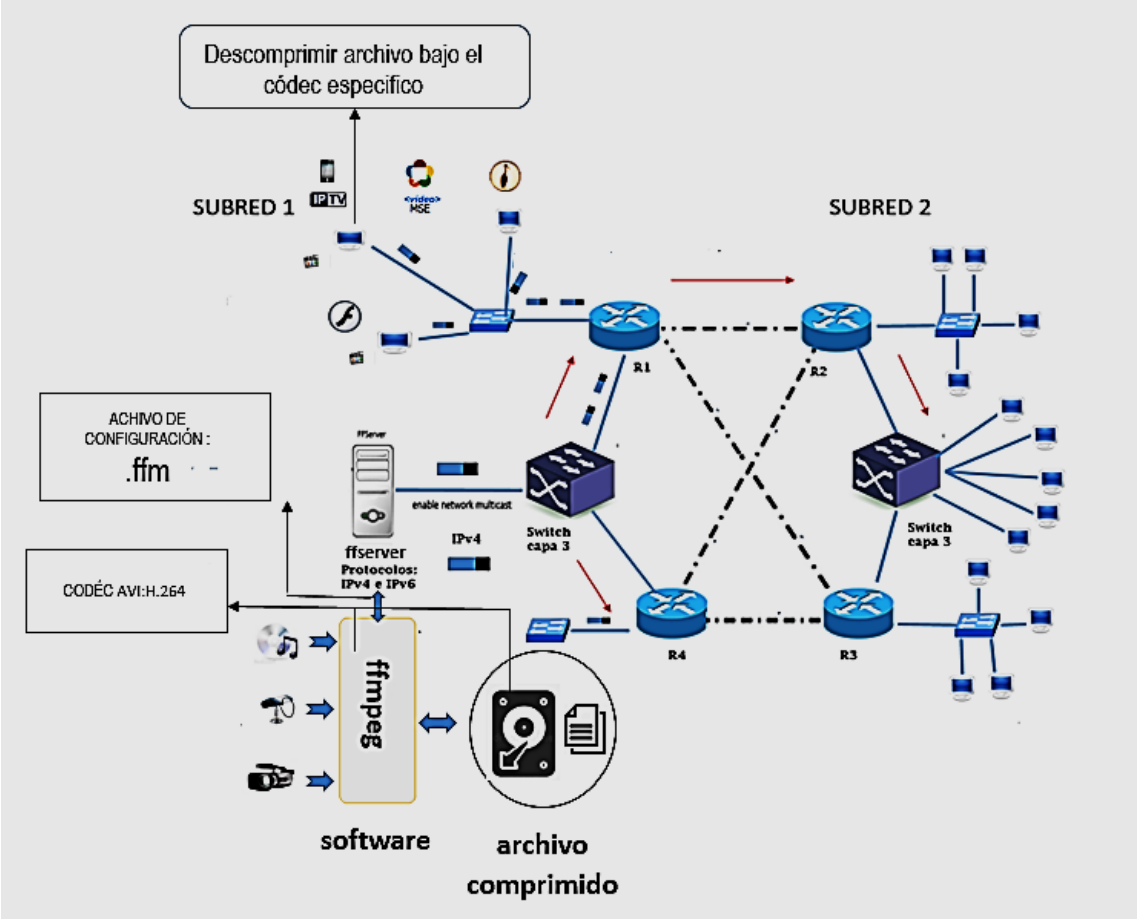


Figura N° 15: Muestra el esquema de trabajo del software ffmpeg en la red.

Se utilizó como video del estudio experimental un tráiler de Star Treck, con una duración de 2 minutos, 11 segundos. El mismo puede ubicarse en el siguiente link Youtube, ver cita Nº [14]:

III.1.2.1. Transmisión FFMPEG

La trasmisión o envío de paquetes de video es una operación que se debe realizar considerando simultáneamente muchos detalles, debido a que esta posee varios parámetros de configuración necesarios, para que las acciones del servidor de streaming se realicen correctamente cuando este en ejecución.

A continuación, se observa con mayor grado de detalle los parámetros de configuración, que se pueden aplicar para la ejecución de la transmisión de un video llamado Stark_Trek:

```
ffmpeg -i 'Stark_Trek.avi'-ab 128k -vcodec libx264 'Stark_Trek.avi'
```

El formato del archivo para la transmisión se especifica de la siguiente manera:

```
ffmpeg -re -i « filename o dirección completa » -vcodec copy -an -f
rtp rtp://127.0.0.1:11111 -acodec copy -vn -f rtp
rtp://[ff1e::e458:4d42]:8876    >   prueba.sdp   -protocol_whitelist
file,udp,rtp
```

Donde:

`-i`: input después del flag –i va a ser el objeto de origen a tratar. Se puede, entonces, especificar la dirección del objeto que se encuentra en la máquina servidora.

`-re`: Se especifica para estimar la velocidad de reproducción en una transmisión real, debido a que por defecto ffmpeg intenta leer las entradas lo más rápido posible. Sin embargo, esta opción puede ser adecuada para aumentar la velocidad de lectura de entrada, y obtener eficiencia en una transmisión en vivo.

`-vn`: Este comando posee varias funcionalidades, dependiendo de la opción a marcar. Por ejemplo, –vn (input) bloquea las transmisiones de un video, mientras, que -vn (–discard) deshabilita las trasmisiones individuales.

`-f`: Este comando fuerza el formato (de tipo de códec), para el archivo tanto de entrada, como para el archivo de salida.

`-acodec copy`: Este comando permite copiar el códec de audio al streaming de salida.

`-vcodec copy`: Este comando copia el video de salida utilizando el mismo de formato del códec que tenía a la entrada.

`-an`: Si se coloca esta opción de entrada, se bloquean todas las transmisiones de audio de un archivo, para que no se filtren o mapeen en una transmisión individual.

`rtp rtp://[ff1e::e458:4d42]:8876`: (rtp rtp:// ip : puerto): Esta opción sive para especificar a que máquina destino, y a que aplicación se va a enviar el flujo streaming.

Se puede, entonces, apreciar, dados los comandos anteriormente mencionados a través de un ejemplo, como es su utilización para el envío y la recepción de un video, para obtener un archivo con extensión .mp4:

```
ffmpeg -re -i 01.mp4 -vcodec copy -vn -f rtp rtp://224.0.0.3:8886 -
acodec copy -an -f rtp rtp://224.0.0.3:8876 > prueba.sdp -
protocol_whitelist file,udp,rtp
```

Recepción

```
ffplay -i prueba.sdp -protocol_whitelist file,udp,rtp
```

III.1.2.1.1. Transmisión Multicast FFMPEG

La trasmisión multicast debe llevarse a cabo teniendo en cuenta la dirección multicast del enlace local. Dicho de otra manera, el servidor, al generar streaming bajo una IP de tipo multicast, hace que los usuarios obtengan su proyección aplicando su URL en alguna aplicación web. De esta forma la URL estará compuesta por la dirección y el puerto (del servidor y de la aplicación ffmpeg). Entonces, cuando un host se le coloca la URL, se estará asociando a una comunicación multicast para recibir el flujo de datos.

Por ejemplo, una vez hecha la lista de configuración ffmpeg con los comandos apropiados, el servidor puede emitir tráfico mediante las siguientes líneas:

```
-  ffmpeg -re -i curso.mp4 -vcodec copy -acodec copy -f mpegts
udp://@224.0.1.2:5000
-  ffmpeg -re -i curso.mp4 -acodec copy -vcodec copy -f mpegts
udp://@224.0.1.2:5000[ttl=1,buffer_size=2097157]
```

Estas líneas generaran flujo de tipo multicast, y del otro lado de la red, el cliente deberá acceder a la trasmisión mediante una aplicación que le permita el ingreso de la URL, como por ejemplo, de la siguiente forma: `udp://@224.0.1.2:5000.`

Además, si fuera necesario el servidor puede generar streaming en el rango de las direcciones `224.0.0.0 a 224.0.0.255` para multicast de enlace local.

III.1.2.2. Detalles del archivo de configuración FFMPEG

El archivo de configuración también posee algunas caracteristicas que pueden aprovecharse, si se saben establecer los parámetros de conversión, para el caso de uso del streaming en el video. Por ejemplo, se pueden establecer los siguientes comandos por consola:

```
-vframes number (output)
```

El parámetro anterior establece los FPS (cuadros por segundos) deseados para el streaming.

```
-r[:stream_specifier] fps (input/output,per-stream)
```

La opción anterior cumple una función similar al comando anterior, solo que si se asume, ignora las marcas de tiempo y los FPS de modo constante.

```
-s[:stream_specifier] size (input/output,per-stream)
```

La opción previa establece el tamaño del video. La cual puede ser usada como alternativa, tanto para la entrada como para la salida. Si se trata como preferencia de salida, establece el tamaño del video convertido, y si es de entrada, es un acceso directo para el tamaño del video.

```
-aspect[:stream_specifier] aspect (output,per-stream)
```

El comando anterior establece el tamaño de aspecto del video de salida. (4:3, 16:9).

Si se usa con el parámetro `-vcodec copy`, modificará el tamaño del aspecto al nivel del contenedor, pero no modificará el aspecto almacenado en los frames codificados, en caso de que estos existan.

`-vn (input/output)`

Este comando precedente, también, establece el tamaño de aspecto del video de salida. (4:3, 16:9).

Al igual que el previo si se usa con`-vcodec copy`, modificará el tamaño de aspecto al nivel del contenedor. Y también, no modificará el aspecto almacenado en los frames codificados, en caso de que estos existan.

`-vcodec códec (output)`

El comando anterior permite setear el códec.

`-pass[:stream:specifier] n (output,per-stream)`

Este parámetro previo permite seleccionar el número de pasadas. Se utiliza para realizar codificaciones de videos en dos pasadas. La primera pasada se registra en un archivo de registro, y al hacer la segunda pasada, ese archivo de registro se utiliza para generar el video a la tasa de bits solicitada.

Por ejemplo:

```
ffmpeg –I foo.mov –c:v libxvid –pass 1 –an –f rawvideo –y NULL
ffmpeg –I foo.mov –c:v libxvid –pass 1 –an –f rawvideo –y /dev/null
```

El parámetro siguiente permite establecer el nombre del archivo de registro con un prefijo mediante un cálculo algorítmico.

-passlogfile [: stream_specifier]

III.2. SOFTWARE WIRESHARK DESCRIPCIÓN

Wireshark [15] fue la aplicación utilizada para la medición y captura de tráfico. Este software comprende una serie de características que sirven para analizar cada uno de los paquetes de datos, o así mismo, hacer una evaluación de conjunto a través de un snifeo, que hace este aplicativo sobre la red. Para el trabajo de tesis se evaluaron ciertos parámetros de tráfico de red, como el retardo, el jitter, la cantidad de paquetes transmitidos, etc.; y que sirvieron para poder deducir importantes aspectos sobre el comportamiento de IPTV sobre la red.

III.2.1. Ejemplo de captura de Paquetes de Datos IPTV

A continuación, se muestran la topología de red utilizada (Figura Nº 16) y capturas reales del tráfico IPTV típicas, del lado de un cliente (Figura Nº 17), y del servidor (Figura Nº 18), en la red de laboratorio controlado de la UTN Regional Mendoza.

La topología genérica utilizada es una computadora funcionando como servidor de streaming, y computadoras de escritorio (PCs) como clientes, todas conectadas en los extremos de una red mixta conformada por routers y switchs, con distintos tipos de enlaces interconectando a los mismos. En esta topología, los enlaces son del tipo FastEthernet, con una velocidad de transmisión de 100 Mbps. Para el funcionamiento entre los routers R1 a R6 se configuró el protocolo de enrutamiento OSPF v2. Para los mismos routers se configuró el protocolo de enrutamiento multicast PIM de modo denso. Los enlaces redundantes existentes se plantearon como opción para considerar una red real, pero se configuró el protocolo de ruteo, para que el tráfico entre el servidor y cada cliente siga siempre una única ruta.

- Como servidor: una computadora de escritorio, con CPU Intel Core I5, con 8 GB de RAM, y sistema operativo Linux Ubuntu.

- Como clientes: computadoras de escritorio CPU AMD Athlon(tm) II X2 250, a 3 GHz, con 4 GB de RAM, y sistema operativo Windows 10 de 64 bits,
- Los routers R1, R2, R3 y R4 fueron modelo Cisco 2811, y los routers R5 y R6 fueron resueltos con switchs multicapa Cisco WS-CS3750.
- Finalmente, para la conexión de los routers a las PCs se usaron switchs Cisco Layer 2 Catalyst Model WS-2950-24.

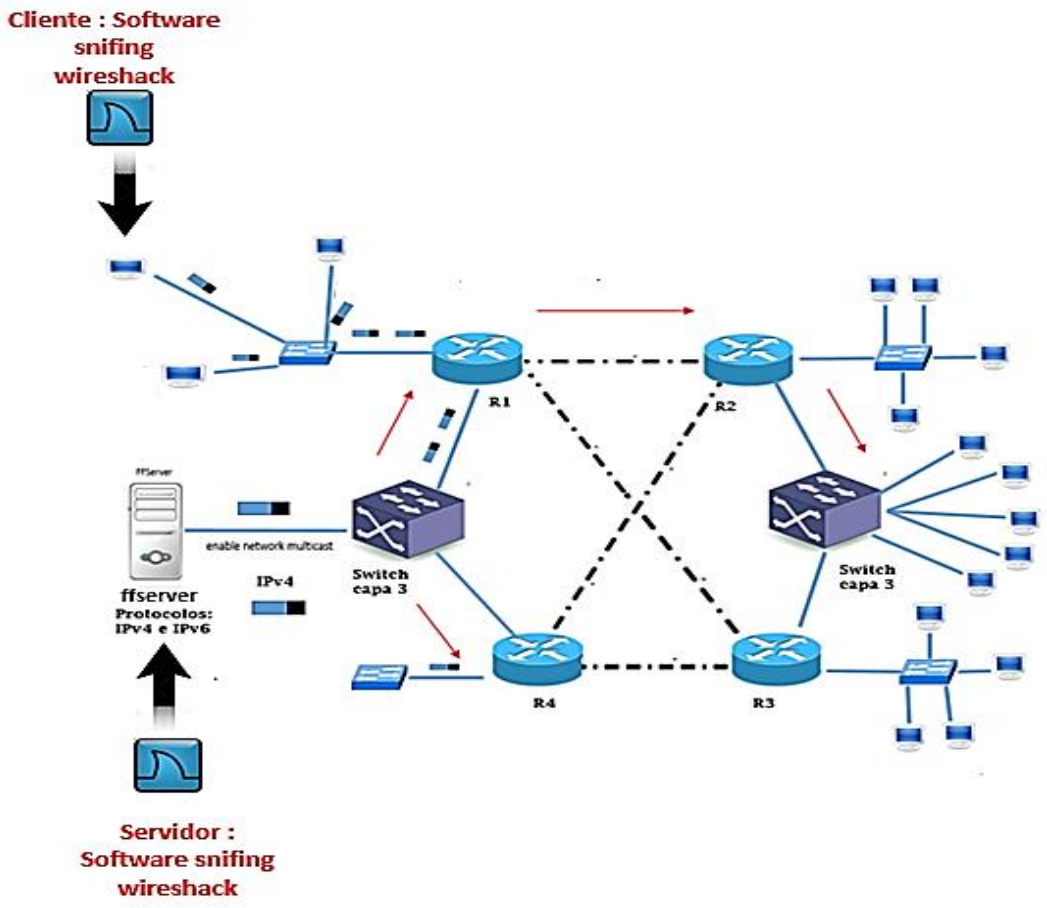


Figura N° 16: Ilustra y ejemplifica la topología experimental IPTV utilizada en la red UTN.

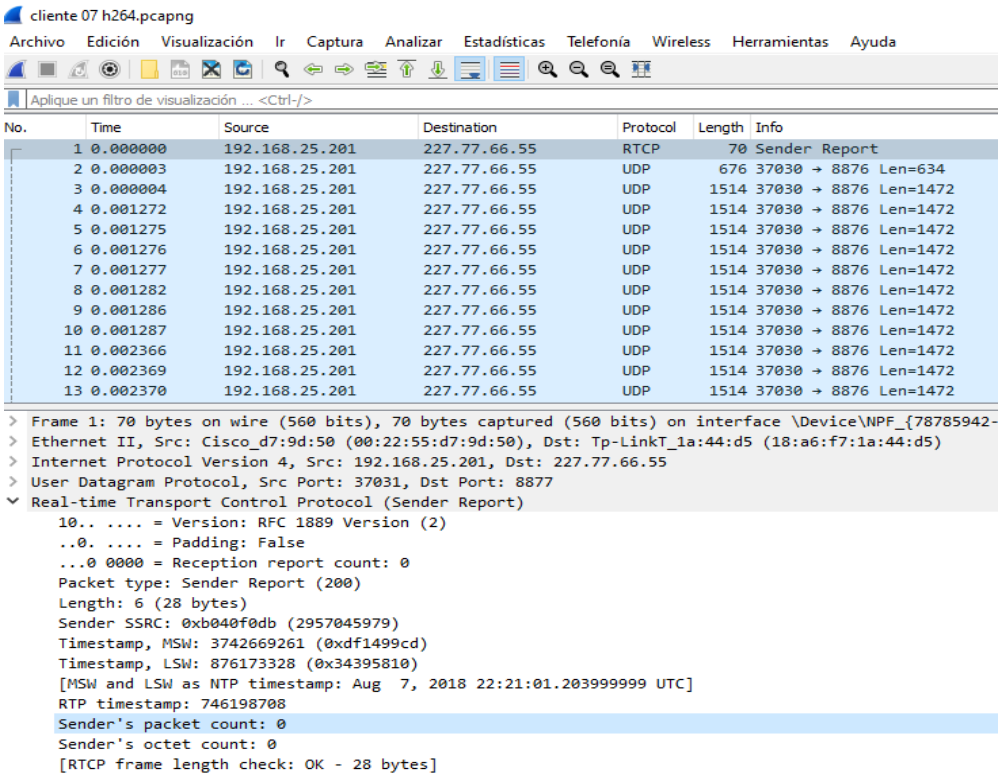


Figura N° 17: Muestra un ejemplo de captura de paquetes recibidos en un cliente.

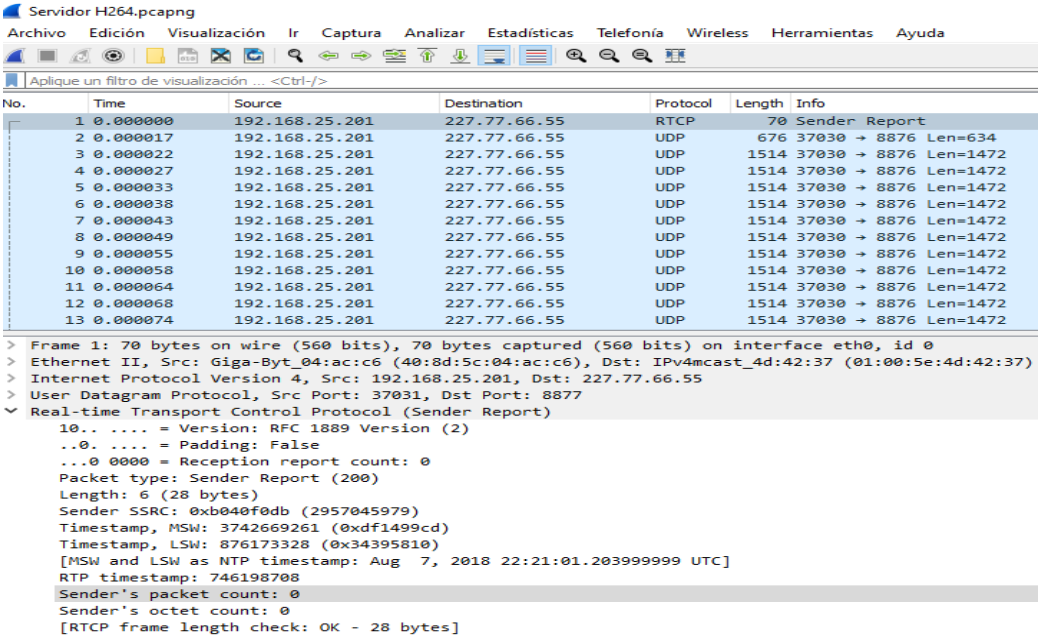


Figura N° 18: Especifica un ejemplo de captura de paquetes enviados desde el servidor.

III.3. ENTORNO DE DESARROLLO INTEGRADO

Se utilizó el IDE (Entorno de Desarrollo Integrado) de Anaconda sobre Spyder, que trabaja con lenguaje Python (Figura Nº 19). Con esta herramienta se realizaron los cálculos para obtener las métricas de red de la actividad experimental, mediante fórmulas a través de dataframes. De manera complementaria, se utilizaron librerías, como Panda y Mathplotlib, para poder establecer los dataframes y los gráficos correspondientes.

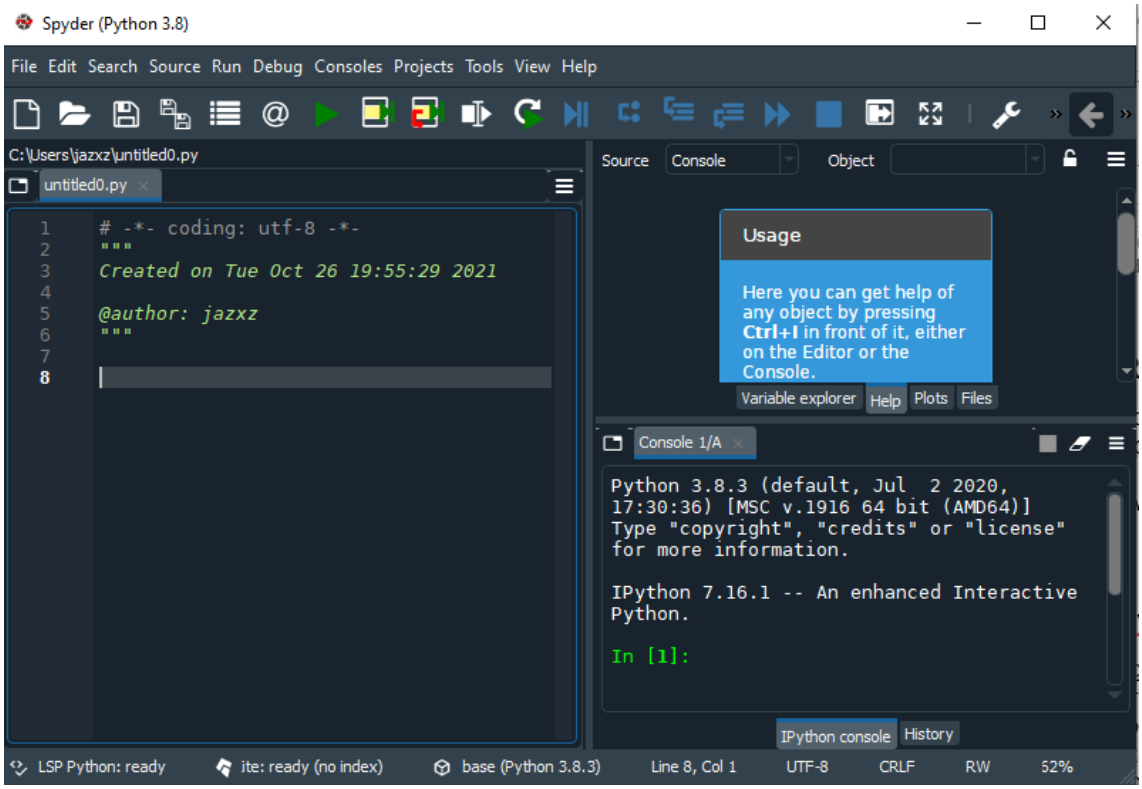


Figura N° 19: Presenta el IDE de Spyder.
Fuente: https://www.spyder-ide.org/

III.3.1. Librería Panda bajo Python

Esta librería se debe considerar para el análisis de grandes cantidades de datos o de registros. Esta librería y el lenguaje Python [16] constituyen una potente combinación para este tipo de aplicaciones. La librería es importada bajo anaconda, con el parámetro "import pandas". Se utiliza para Ciencia de Datos y Machine Learning [17]. Mientras que Python ofrece una estructura de

manejo de datos poderosa y flexible. La estructura de datos que maneja la librería panda por defecto es dataframe.

Un dataframe es una estructura de datos, en forma de array, que posee un índice para las filas, y etiquetas para las columnas. Cada celda, a su vez, es un contenedor, y se especifica por un valor de fila y un valor de columna.

Esta herramienta fue muy útil para el estudio del trabajo experimental, dado que permitió ejecutar, procesar y analizar los datos que nos ofrecen las capturas de Wireshark una vez exportadas a un archivo .csv[36].

III.3.2. Librería Mathplotlib bajo Python

La librería Mathplotlib es una librería utilizada para el uso exclusivo del lenguaje Python. Su función es específica para poder crear gráficos de tipo 2D. En el análisis de los resultados del trabajo experimental se utilizó la librería mathplotlib. Esta librería puede importarse mediante el parámetro "import mathplotlib" en Spyder. Es muy conveniente para construir gráficos rápidamente.

Mathplotlib se utilizó en conjunto con la librería seaborn, que trabaja sobre operaciones similares a la librería mathplotlib. Seaborn es una complementación de mathplotlib. La utilización de esta librería permitió la creación de gráficos mediante la llamada al archivo Excel, previamente convertido en matriz, desde Python. Sirvió para trabajar con más precisión cuando se necesitó graficar con una gran cantidad de datos.

[36]CSV pertenecen a las siglas de archivos defindos con extensión .csv y que dentro del mismo define valores delimitados por comas. Por lo general es un archivo de texto con propiedades de contenido de tablas muy similar a excel.

III.3.3. Librería openpyxl bajo Python

La librería openpyxl es otra de las librerías exclusivas de python y, al igual que las anteriores, también permite trabajar con dataframe. Pero tiene algunas diferencias con panda, ya que openpyxl trabaja con varias hojas de Excel, con un libro de Excel, entre otras cosas. El objetivo de su uso fue para darle un contexto, y agregar los gráficos a una hoja de Excel, para algún libro creado bajo el parámetro de programación load_workbook.

CAPÍTULO IV

IV.ANALISIS EXPERIMENTAL DE LAS CARACTERISTICAS DE QoS DEL TRÁFICO IPTV EN LA RED DE ESTUDIO

En el presente Capítulo se analiza el comportamiento de QoS del tráfico IPTV, en una red LAN que actúa de laboratorio experimental controlado, dentro de la red de la UTN Regional Mendoza. El análisis se efectúa determinando las métricas de prestaciones sobre diversos subescenarios propuestos al efecto, utilizando distintos códecs de video para contraste. El tráfico es inyectado con Ffmpeg sobre la topología descripta y presentada en la Figura Nº 16, y capturado con Wiresharck para su estudio.

IV.1. CALIDAD DE SERVICIO QoS

La Calidad de Servicio puede considerarse en base al comportamiento de ciertos parámetros de red, de un tráfico determinado [18], en una arquitectura dada. El objetivo es obtener, mediante mediciones, los valores de tales parámetros, que deben garantizar el cumplimiento de ciertas premisas, según el tipo de tráfico dentro de la red. Los parámetros de calidad de servicio son aquellos denominados: delay, jitter, y pérdida de paquetes. Los valores de estos tres parámetros dependerán de la arquitectura de red, y de la cantidad y tipo de tráfico que viaje por la misma, en un momento determinado, o sea de la ocupación del ancho de banda. Hay formas de establecer los valores de estos parámetros:

El delay puede ser calculado como la diferencia en tiempo, en milisegundos, de los paquetes, que salen de un host servidor a un host final medidos dentro de un flujo de datos. El jitter puede determinarse como la diferencia de fluctuaciones en tiempo entre paquetes. Por ejemplo, cuando un paquete origen fluctúa en tiempo con uno de llegada. Dicho de otra manera, cuando en una cadena de paquetes que lleva un sincronismo, hay algunos de ellos que por alguna razón pierden esa sincronización con el resto, de tal manera que se genera un retraso o adelanto en un paquete en relación a otro consecutivo que

están viajando hacia el destino. Por último, para determinar una secuencia de pérdida de paquetes en un tráfico de datos TCP[37] debe, por ejemplo, corroborase si dicha secuencia u orden de los paquetes de envío y de llegada son correctos. Supongamos en una lista de paquetes analizados, si hay dos paquetes en esa lista que tienen el mismo número de secuencia es porque hubo pérdida de paquetes en un ellos, ya que el sistema volvió a reenviar el mismo por algún motivo.

Los valores de referencia para estas métricas de red se presentan en la Tabla Nº 1:

Métrica de QoS	*Objetivo de desempeño*	*Valor de referencia*
Retardo	Límite superior del promedio de valores del retardo	100 ms
Jitter	Límite superior en el mayor cuartil menos el mínimo valor	50 ms
PLR	Límite superior de la pérdida de paquetes	0,1 %

Tabla N° 1: Tabla referencial del retardo, jitter y pérdida de paquetes establecidos por la UIT.
Fuente: https://www.itu.int/

[37]TCP protocolo que permite establecer una sesión a través de una conexión previa, tiene algunas ventajas como la elección de un ventana variable de transmisión según el tráfico, además el mismo trabajo en conjunto con el protocolo IP para proporcionar integridad y disponibilidad en el manejo de los datos en la red.

IV.2. VISIÓN GENERAL DE CÓDECS DE VIDEO

Como se afirma en el árticulo [19] que el almacenamiento o tráfico de video digital es un gran desafío debido a que el video original o sin comprimir requiere una gran cantidad de información digital. Según la definición del estándar NTSC (National Television System Committee), un video de 720x480 se puede digitalizar usando 4:2:2 YCrCb, a 30 cuadros/segundo, donde 4:2:2 es una relación de frecuencia de muestreo para una señal de video digitalizada. El primer número se refiere a la parte de luminancia de la señal, los segundos dos se refieren al croma (color). En este caso, la luminancia se muestrea 4 veces a 3.37 MHz (13.5 MHz en total), y los valores de croma dos veces a la misma velocidad. En esta disposición, la transmisión del video requeriría una tasa o velocidad de datos de más de 165 Mbps. Para archivar un video de 1 minuto de duración se necesitaría más de 1 Gbyte. En el estándar HD (High Definition), un video de la misma duración requeriría casi 5 veces más. Incluso con una resolución de video más baja, usando el estándar CIF (Common Interchange Format), con una resolución de 352x288 usando 4:2:0 a 30 cuadros/segundo, frecuentemente usado en aplicaciones de streaming de vídeo, se requeriría más de 36.5 Mbps. Estos valores exceden lo que puede transmitirse en ciertas redes, que necesitan también ancho de banda para las otras aplicaciones.

Se observa, por lo expuesto, que se necesita algún mecanismo de compresión, para transmitir o almacenar video digitalizado. La idea es representar una secuencia de imágenes, con la menor cuenta de bits posible, mientras se conserva la apariencia visual (el mismo procedimiento que se aplica en fotografía digital).

La mayoría de las veces, la compresión de video demanda compromisos entre los requerimientos de calidad de imagen, y otras necesidades vinculadas al contexto en que se ejecuta la aplicación que lo necesita. En estos casos, a título ilustrativo debería tenerse en cuenta:

- La velocidad de bits máxima expresada en bps (bits por segundo), en caso de transmisión.
- La capacidad de almacenamiento disponible, y la duración de la grabación, en caso de almacenamiento de video.
- Si la comunicación de video es bidireccional.
- La tolerancia de retardo aceptable entre extremos.

La palabra compresión se usa habitualmente como sinónimo de codificación. Por ese motivo, cuando se refiere a la operación de compresión se hace referencia a un dispositivo llamado códec (coder-decoder), que se utiliza en los ambientes multimedia. Si se desea reproducir un archivo multimedia comprimido o codificado con un códec en particular, se necesitará descomprimir o decodificar el archivo (en el destino remoto de la transmisión o al momento de recuperarlo de un almacenamiento), para poder reproducirlo o leerlo. Normalmente los códecs permiten elegir entre varios bit/rate, e incluso que se vaya adaptando, en base al nivel de compresión requerido, por los cambios en los cuadros sucesivos.

En la compresión de video puede verse un patrón en la evolución en el tiempo de los códecs de video. Desde los 1990s, cada nuevo códec es casi el doble de eficiente que el anterior. Los métodos de compresión más nuevos son mucho más complejos, y más sofisticados que los anteriores, pero demandan mucha más potencia computacional. Los primeros estándares, como MPG (Media Player Codec), se aplicaron en dispositivos de codificación y decodificación dedicados. Los siguientes estándares de video, como H.263, AVC (Advanced Video Coding), se diseñaron cuando las PCs pudieron codificar y decodificar un video. Con la introducción de HEVC (Description High Efficiency Video Coding), se produjo un cambio significativo en los códecs. Se introdujo el procesamiento paralelo en hardware, para acelerar los cálculos en la edición de software de códecs. El muy reciente codec H.266/VVC (Versatile Video Coding) llega con la idea de aportar más facilidades a la hora de consumir vídeo en streaming.

Para el desarrollo experimental se seleccionaron 4 códecs característicos, representativos y contrastables, para la evaluación del comportamiento del tráfico IPTV, todos soportados por el servidor FFMPEG:

- H.264/MPEG-4 Parte 10 AVC: Es una norma promovida por la UIT y la ISO. Ofrece un significativo avance en la eficiencia de compresión. Logra una reducción de alrededor de 2 veces en la velocidad de bits, respecto a MPEG-2 y MPEG-4 de perfil simple. Última versión: Junio 2019.
- H.265/ MPEG-H Parte2/ HEVC (High Efficiency Video Coding): Define un formato de compresión de video, sucesor de H.264/MPEG-4 AVC. Ha sido desarrollado por la ISO/IEC Moving Picture Experts Group (MPEG) y la ITU-T Video Coding Experts Group (VCEG), como ISO/IEC CD 23008-2 High Efficiency Video Coding. Puede utilizarse para ofrecer mejor calidad de videos de bajo bitrate, con la misma tasa de datos. Es compatible con la TV en ultra-alta definición y resoluciones hasta 8192x4320. En comparación con AVC, ofrece del 25% al 50% de ahorro, según contexto. Última versión: junio 2019.
- Theora: Es un formato de compresión de video desarrollado por la Fundación Xiph.Org. Se basa en el codec VP3. Theora es un códec de vídeo de propósito general con bajo consumo de CPU. Se distribuye sin costo de licencia, junto con sus otros proyectos de medios abiertos, incluido el formato de audio Vorbis y el contenedor Ogg. Google, desde 2010, ha contribuído en parte al financiamiento del proyecto de Ogg Theora Vorbis. Última versión: Junio 2011.
- Google VP8: Es un formato de compresión de video abierto y libre de regalías, anunciado en 2008 por On2 Technologies, como sucesor de VP7, y propiedad de Google desde 2010. Google liberó el códec VP8 como código abierto. VP8 permite implementaciones con una memoria relativamente pequeña. Última versión: Septiembre 2011.

IV.3. METODOLOGÍA DE ANÁLISIS DE LAS MEDICIONES OBTENIDAS EN LOS ENSAYOS EXPERIMENTALES

IV.3.1. Procesamiento de los archivos Wireshark de captura del tráfico de video IPTV

Las capturas de tráfico IPTV, tanto en el servidor como en los clientes, correspondientes al video de un tráiler Star Trek, inyectado por el servidor FFmpg, para cada uno de los 4 codecs seleccionados, se filtraron por tipo de paquetes RTCP o UDP como muestra la Figura N° 20.

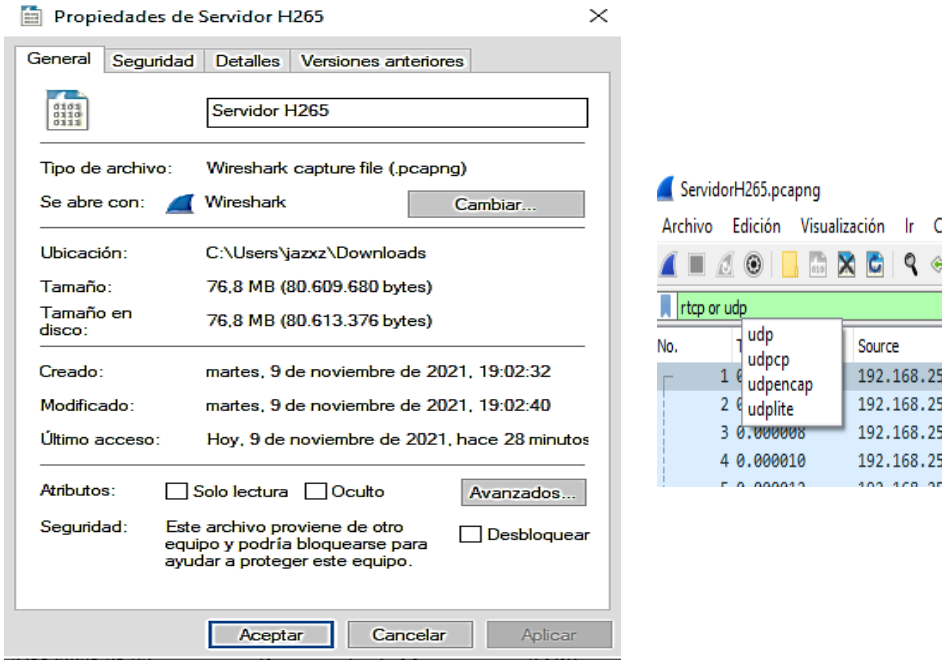


Figura N° 20: Muestra el filtrado del tráfico IPTV.

Luego, las capturas se exportaron desde archivos tipo .pcap de Wireshark a archivos tipo .csv (Figura Nº 21). Un archivo en el formato .csv es, básicamente, un archivo de texto, que permite guardar la captura como un vector de tramas, para ser analizados bajo excel.

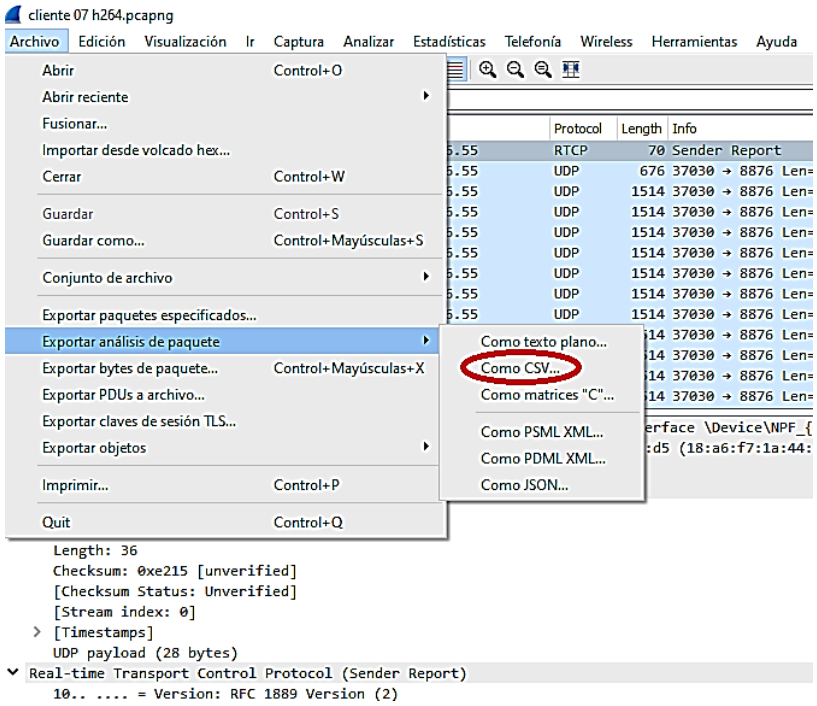


Figura N° 21: Ejemplifica la conversión de la captura Wireshark a un archivo con formato .csv.

La Tabla Nº 2 resume la cantidad de tramas, y sus longitudes mínimas y máxima, para cada uno de lo códecs bajo estudio, para la transmisión IPTV de los 2 minutos 11 segundos, del video del tráiler indicado.

Códec	
H.264	
Cantidad de Tramas	Longitud mínima/máxima de las tramas
59644	64/1518 bytes
H.265	
Cantidad de Tramas	Longitud mínima/máxima de las tramas
54202	64/1518 bytes
Theora	
Cantidad de Tramas	Longitud mínima/máxima de las tramas
46502	64/1518 bytes
VP8	
Cantidad de Tramas	Longitud mínima/máxima de las tramas
61986	64/1518 bytes

Tabla N° 2: Muestra la cantidad de tramas y las longitudes mínima y máxima para la transmisión de un tráiler de Star Trek de 2 minutos y 11 segundos.

IV.3.2. Análisis de la métrica Retardo (Delay)

Para el estudio característico del delay se obtuvo la media, el desvío estándar, el máximo y el mínimo. Con dicho objeto se desarrolló código Python específico. En el IDE, se procedió a la utilización de librerías de la sección panda, openpyxl y algunas como mathplolib para gráficos.

Se seleccionó esta metodología y recursos debido a que cada exportación de cada captura de tráfico implicaba la manipulación de una gran cantidad de tramas, del orden de los 59400 registros.

El código tiene en cuenta el comando pd.read_excel, que sirve para importar archivos desde Excel; líneas para convertir el tiempo de envío del servidor y el tiempo de recepción del cliente de nanosegundos a milisegundos; y líneas para los métodos estadísticos. Finalmente, el archivo resultante se almacenó en un archivo de salida con extensión Excel.

A continuación se ilustra el resultado para el código del codec H.264. Este código se utilizó para cada uno de los otros códecs, con el mismo objetivo. Para ver con mayor grado de precisión el proceso puede consultarse **ANEXO A**

La Figura Nº 22 presenta los resultados estadísticos obtenidos desde el código.

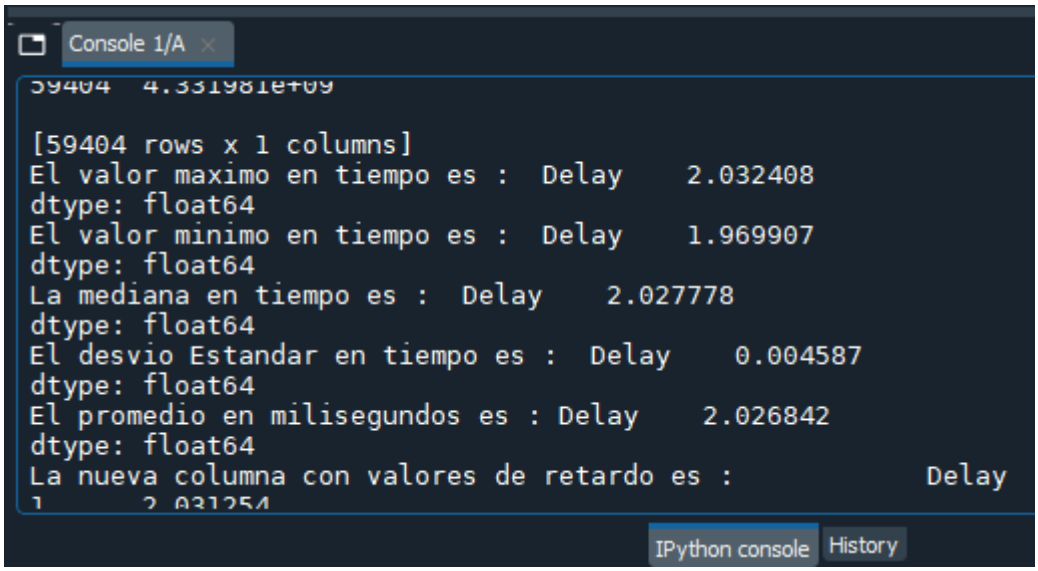


Figura N° 22: Muestra los resultados estadísticos del delay.

A continuación, se procedió al análisis mediante la librería mathplotlib para obtener un plot o gráfico de los valores obtenidos, agrupados bajo el método de Sturges en intervalos decimales o flotantes.

La Figura Nº 23 presenta las últimas líneas del código para la realización de los gráficos.

```
50  Tiempo_Servidor3 = pd.DataFrame(Tiempo_Servidor2, columns = ["TimeServidor"])
51  Tiempo_Servidor3["TimeServidor"] = pd.Series([round(val,5) for val in Tiempo_Servidor3[
52  Tiempo_Servidor3 = Tiempo_Servidor3[:-1]
53  print (Tiempo_Servidor3)
54
55  resultado = Tiempo_Cliente3["TimeCliente"] - Tiempo_Servidor3["TimeServidor"]
56  resultado_delay_H264 = pd.DataFrame (resultado, columns=["Delay"])
57
58  maximo = resultado_delay_H264.max()
59  minimo = resultado_delay_H264.min()
60
61  cantidad =  resultado_delay_H264.count()
62  cantidad_de_intervalos = (1 + 3.322 * ma.log10(cantidad))
63
64  amplitud_intervalo = (maximo - minimo) / cantidad_de_intervalos
65  primer_valor = minimo + amplitud_intervalo
66
67  print ("------------------")
68  print ("Datos para Graficar")
69
70  print ("El valor minimo es : ", minimo)
71  print ("La cantidad de intervalos es : ",cantidad_de_intervalos)
72  print ("La amplitud del intervalos es : ", amplitud_intervalo)
73  print ("El primer valor es : ", primer_valor)
74
75  libro_grafico = Workbook()
76  filesheet = "./grafico_H264.xlsx"
77  hoja3 = libro_grafico.active
78
79  grafico = pd.read_excel("resultado_H264.xlsx")
80
81  x = grafico["Intervalos"]
82  y = grafico["Repetición"]
83  plt.plot(x,y)
84  plt.xlabel("tiempo_intervalos")
85  plt.ylabel("Frecuencia_repetición")
86
87  plt.savefig("Plot2.png")
88  img = Image("Plot2.png")
89  hoja3.add_image(img, "D9")
90  plt.show()
91
92  libro_grafico.save(filesheet)
93
```

Figura N° 23: Muestra la programación sobre las últimas líneas para los cálculos del plot.

IV.3.3. Análisis de la métrica Diferencia de Retardo (Jitter)

Para el estudio característico del jitter se procedió de igual forma que para el delay (ver **ANEXO A**). Se obtuvo la media, el desvío estándar, el máximo y el mínimo.

La norma RFC 3393 considera el jitter como la fluctuación de retardo entre dos paquetes recibidos consecutivamente. La fórmula específica que el cálculo del jitter puede expresarse como:

Jitter = (tiempo rx – tiempo tx)^{i} – (tiempo rx – tiempo tx)$^{i-1}$

Para poder establecer el cálculo en Python, fue necesario primero realizar la desviación entre los tiempos de cada paquete desde el cliente rx) al servidor (tx). Una vez, que se hizo este cálculo, se debió efectuar la desviación entre los retardos de envío. Para ello se estableció la resta entre los tiempos medidos entre los intervalos de trasmisión (rx - tx) de uno de los paquetes menos el paquete anterior al mismo.

A continuación se ilustra el resultado para el codec H.264 a través de la figura Nº 24.

```
   ...: resultado_jitter.to_csv ("resultados_jitter_H264.csv")
   ...: resultado_jitter.to_excel ("resultados_jitter_H264.xlsx")
El valor maximo en tiempo es :  Time_Jitter    0.059028
dtype: float64
El valor minimo en tiempo es :  Time_Jitter   -0.061343
dtype: float64
La mediana en tiempo es :  Time_Jitter    0.0
dtype: float64
El desvio Estandar en tiempo es :  Time_Jitter    0.001739
dtype: float64
El promedio en milisegundos es : Time_Jitter    0.0
dtype: float64
La nueva columna con valores de retardo es :         Time_Jitter
0         0.002317
```

Figura N° 24: Publica los resultados por consola de los valores estadísticos.

A continuación, se procedió al análisis mediante la librería mathplotlib para obtener un plot o gráfico de los valores obtenidos, agrupados en intervalos decimales o flotantes. A continuación, en la Figura Nº 25 se presentan los resultados numéricos del código Python para la representación gráfica de la tabla de frecuencias del jitter.

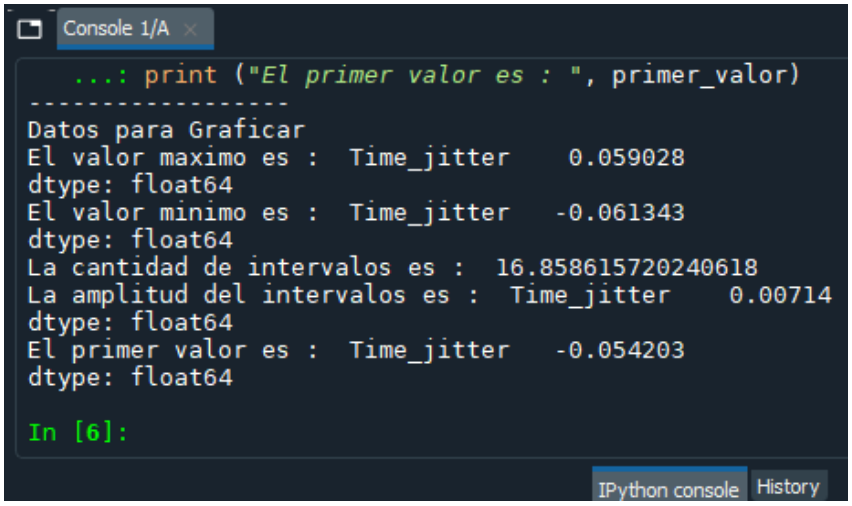


Figura N° 25: Presenta los valores de jitter en consola usados para la representación del jitter.

IV.4. ANÁLISIS DEL TRÁFICO IPTV CON EL CÓDEC H.264

IV.4.1. Análisis de la métrica Retardo

En la Tabla N° 3, se observan la cantidad de repeticiones de los retardos, con estos últimos ordenados de menor a mayor. Y en las Figuras Nº 26 y 27 se representan los retardos en abcisas y las repeticiones en ordenadas. Se destaca un pico de 30.903 repeticiones para un retardo de 2,029219 ms, y a ambos lados de este máximo, 10.315 y 13.355 repeticiones, para un delay de 2,025512 y 2,032926 mseg, respectivamente. Estos tres valores concentran casi el 91% del total de los retardos de todas las tramas (59.644).

.

	Delay (ms)	Repetición
1	1,973614	36
2	1,977321	0
3	1,981028	7
4	1,984735	23
5	1,988442	37
6	1,992149	16
7	1,995856	25
8	1,999563	61
9	2,00327	68
10	2,006977	258
11	2,010684	274
12	2,014391	586
13	2,018098	1123
14	2,021805	2317
15	2,025512	10315
16	2,029219	30903
17	2,032926	13355

Tabla N° 3: Procede a mostrar la cantidad de repeticiones por tiempo de retardo.

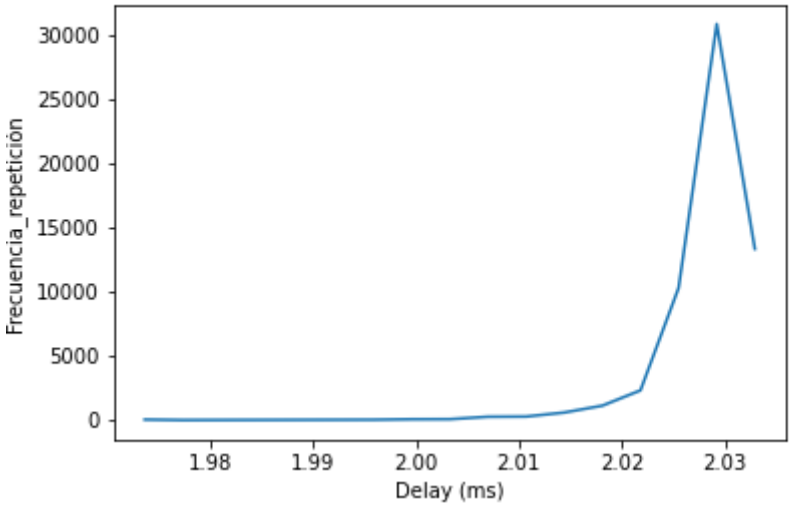


Figura N° 26: Muestra el plot que gráfica la cantidad de repeticiones por retardo.

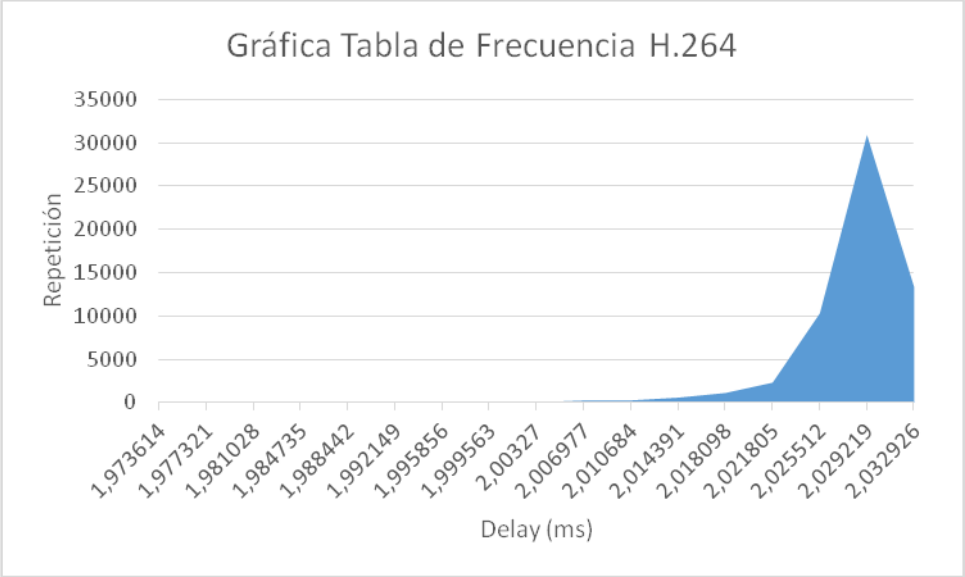


Figura N° 27: Muestra el gráfico en Excel con los resultados en frecuencia.

IV.4.2. Análisis de la métrica Jitter

En la Tabla N° 4, se presenta la cantidad de repeticiones del valor del jitter, con estos últimos ordenados de los valores negativos a los positivos. Y en las Figuras Nº 28 y 29 se observan los valores de jitter en abcisas y las repeticiones en ordenadas. Se destaca una concentración alrededor del valor 0,002917 con 57.385 repeticiones, y valores rápidamente decrecientes a ambos lados. Esta concentración implica casi el 96% del total de los jitter de todas las tramas (59.644).

Jitter (ms)	Repetición
-0,054203	1
-0,047063	1
-0,039923	4
-0,032783	3
-0,025643	3
-0,018503	13
-0,011363	48
-0,004223	493
0,002917	57385
0,010057	1254
0,017197	139
0,024337	36
0,031477	12
0,038617	3
0,045757	2
0,052897	6
0,060037	1

Tabla N° 4: Presenta la cantidad de repeticiones por valor de jitter.

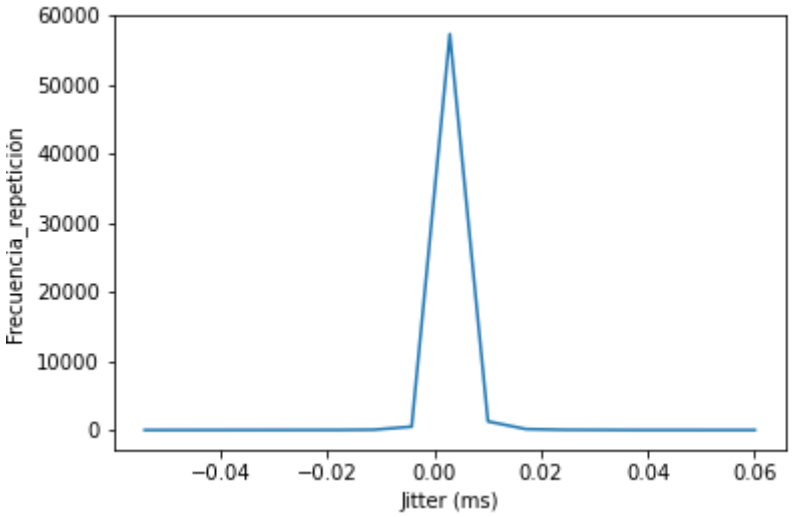


Figura N° 28: Presenta el plot que muestra la cantidad de repeticiones por jitter.

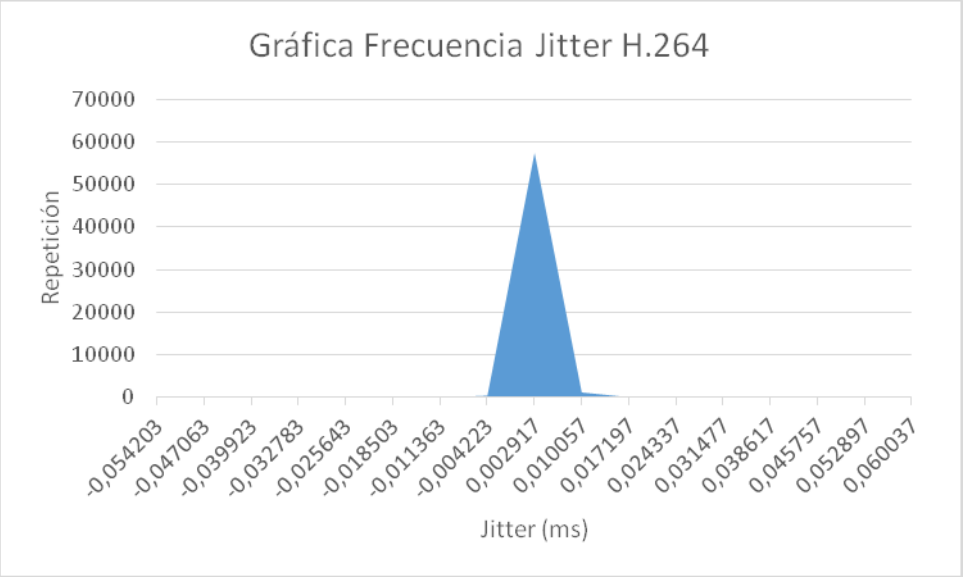


Figura N° 29: Ilustra el gráfico en Excel con los resultados en frecuencia.

Finalmente, la Tabla Nº 5 resume los valores estadísticos principales de las métricas bajo estudio, para el códec H.264:

Métricas	Resultados
Delay entre Tramas	2,026842 ms
Delay Máximo	2,032408 ms
Delay mínimo	1,969907 ms
Delay Desvío Estándar	0,004587 ms
Delay Mediana	2,027778 ms
Jitter Valor máximo	0,059028 ms
Jitter Valor mínimo	-0,061343 ms
Jitter Promedio	0 ms
Jitter con mayor número de repeticiones	0,002917 ms
Jitter Desvío Estándar	0,001739 ms
Jitter Mediana	0 ms

Tabla N° 5: Especifica los valores estadísticos para las métricas del códec H.264.

IV.5. ANÁLISIS DEL TRÁFICO IPTV CON EL CÓDEC H.265

IV.5.1. Análisis de la métrica Retardo

En la Tabla N° 6 se destaca la tendencia creciente y el valor máximo de repeticiones de los retardos, con estos últimos ordenados de menor a mayor. Y en las Figuras Nº 30 y 31 se representan los retardos en abcisas y las repeticiones en ordenadas. Se destaca un pico de 25.469 repeticiones para un retardo de 2,056046 ms, y a la izquierda este máximo, 22.298, para un retardo de 2,050162 mseg. Estos dos valores suman casi el 88% del total de los retardos de todas las tramas (54.202).

	Delay (ms)	Repetición
1	1,961902	26
2	1,967786	0
3	1,97367	0
4	1,979554	3
5	1,985438	34
6	1,991322	0
7	1,997206	30
8	2,00309	42
9	2,008974	65
10	2,014858	146
11	2,020742	229
12	2,026626	429
13	2,03251	623
14	2,038394	1158
15	2,044278	3400
16	2,050162	22298
17	2,056046	25469

Tabla N° 6: Muestra la cantidad de repeticiones por tiempo de retardo.

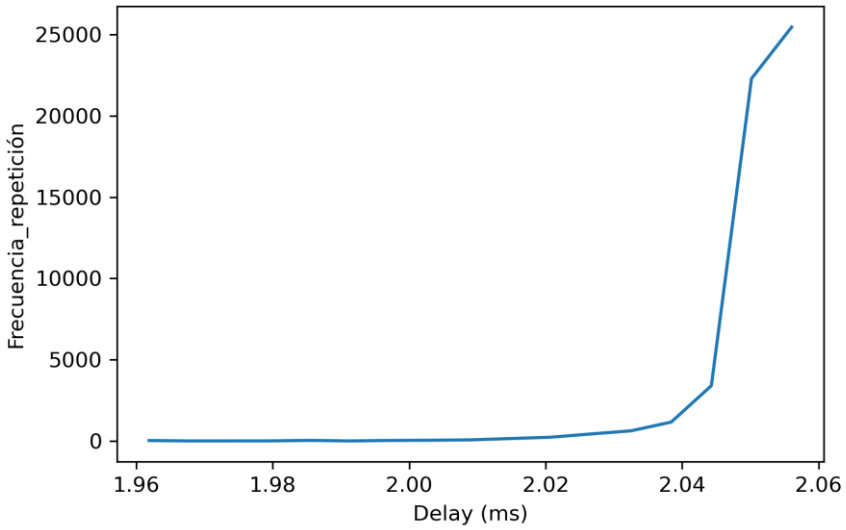


Figura N° 30: Representa el plot sobre la cantidad de repeticiones por retardo.

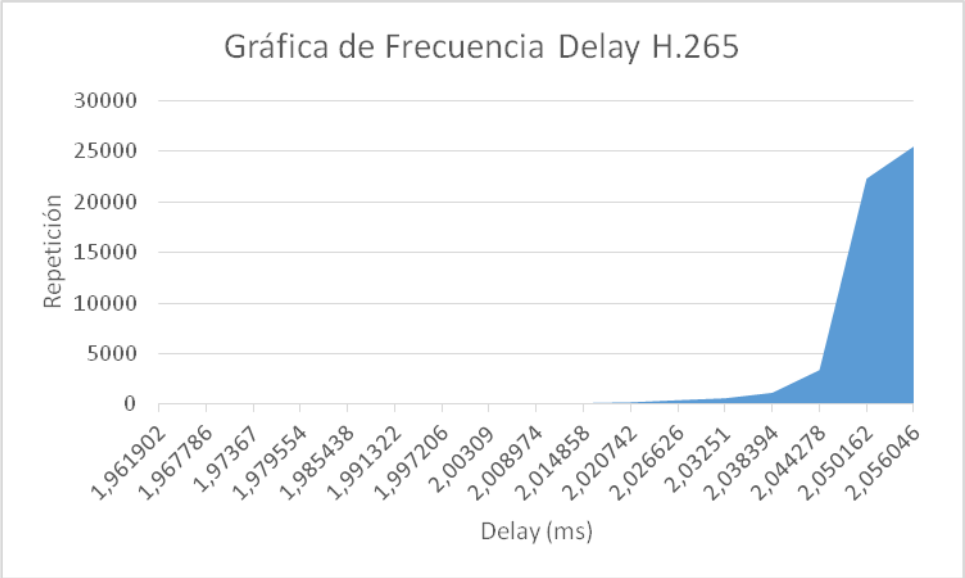


Figura N° 31: Ilustra el gráfico en Excel con los resultados en frecuencia.

IV.5.2. Análisis de la métrica Jitter

En la Tabla N° 7, se observa la cantidad de repeticiones del valor del jitter, con estos últimos ordenados de los valores negativos a los positivos. Y en las Figuras Nº 32 y 33 se observan los valores de jitter en abcisas y las repeticiones en ordenadas. Se destaca una concentración alrededor del valor 0,004601 con 52.567 repeticiones, y valores rápidamente decrecientes a ambos lados. Esta concentración implica casi el 97% del total de los jitter de todas las tramas (54.202).

Jitter (ms)	Repetición
-0,06829	2
-0,060191	1
-0,052092	0
-0,043993	1
-0,035894	3
-0,027795	6
-0,019696	15
-0,011597	55
-0,003498	458
0,004601	52567
0,0127	717
0,020799	79
0,028898	33
0,036997	6
0,045096	2
0,053195	5
0,061294	2

Tabla N° 7: Ilustra a través de excel la cantidad de repeticiones por valor de jitter.

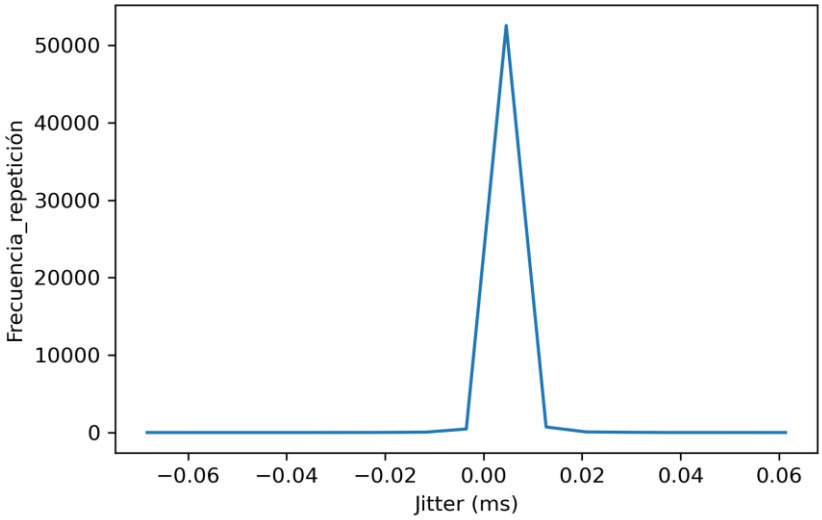


Figura N° 32: Ilustra la cantidad de repeticiones por jitter.

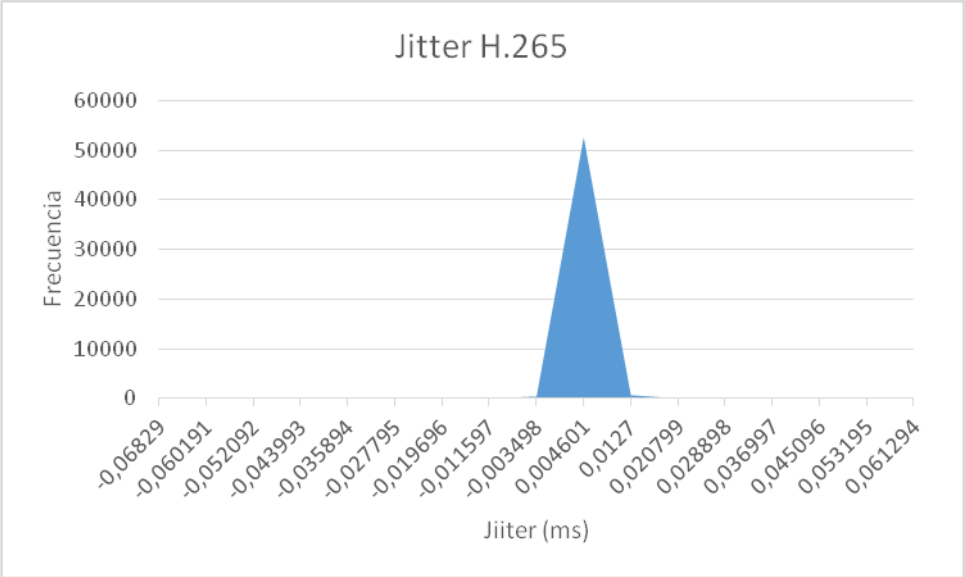


Figura N° 33: Muestra a través de graficación en Excel los resultados en frecuencia.

Finalmente, la Tabla Nº 8 resume los valores estadísticos principales de las métricas bajo estudio, para el códec H.265.

Características	Resultados
Delay entre Tramas	2,048551 ms
Delay Máximo	2,054399 ms
Delay Mediana	2,049769 ms
Delay Desvío Estándar	0,006439 ms
Delay mínimo	1,956017 ms
Jitter Valor máximo	0,059028 ms
Jitter Valor mínimo	-0,076389 ms
Jitter Promedio	0ms
Jitter con mayor número de Repeticiones	0,004601ms
Jitter Desvío Estándar	0,001891 ms
Jitter Mediana	0 ms

Tabla N° 8: Muestra los valores estadísticos para las métricas del códec H.265.

IV.6. ANÁLISIS DEL TRÁFICO IPTV CON EL CÓDEC THEORA

IV.6.1. Análisis de la métrica Retardo

En la Tabla N° 8, se observa la cantidad de repeticiones de los retardos, con estos últimos ordenados de menor a mayor. Y en las Figuras Nº 34 y 35 se representan los retardos en abcisas y las repeticiones en ordenadas. Se destaca un pico de 33.017 repeticiones para un retardo de 1,329247 ms, y a ambos lados de este máximo, 2.077 y 8.774 repeticiones, para un delay de 1,318724 y 1,339770 mseg, respectivamente. Estos tres valores concentran casi el 94% del total de los retardos de todas las tramas (46.502).

	Delay (ms)	Repetición
1	1,171402	160
2	1,181925	220
3	1,192448	8
4	1,202971	1
5	1,213494	0
6	1,224017	0
7	1,234540	0
8	1,245063	0
9	1,255586	0
10	1,266109	76
11	1,276632	82
12	1,287155	357
13	1,297678	364
14	1,308201	1128
15	1,318724	2077
16	1,329247	33017
17	1,339770	8774

Tabla N° 9: Muestra la cantidad de repeticiones por tiempo de retardo.

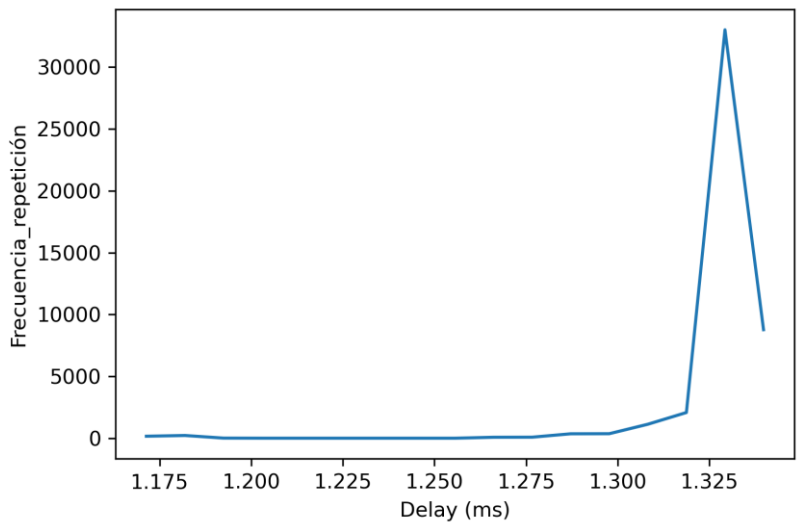


Figura N° 34: Muestra en consola la cantidad de repeticiones por retardo.

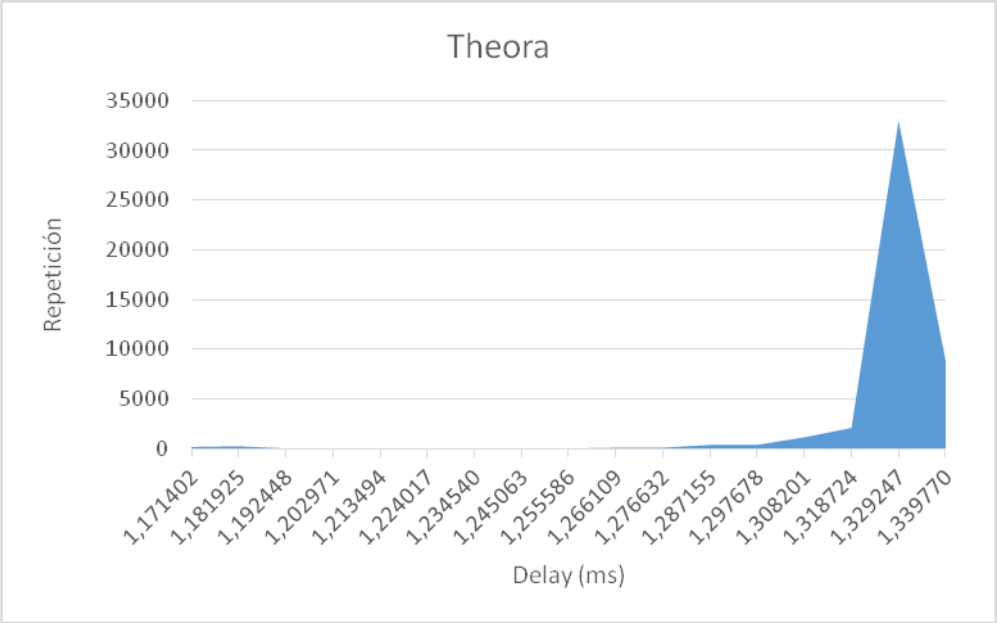


Figura N° 35. Muestra a través de graficación en Excel los resultados en frecuencia.

IV.6.2. Análisis de la métrica Jitter

En la Tabla N° 9, se presenta la cantidad de repeticiones del valor del jitter, con estos últimos ordenados desde los valores negativos a los positivos. Y en las Figuras Nº 36 y 37 se observan los valores de jitter en abcisas y las repeticiones en ordenadas. Se destaca una concentración alrededor del valor 0,00502800 con 46.056 repeticiones, y valores rápidamente decrecientes a ambos lados. Esta concentración implica casi el 99% del total de los jitter de todas las tramas (46.502).

Jitter (ms)	Repetición
-0,09760600	1
-0,08294400	0
-0,06828200	0
-0,05362000	0
-0,03895800	1
-0,02429600	3
-0,00963400	42
0,00502800	46056
0,01969000	142
0,03435200	11
0,04901400	4
0,06367600	2
0,07833800	1
0,09300000	0
0,10766200	0
0,12232400	0
0,13698600	1

Tabla N° 10: Presenta la cantidad de repeticiones por valor de jitter.

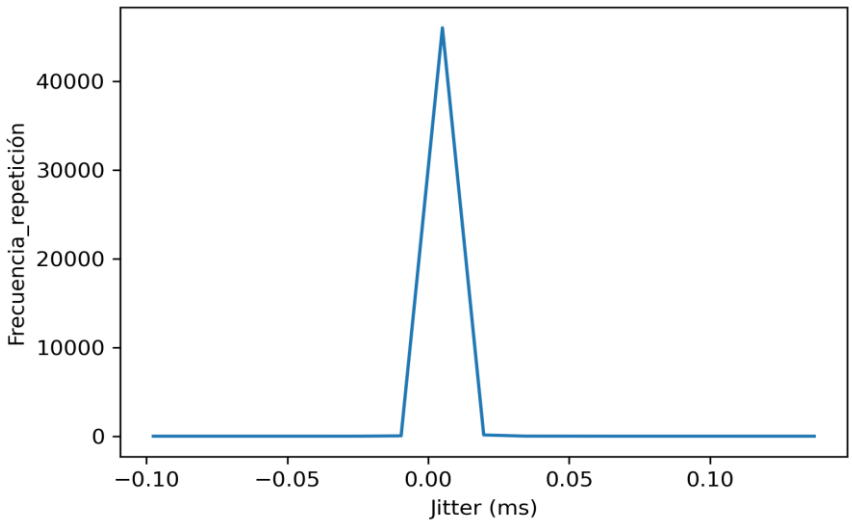


Figura N° 36: Presenta la cantidad de repeticiones por jitter.

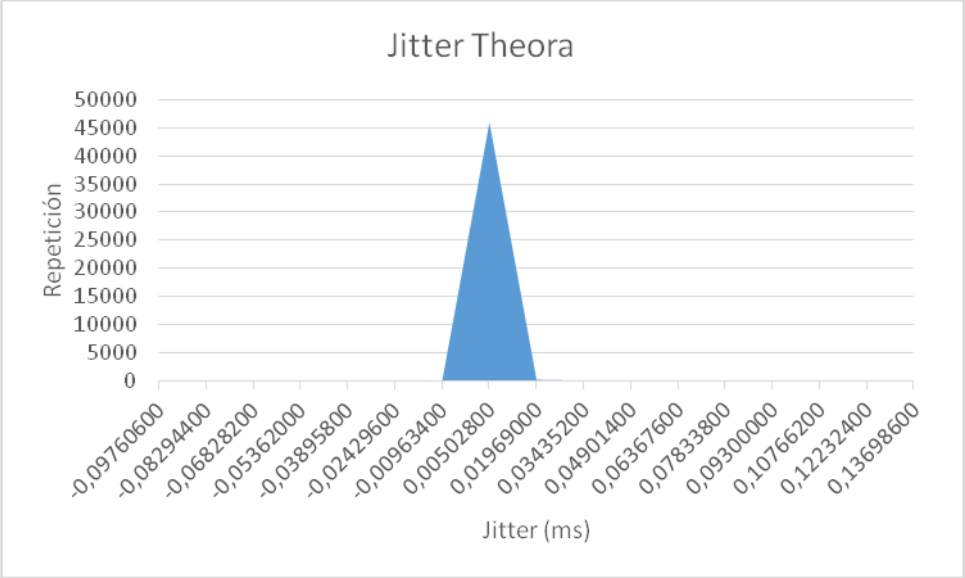


Figura N° 37: Muestra la graficación en Excel con los resultados en frecuencia.

En la Tabla Nº 11, se presentan las características de los resultados en las mediciones para el códec Theora:

Características	Resultados
Delay entre Tramas	1,32442 ms
Delay Máximo	1,334491 ms
Delay Mediana	1,327546 ms
Delay Desvío Estándar	0,01603 ms
Delay mínimo	1,160879 ms
Jitter Valor máximo	0,12963 ms
Jitter Valor mínimo	-0,112268 ms
Jitter Promedio	0 ms
Jitter con mayor número de Repeticiones	0,005028 ms
Jitter Desvío Estándar	0,001517 ms
Jitter Mediana	0 ms

Tabla N° 11: Muestra los valores estadísticos para métricas del códec Theora.

IV.7. ANÁLISIS DEL TRÁFICO IPTV CON EL CÓDEC VP8

IV.7.1. Análisis de la métrica Retardo

En la Tabla N° 12 se presenta la cantidad de repeticiones de los retardos, con estos últimos ordenados de menor a mayor. Y en las Figuras Nº 38 y 39 se representan los retardos en abcisas y las repeticiones en ordenadas. Se destaca un pico de 28.642 repeticiones para un retardo de 1,023838 ms, y a ambos lados de este máximo, 5.720 y 23.921 repeticiones, para un delay de 1,019527 y 1,028149 mseg, respectivamente. Estos tres valores concentran casi el 94% del total de los retardos de todas las tramas (61.986).

	Delay (ms)	Repetición
1	0,959173	5
2	0,963484	0
3	0,967795	5
4	0,972106	12
5	0,976417	3
6	0,980728	8
7	0,985039	13
8	0,98935	4
9	0,993661	27
10	0,997972	52
11	1,002283	108
12	1,006594	240
13	1,010905	674
14	1,015216	2311
15	1,019527	5720
16	1,023838	28642
17	1,028149	23921

Tabla N° 12: Muestra la cantidad de repeticiones por tiempo del retardo.

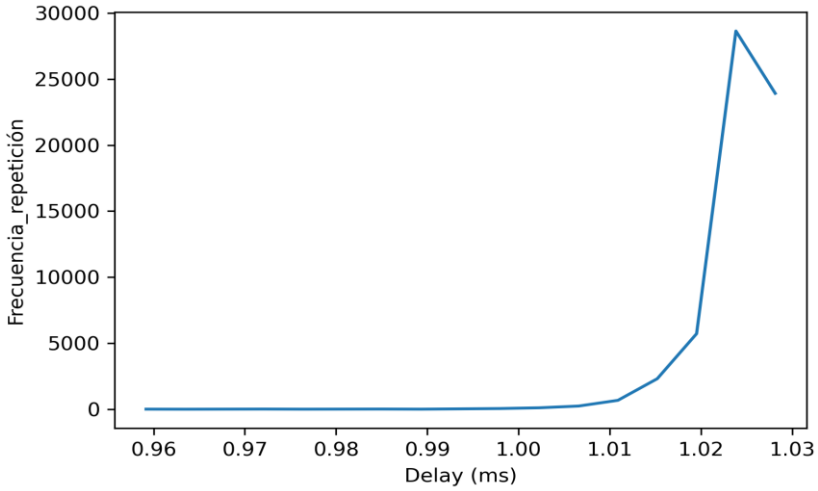


Figura N° 38: Presenta la cantidad de repeticiones por retardo.

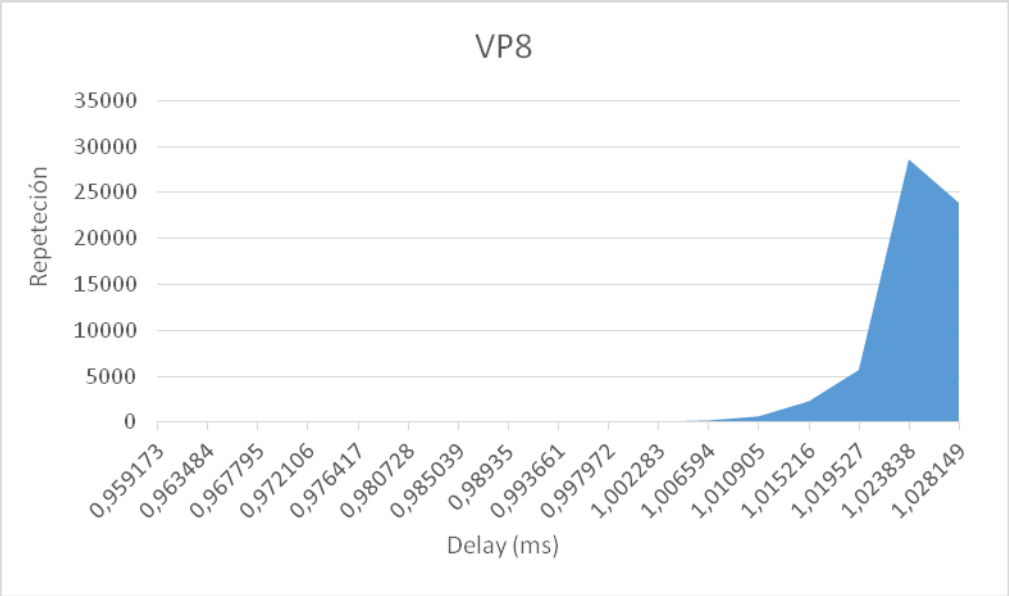


Figura N° 39: Muestra el gráfico en Excel con los resultados en frecuencia.

IV.7.2. Análisis de la métrica Jitter

En la Tabla N° 13, se presenta la cantidad de repeticiones del valor del jitter, con estos últimos ordenados de los valores negativos a los positivos. Y en las Figuras Nº 40 y 41 se observan los valores de jitter en abcisas y las repeticiones en ordenadas. Se destaca una concentración alrededor del valor 0,006515 con 55.498 repeticiones, y valores rápidamente decrecientes a ambos lados. Esta concentración implica casi el 90% del total de los jitter de todas las tramas (61.986).

Jitter (ms)	Repetición
-0,061849	1
-0,054253	1
-0,046657	2
-0,039061	1
-0,031465	1
-0,023869	2
-0,016273	15
-0,008677	113
-0,001081	5591
0,006515	55498
0,014111	454
0,021707	39
0,029303	18
0,036899	4
0,044495	0
0,052091	2
0,059687	3

Tabla N° 13: Presenta la cantidad de repeticiones por valor de jitter.

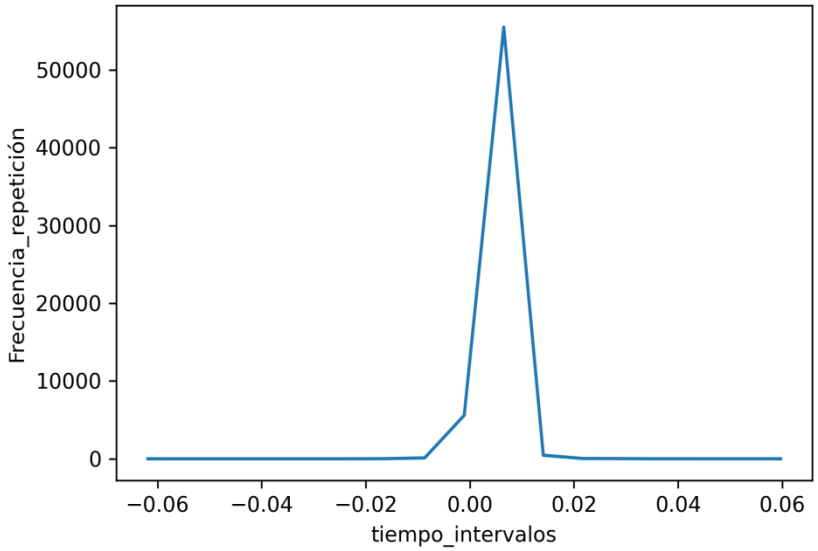


Figura N° 40: Presenta la cantidad de repeticiones por jitter.

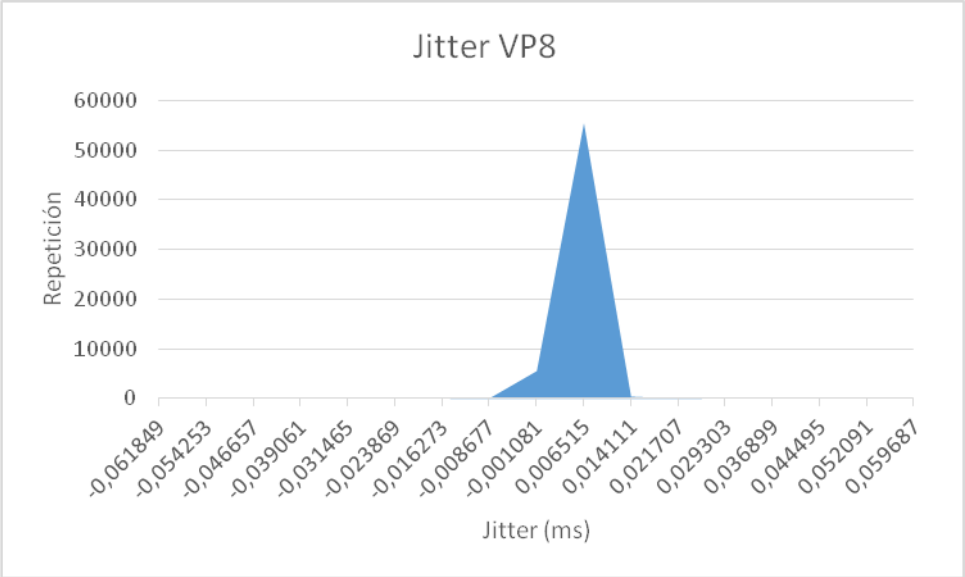


Figura N° 41: Ilustra el gráfico en Excel con los resultados en frecuencia.

Finalmente, la Tabla Nº 14 resume los valores estadísticos principales de las métricas bajo estudio, para el códec VP8.

Características	Resultados
Delay entre Tramas	1,022103 ms
Delay Máximo	1,027778 ms
Delay Mediana	1,023148 ms
Delay Desvío Estándar	0,003951 ms
Delay mínimo	0,954862 ms
Jitter Valor máximo	0,059029 ms
Jitter Valor mínimo	-0,069445 ms
Jitter Promedio	0 ms
Jitter con mayor número de Repeticiones	0,006515 ms
Jitter Desvío Estándar	0,001691 ms
Jitter Mediana	0 ms

Tabla N° 14: Muestra los valores estadísticos para las métricas del códec VP8.

IV.8. ANÁLISIS COMPARATIVO DEL COMPORTAMIENTO DE MÉTRICAS DE QoS DE LOS CÓDECS EN LA RED EXPERIMENTAL

IV.8.1. Análisis de la métrica Retardo

En la Figura Nº 42 se pueden apreciar, a los fines comparativos, la métrica de retardo para cada uno de los códecs analizados. Se observa que los valores satisfacen holgadamente los requerimientos de QoS, y que en la banda de 1 a 2 ms, los codecs presentan un comportamiento claramente diferenciado, a excepción de los codecs H.264 y H.265.5. Estos últimos codecs concentran sus retardos en el orden de los 2 mseg, mientras que el códec Theora lo hace en el orden de los 1,5 mseg. Finalmente, el códec VP8 concentra sus retardos en el orden de 1 mseg.

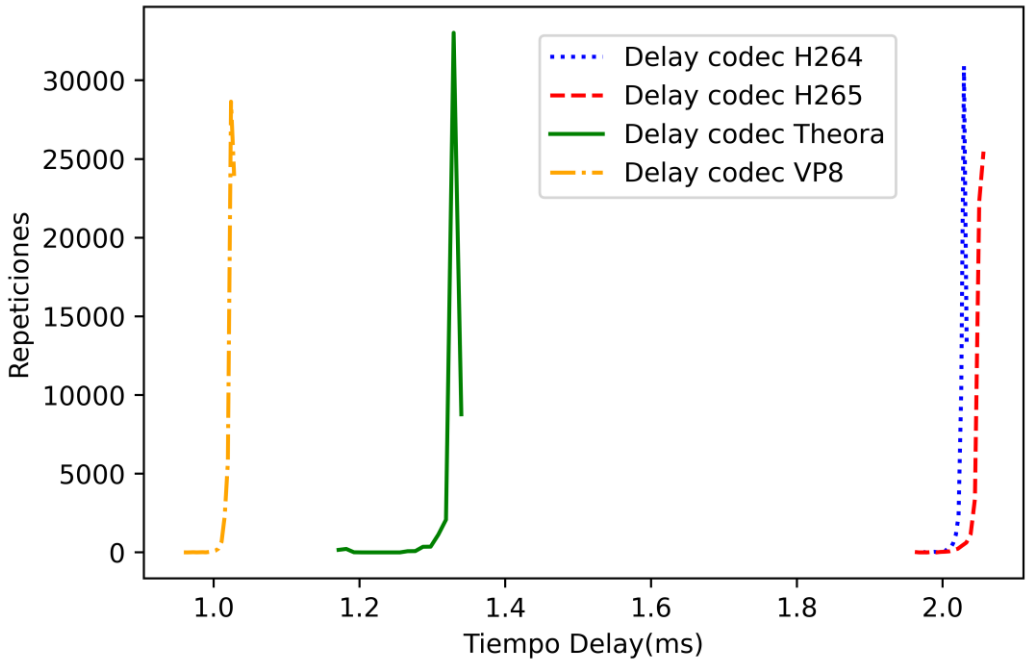


Figura N° 42: Muestra el comportamiento solapado de los codecs para la métrica retardo.

IV.8.2. Análisis de la métrica Jitter

De la misma forma, en la misma Figura Nº 43 se presenta el comportamiento de los distintos códecs bajo estudio para la métrica jitter. Se observa que los valores satisfacen holgadamente los requerimientos de QoS, y que en la banda de los -0,010 a los + 0,020 ms, los codecs presentan un comportamiento claramente diferenciado para esta métrica. Cada códec tiene una respuesta en la forma de triángulo. Sin embargo, se destaca los códec tienen bases y alturas distintas, lo que implica la amplitud de banda de valores de jitter, y el nivel de repetición con que se concentran la mayor cantidad de valores.

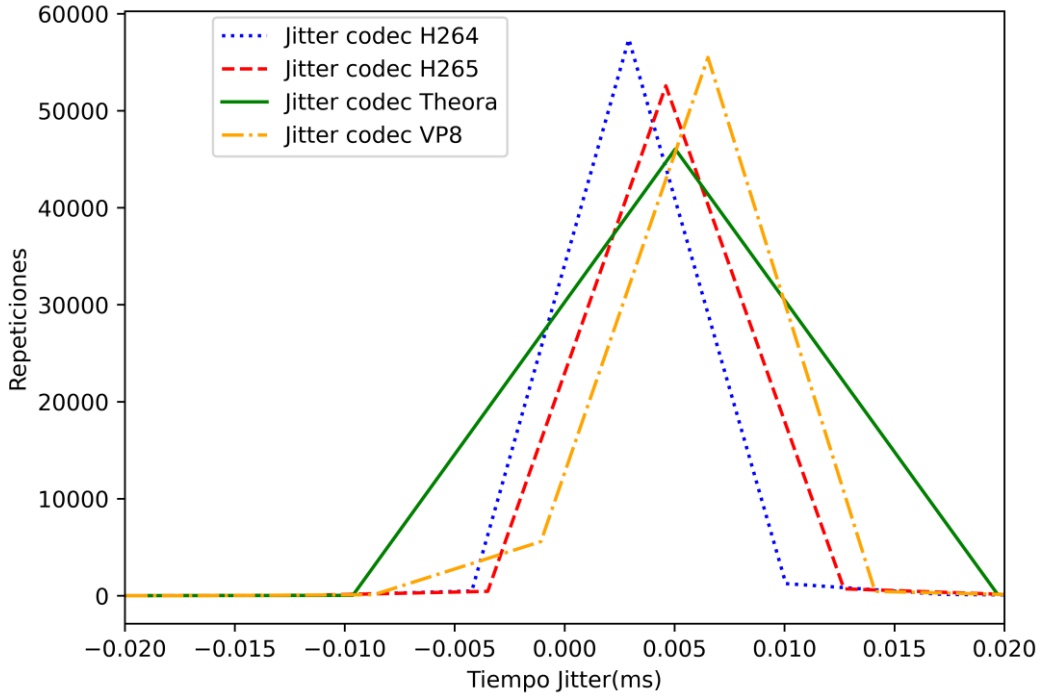


Figura N° 43. Se observa el comportamiento solapado de los codecs para la métrica jitter.

Para conocer acerca de los cálculos implementados, para realizar las gráficas, se puede consultar el **ANEXO C.**

CAPÍTULO V

V.ANALISIS EXPERIMENTAL DE LAS CARACTERISTICAS DE QoE DEL TRÁFICO IPTV EN LA RED DE ESTUDIO

En este Capítulo se evalúa el comportamiento de QoE del tráfico IPTV, en una red LAN que actúa de laboratorio experimental controlado, dentro de la red de la UTN Regional Mendoza. El análisis se efectuó sobre determinandos valores de QoE en base a mapeos QoE/QoS, es decir, desde las métricas de QoS. Para ello, se utilizaron los resultados de las métricas obtenidas de los distintos códecs de video, para contraste, tal como fue descripto en el Capítulo anterior.

V.1. CALIDAD DE EXPERIENCIA QoE

Los procesos de QoS, por si solos, no son totalmente adecuados para proporcionar una garantía de rendimiento, debido a que no tienen en cuenta la percepción del usuario sobre el rendimiento de la red y la calidad del servicio. Esto condujo a la disciplina emergente de Calidad de Experiencia (QoE, Quality of Experience).

Puede visualizarse en base a los procesos de QoS que existen otras perspectivas de estudios de QoS/QoE para otros autores como se menciona en los artículos [20] [21].

Asi mismo, la proliferación de diferentes tipos de dispositivos de acceso destaca aún más la importancia de los frameworks de QoE. Como una ilustración, la QoE para un usuario, que mira un clip de noticias en una PDA, probablemente, diferirá de otro usuario, que vea ese mismo clip de noticias en un teléfono móvil 4G. Esto es debido a que las dos terminales tienen distintas pantallas de visualización, distintas capacidades de ancho de banda, de velocidad de paquetes, códecs y potencia de procesamiento. Por lo tanto, la entrega de contenidos o servicios multimedia a estos dos tipos de terminales, sin considerar cuidadosamente las expectativas o requerimientos de calidad de los usuarios, para estos tipos de dispositivos, podría llevar a un exceso de provisión de servicios y desperdicio de recursos de red.

Informalmente, la QoE se refiere a la percepción del usuario de un servicio particular. La QoE necesita ser una de las métricas centrales empleadas durante el diseño y la administración de las redes, de los sistemas de entrega de contenidos y de otros procesos de ingeniería. Esto es porque se refiere a una medición del rendimiento extremo a extremo, en el nivel de servicio, desde la perspectiva de los usuarios, medido en los dispositivos del usuario final.

La calidad de la experiencia (QoE), desde una visión teórica, es el grado de satisfacción o molestia del usuario de una aplicación o servicio. Resulta del cumplimiento de sus expectativas, con respecto a la utilidad/disfrute de la aplicación o servicio, a la luz de la personalidad y el estado actual del usuario.

En la práctica, los hallazgos clave de los proyectos relacionados con QoE muestran qué, para muchos servicios, los múltiples parámetros de QoS contribuyen a la percepción general de la calidad de los usuarios. Esto ha dado lugar a la aparición del concepto del enfoque en capas QoE/QoS, en el que los requisitos de los usuarios dirigen las estrategias de dimensionamiento de la red.

El enfoque en capas de QoE/QoS no ignora el aspecto de QoS de la red, por el contrario, se complementa con las perspectivas a nivel del usuario y del servicio, como se muestra en la Figura Nº 44.

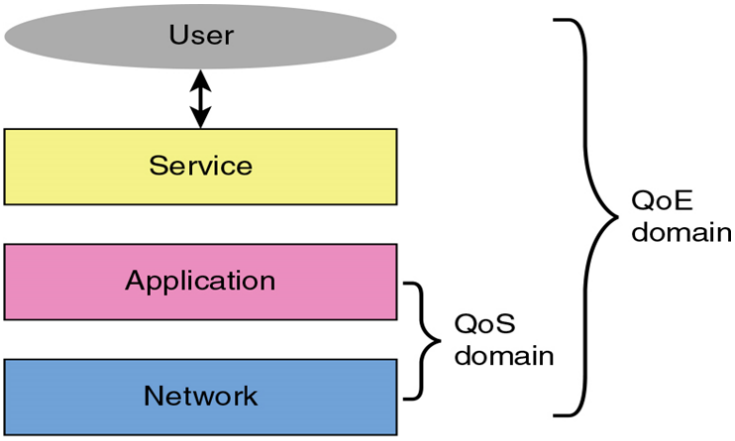


Figura Nº 44: Ilustra la superposición entre los dominios de QoE y QoS.
https://www.itu.int/

Debe tenerse en cuenta que debido a que existe una superposición entre los dominios de QoE y QoS, hay una cantidad considerable de intercambio de información/retroalimentación entre los frameworks.

Los niveles en el enfoque en capas son los siguientes:

- *Usuario*: El usuario interactúa con el servicio. Es su grado de satisfacción o molestia con el uso del servicio que se va a medir. Al estar vinculado a la percepción humana, la QoE es difícil de describir de forma cuantitativa y varía de persona a persona. Las complejidades de QoE, a nivel de usuario, derivan de las diferencias entre las características de los usuarios individuales, algunas de las cuales podrían variar en el tiempo, mientras que otras son de naturaleza relativamente estable. Los ejemplos pueden incluir género, edad, actitudes, experiencia previa, expectativas, estatus socioeconómico, antecedentes culturales, nivel educativo, etc. Por lo tanto, se convierte en un desafío derivar métricas de QoE unificadas para todos los usuarios y sus contextos. La práctica actual, en cualquier medición de QoE, es identificar y controlar las características relativamente estables de un usuario, de una manera que sea satisfactoria para al menos una gran proporción del posible grupo de usuarios.
- *Servicio*: El nivel de servicio proporciona un nivel virtual donde se puede medir la experiencia del usuario del rendimiento general del servicio. Es la interfaz donde el usuario interactúa con el servicio (por ejemplo, la pantalla visual para el usuario). También es donde se miden los umbrales de tolerancia. Como una ilustración, las medidas de QoE, desde la perspectiva del usuario, para las aplicaciones de transmisión, podrían ser el tiempo de inicio, la calidad audio/visual, el retardo de cambio de canal y las interrupciones del búffer. Sin embargo, las medidas de QoE para las aplicaciones de navegación web podrían ser los tiempos de espera de carga de la página.
- *QoS a nivel de aplicación (AQoS, Application-level QoS)*: La AQoS se ocupa del control de los parámetros específicos de la aplicación, como la

resolución de contenidos, la velocidad de bits, la velocidad de cuadros, la profundidad de color, el tipo de codec, la estrategia de capas y la velocidad de muestreo. La capacidad de la red, a menudo, determina el ancho de banda que se asignará a un servicio para la transmisión. Debido a este recurso fijo subyacente, algunos parámetros, al nivel de aplicación, normalmente se ajustan y controlan para lograr el nivel de calidad deseado. Por ejemplo, para un servicio de audio, una frecuencia de muestreo de 96 kHz podría permitir que se percibiera audiblemente más información, en comparación con una frecuencia de 48 kHz. Pero esta mayor tasa de muestreo conlleva el gasto de generar tamaños de archivo de audio más grandes. Esto se debe a que la frecuencia de muestreo es la cantidad de veces por segundo que se mide una señal de sonido analógico. Cada una de estas mediciones (o muestras) se almacena o transmite como un valor digital.

Como otro ejemplo, para los servicios de video, existe una gran variedad de tamaños de pantalla de dispositivos (cada uno con diferentes relaciones de aspecto) entre los que elegir. La única característica común en este conjunto de equipos es que todos son capaces de volver a reescalar imágenes de video. Para una tasa de bits determinada, puede haber una compensación entre las imágenes de resolución más baja, que son ligeramente borrosas, y con menos artefactos digitales (anomalías visuales), en comparación con las imágenes de resolución más alta, que proporcionan imágenes más nítidas, pero posiblemente con más artefactos. La tasa de bits, por lo general, proporciona una indicación de la calidad de un archivo de video (o audio). Esto se debe a que representa el número de bits utilizados en la codificación de cada segundo de un archivo. La mayoría de los estándares de compresión utilizan esquemas de codificación de compensación basados en bloque y movimiento, como resultado, se agregan artefactos de compresión adicionales al video decodificado.

- *QoS a nivel de red (NQoS, Network-level QoS)*: Este nivel se ocupa de los parámetros de red de bajo nivel, como la cobertura del servicio, el ancho

de banda, el retardo, el rendimiento y la pérdida de paquetes. Hay varias formas en que los parámetros de QoS, a nivel de red, impactan en la QoE. Una de esas formas es a través de la demora de la red, lo que afecta la QoE, especialmente en los servicios interactivos. Por ejemplo, la naturaleza interactiva de la navegación web, que requiere múltiples eventos de recuperación, dentro de una determinada ventana de tiempo, podría verse afectada por las variaciones de retardo de la red. Los servicios de voz sobre IP (VoIP) podrían tener estrictos requisitos de tiempo de respuesta, mientras que los servicios de correo electrónico podrían tolerar demoras mucho más largas.

Los diferentes métodos de distribución de transmisión de video a través de la red también afectan la QoE de diferentes maneras. Por ejemplo, la distribución adaptativa basada en HTTP, que utiliza TCP, reacciona a las restricciones de ancho de banda y la capacidad de la CPU de una de las siguientes maneras:

- Cambiando la transmisión por el uso de otras codificaciones de velocidad de bits disponibles (dependiendo de los recursos disponibles).
- El congelamiento de frame (rebuffering) que se produce debido a la inanición de entrada de paquetes en el búffer del player.

La conmutación en la velocidad de bits y el rebuffering tienen un efecto adverso en la QoE.

La discusión anterior sugiere que el efecto de QoE podría atribuirse únicamente a la capa de aplicación, o a una combinación de las capas de red y de aplicación. Aunque los intercambios entre la calidad y la capacidad de la red pueden comenzar con la QoS a nivel de la capa de aplicación, debido a consideraciones de capacidad de la red, una comprensión de los requisitos del usuario a nivel de servicio (es decir, en términos de medidas de QoE) permitiría una mejor elección de los parámetros de QoS a nivel de la aplicación, que se asignarán a los parámetros de QoS a nivel de red. Existen escenarios de trabajo que apuntan a controlar la QoE utilizando parámetros de QoS como actuadores.

V.2 MODELOS DE MAPEO DE QoS/QoE en IPTV

El crecimiento acelerado que ha presentado el servicio de IPTV (Televisión sobre Protocolo IP), sobre Internet, ha obligado a los proveedores de servicio a implementar esquemas, para monitorear la calidad de experiencia sobre el servicio ofrecido. Esta calidad de la experiencia, la Unión Internacional de Telecomunicaciones en su Recomendación P.10/100 la define como el grado de satisfacción o inconformidad del usuario por una aplicación o servicio".

Es así que para evaluar o medir la calidad de la experiencia percibida por el usuario se han propuesto tres métodos: (i) métodos subjetivos, (ii) métodos objetivos y (iii) métodos indirectos.

Los métodos subjetivos están relacionados con la utilización de personas, para evaluar la calidad del video en un ambiente controlado, mediante el uso de encuestas. Este método es costoso, porque demanda tiempo y una logística adecuada para la realización de dichas pruebas.

Los métodos objetivos son algoritmos, que utilizan una señal de referencia completa, parcial o sin utilizar señal de referencia, para medir calidad del video. Muchos de estos algoritmos ya están implementados comercialmente, pero requieren alto procesamiento, o utilizan pocas variables de análisis, para realizar la medición de calidad del video.

Por último, están los métodos indirectos, que mediante un modelo matemático evalúan la calidad de experiencia asociada al video. Este modelo matemático es generado mediante la variación de métricas de calidad de servicio, y la utilización de un método subjetivo u objetivo, para evaluar la calidad de experiencia asociada al video. Con este conjunto de datos se realiza un procedimiento matemático, que permite obtener el modelo. A su vez, este tipo de enfoque permite realizar mediciones en vivo, y computacionalmente no requieren un procesamiento elevado, como los métodos objetivos.

Otros autores han propuesto modelos [22], basándose en una variante de los métodos indirectos, esto con el fin de no depender de pruebas subjetivas, ni de

la utilización de métodos objetivos, para evaluar la calidad de experiencia asociada al video. Los modelos propuestos utilizan los tres parámetros de calidad de servicio (retardo, variación del retardo y pérdida de paquetes).

V.3. METODOLOGÍA DE ANÁLISIS DE LAS QoE

V.3.1. Introducción

La naturaleza altamente inelástica del contenido de IPTV exige que se definan límites estrictos de alto nivel, para QoS y QoE de los usuarios finales. Desde la perspectiva del rendimiento de la red, las métricas de QoS se utilizan tradicionalmente para evaluar la distribución de (entre otros) de contenido IPTV:

- *Pérdida de paquetes*: Las redes IP de mejor esfuerzo de todo tipo son susceptible a la pérdida de paquetes. Debido a que el contenido de IPTV está encapsulado en datagramas UDP, no hay ningún mecanismo que maneje los paquetes perdidos/caídos. La pérdida de paquetes puede deteriorar enormemente la calidad del contenido multimedia. Para ocultar los errores en el flujo del video, los paquetes perdidos son reemplazados por marcos previamente decodificados, que pueden causar pixelización, cuadros bloqueados y tartamudeo en la salida del video, y con problemas equivalentes a la transmisión de audio.
- *Retardo del paquete*: A medida que los paquetes viajan desde el origen hasta el destino hay un retardo de propagación. Si esta latencia alcanza valores altos, puede ocurrir el bloqueo de imágenes e imágenes corruptas.
- *Jitte*r: Si el retardo de propagación no es de tiempo constante, la latencia a través de una red varía. La métrica que mide esta variabilidad es el jitter. Los decodificadores IPTV y Set-Top-Boxes (STB) requieren un flujo de IP constante y las fluctuaciones puede causar tanto el desbordamiento, como el agotamiento del búffer, lo que resulta en pixelización y congelación de fotogramas del contenido visualizado.

V.3.2. Determinación de la QoE

Como se expresó anteriormente, algunos modelos [23], a los efectos de eliminar, inicialmente, la subjetividad del usuario y dar una opinión preliminar de la situación de la red, expresan la QoE basándose en las métricas de QoS. Por ello, el primer problema que se debe resolver es cómo diferenciar la QoE para cada red, incluyendo las Redes de Datos LAN, en función de algunos parámetros medibles.

Lógicamente que pueden encontrarse, intuitivamente, algunas relaciones entre la QoE de la red, y los valores observados de delay y jitter en las Redes de Datos. Cuando los valores de estas métricas son altos, el parámetro QoE de la red debe tener un valor bajo. Por otro lado, aunque la pérdida de paquetes es muy mala para la QoE, podría ser cero, por lo que no se espera que multiplique directamente al dividendo de cualquier expresión matemática que modele y caracterice el comportamiento. Además, los valores más altos de pérdida de paquetes afectan más al valor de QoE

Teniendo en cuenta lo mencionado, se puede considerar que la QoE de la red podría aproximarse con la expresión dada en [24]:

$$QoE = \frac{1}{(Delay + K * Jitter) * e^{\wedge (packet\ loss)}}$$

Donde K permite balancear el impacto del parámetro jitter, con respecto al retardo, para el cáculo de la QoE de los usuarios. Ninguno de los parámetros utilizados en la expresión podría ser negativo. En la Figura Nº 45 se muestra, en un ambiente experimental, el valor de QoE como una función de retardo de la red, para varios jitter y un K=2, y una pérdida de paquetes de 0,01. Se prefieren los valores más altos posibles para la QoE de la red, y tendiendo a 1, en la medida que, idealmente, el retardo, el jitter y la pérdida de paquetes se aproximan a 0.

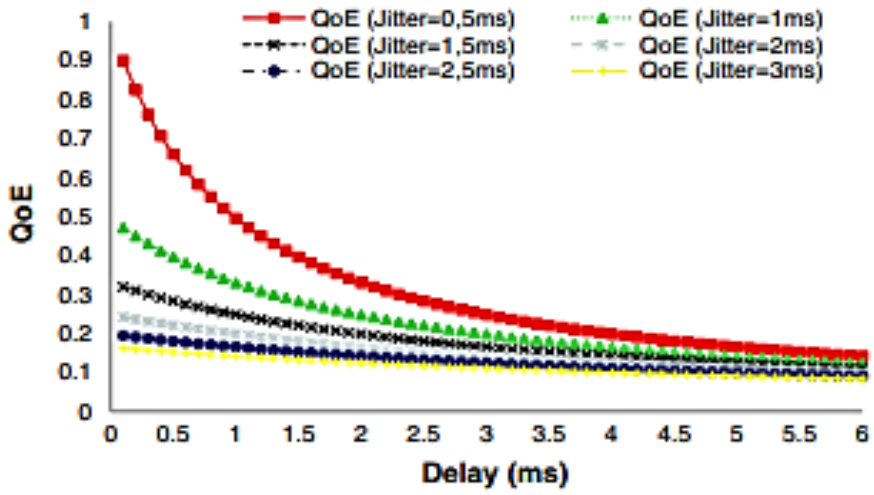


Figura Nº 45: Presenta los valores de QoE en función de las métricas de QoS de la red.
Fuente : https://citeseerx.ist.psu.edu/ [25]

V.3.3. Aplicación en los escenarios de experimentación

Para el estudio de la QoE se utilizó la fórmula anterior, aunque considerando un valor apropiado de K, para la red LAN experimental, y usando los valores de las métricas de retardo y jitter que se presentaron en el Capítulo anterior. La métrica pérdida de paquetes no se utiliza en la fórmula, debido a que la red LAN de laboratorio, que permitía un análisis de tráfico controlado, no presentó pérdidas.

$$QoE = \frac{1}{(Delay + K * Jitter) * e^{\wedge (packet\ loss)}}$$

De tal forma, y a título ilustrativo, se ejemplifica aquí para el Codec H.264. La fórmula siguiente obtiene el valor de QoE para el máximo jitter y delay obtenido:

$$\frac{1}{2{,}032408 + 2 * 0{,}059028} = 0{,}46501$$

De la misma manera, la siguiente expresión obtiene el valor de QoE para el promedio de valores del jitter y delay:

$$\frac{1}{2{,}026842 + 2 * 0} = 0{,}49337$$

Y la siguiente, el valor de QoE para el mínimo jitter y delay:

$$\frac{1}{1{,}969907 + 2 * -0{,}061343} = 0{,}54135$$

Teniendo en cuenta esto, y para la representación gráfica de las tendencias, nuevamente aquí, se desarrolló código Python específico. En el IDE, se procedió a la utilización de librerías de la sección panda, openpyxl y algunas como mathplolib para los gráficos. Esto debido, especialmente, a la necesidad de manipular una gran cantidad de tramas.

De esta forma fue posible obtener múltiples gráficas, para sus análisis, representando la QoE en función del retardo o de jitter, para los valores promedio, máximo y mínimo de su contraparte como parámetro.

Si se desea conocer con más detalle el procedimiento, para obtener la QoE en su totalidad, se puede consultar el **ANEXO B.**

V.4. ANÁLISIS DE LA QoE PARA EL CÓDEC H.264

En las Figuras Nº 48 y 49 se representan los valores de QoE en ordenadas, en función de los valores de retardo, en el rango de trabajo, utilizando el jitter promedio o el jitter máximo como parámetro, respectivamente, para el caso del códec H.264.

De esta forma, se muestra en la Figura Nº 48 que el jitter promedio de 0 mseg genera una línea recta descendente para la QoE, con un valor de 0.508 para un retardo de 1.97 mseg, y de 0.492 para un valor de retardo de 2.03 mseg. Es decir, que ha medida que el retardo disminuye, la QoE crece con una pendiente que depende del jitter.

Por otro lado, en la Figura Nº 49, se observa que el jitter máximo de 0.059028 msg genera una línea recta descendente para la QoE, con un valor de 0.480 para un retardo de 1.97 mseg, y de 0.466 para un valor de retardo de 2.03 mseg. Es decir, que ha medida que el retardo disminuye, la QoE crece con una menor pendiente que la dependencia del valor promedio del jitter.

Como dato adicional deberá tenerse en cuenta que en cada una de las figuras que aparecen en esta sección hay un recuadro mas dentro del recuadro que remarca la imagen. Este recuadro interno es un indicativo que muestra los valores de degradación en base a los colores según los valores de QoE. Por ejemplo, a medida que el valor de calidad de experiencia es mayor se podrá observar un color mas violeta, así mismo a medida que va bajando este valor la degradación torna a un color mas rosa claro.

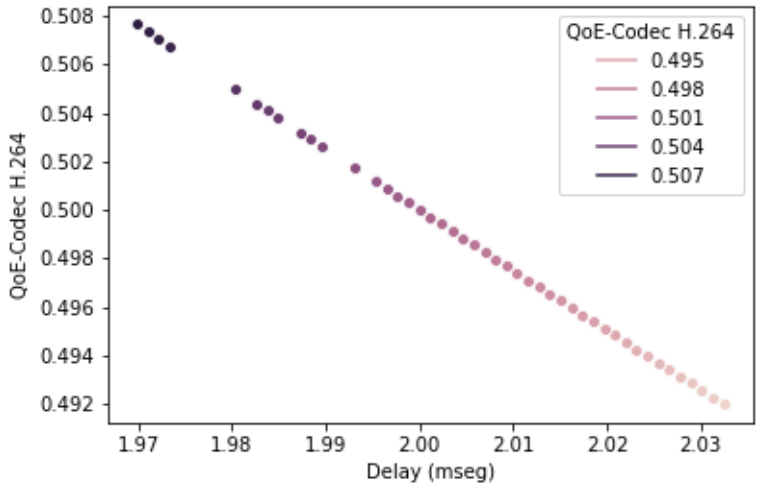


Figura N° 48: Ilustra la QoE en función del retardo y el jitter promedio como parámetro fijo para el códec H.264.

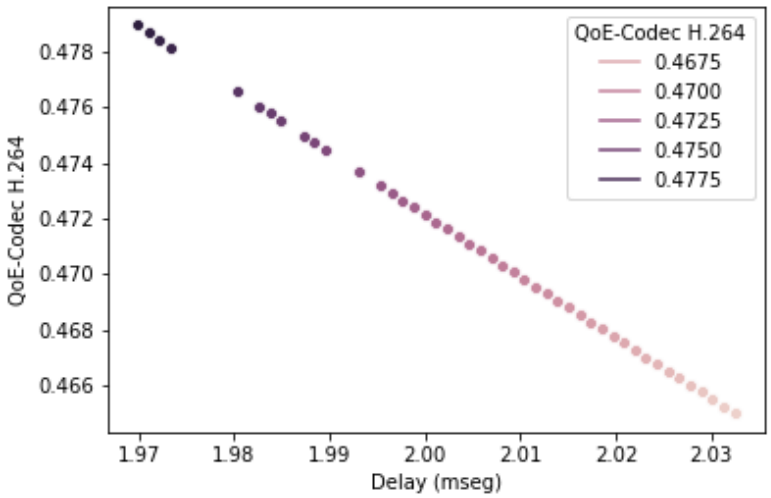


Figura N° 49: Ilustra la QoE en función del retardo y el jitter máximo como parámetro fijo para el códec H.264.

Mientras que en las Figuras Nº 50 y 51 se representan los valores de QoE en ordenadas, en función de los valores de jitter, en el rango de trabajo, utilizando el retardo promedio o el retardo máximo como parámetro, respectivamente, para el caso del códec H.264.

De esta forma, se muestra en la Figura Nº 50, que el retardo promedio de 2.026842 mseg genera una línea recta descendente para la QoE, con un valor

de 0.52 para un jitter de -0.06 mseg, y de 0.47 para un valor de jitter de 0.05 mseg. Es decir, que ha medida que el jitter disminuye, la QoE crece con una pendiente que depende del retardo.

Por otro lado, se muestra en la Figura Nº 51, que el retardo máximo de 2,032408 mseg genera una línea recta descendente para la QoE, con un valor de 0.54 para un jitter de -0.06 mseg, y de 0.47 para un valor de jitter de 0.05 mseg. Es decir, que ha medida que el jitter disminuye, la QoE crece con una pendiente que depende del retardo.

Los valores son bastantes similares por la poca diferencia en los valores promedio y máximo del retardo.

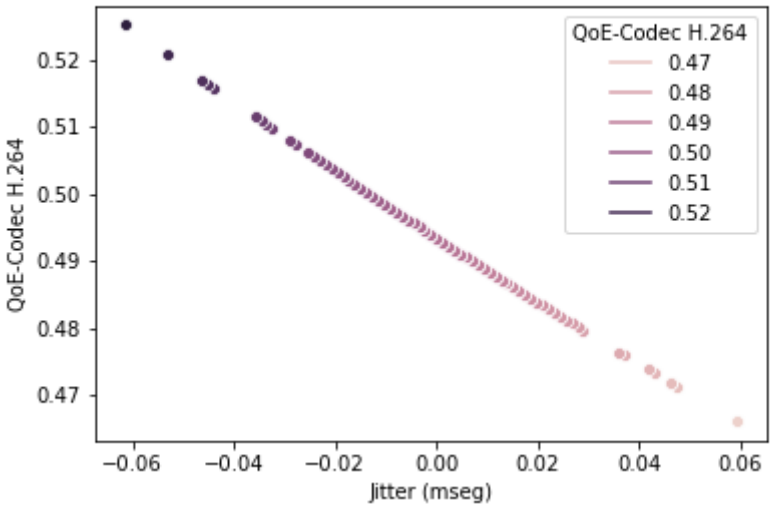


Figura N° 50: Ilustra la QoE en función del jitter y del retardo promedio como parámetro fijo para el códec H.264.

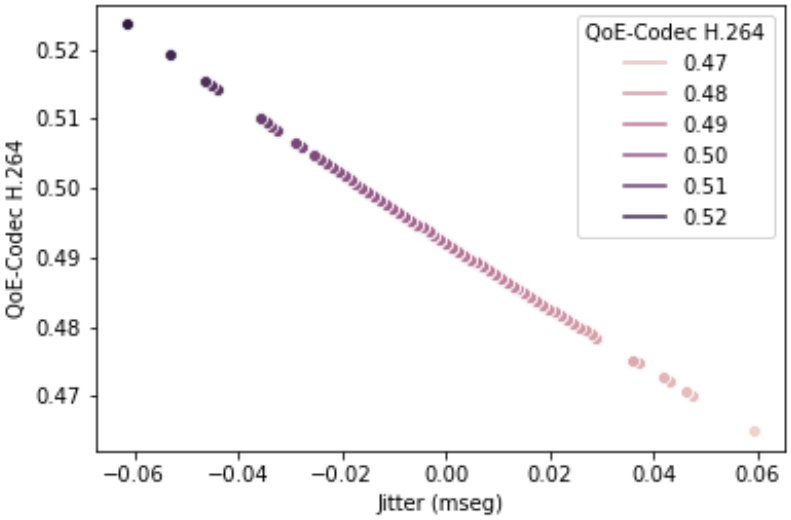


Figura N° 51: Ilustra la QoE en función del jitter y del retardo máximo como parámetro fijo para el códec H.264.

Finalmente, en la Figura Nº 52, y a los fines comparativos, se presentan los 4 escenarios anteriores en una sola figura:

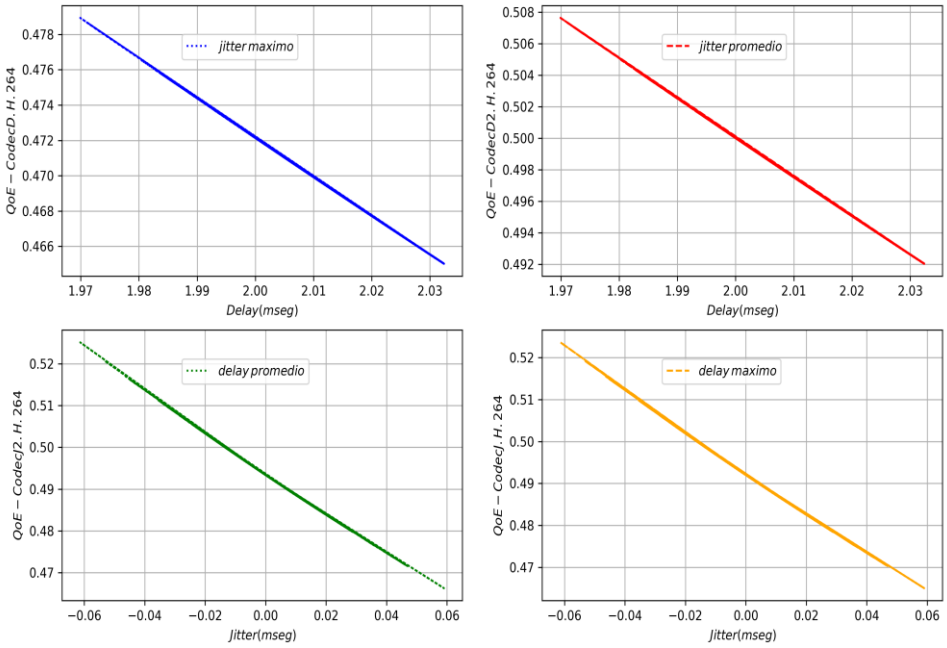


Figura N° 52: Ilustra la QoE en función del retardo o el jitter, para un valor promedio y máximo de los mismos respectivamente.

V.5. ANÁLISIS DE LA QoE PARA EL CÓDEC H.265

En las Figuras Nº 53 y 54 se representan los valores de QoE en ordenadas, en función de los valores de retardo, en el rango de trabajo, utilizando el jitter promedio o el jitter máximo como parámetro, respectivamente, para el caso del códec H.265.

De esta forma, se muestra en la Figura Nº 53, que el jitter promedio de 0 mseg genera una línea recta descendente para la QoE, con un valor de 0.510 para un retardo de 1.97 mseg, y de 0.490 para un valor de retardo de 2.04 mseg. Es decir, que ha medida que el retardo disminuye, la QoE crece con una pendiente que depende del jitter.

Por otro lado, en la Figura Nº 54, se observa que el jitter máximo de 0,059028 mseg genera una línea recta descendente para la QoE, con un valor de 0.490 para un retardo de 1.97 mseg, y de 0.460 para un valor de retardo de 2.06 mseg. Es decir, que ha medida que el retardo disminuye, la QoE crece con una pendiente menor, que depende del jitter máximo, respecto a como lo hace cuando se usa el valor de jitter promedio.

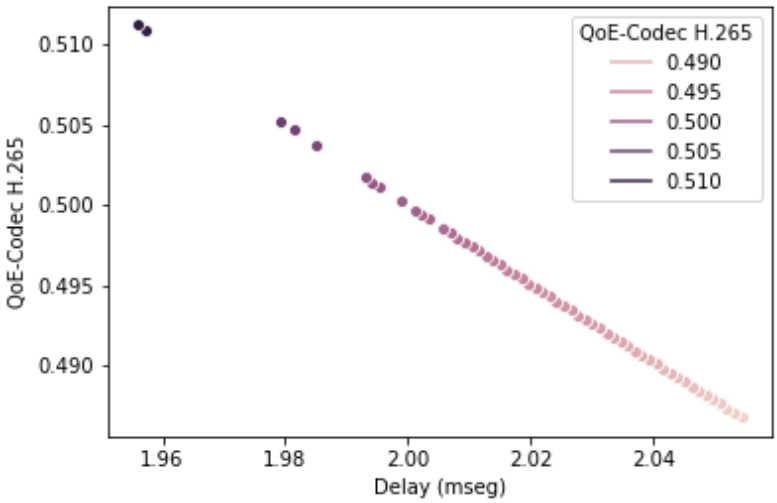


Figura N° 53: Ilustra la QoE en función del retardo y el jitter promedio como parámetro fijo para el códec H.265.

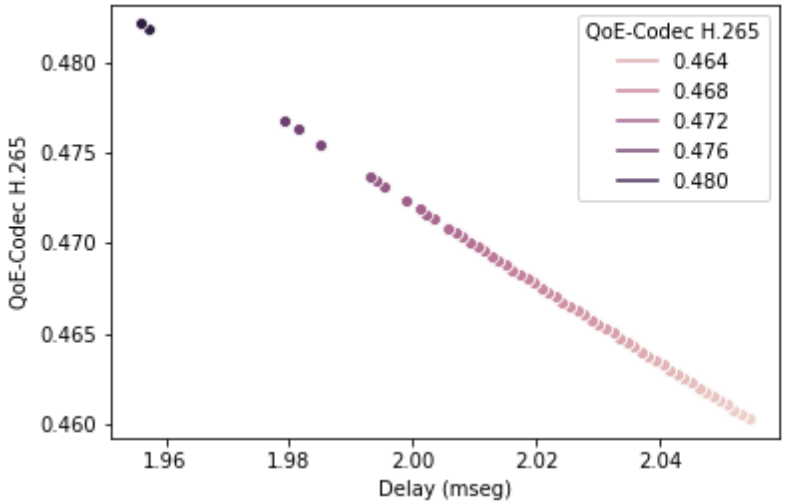


Figura N° 54: Ilustra la QoE en función del retardo y del jitter maxímo como parámetro fijo para el códec H.265.

Por otro lado, en las Figuras Nº 55 y 56 se representan los valores de QoE en ordenadas, en función de los valores de jitter, en el rango de trabajo, utilizando el retardo promedio o el retardo máximo como parámetro, respectivamente, para el caso del códec H.265.

De esta forma, se muestra en la Figura Nº 55, que el retardo promedio de 2,048551 mseg genera una línea recta descendente para la QoE, con un valor de 0.53 para un jitter de -0.08 mseg, y de 0.45 para un valor de jitter de 0.05 mseg. Es decir, que ha medida que el jitter disminuye, la QoE crece con una pendiente que depende del retardo.

Por otro lado, se muestra en la Figura Nº 56, que el retardo máximo de 2,054399 mseg genera una línea recta descendente para la QoE, con un valor de 0.53 para un jitter de -0.08 mseg, y de 0.45 para un valor de jitter de 0.04 mseg. Es decir, que ha medida que el jitter disminuye, la QoE crece con una pendiente que depende del retardo. Los valores son bastantes similares por la poca diferencia en los valores promedio y máximo del retardo.

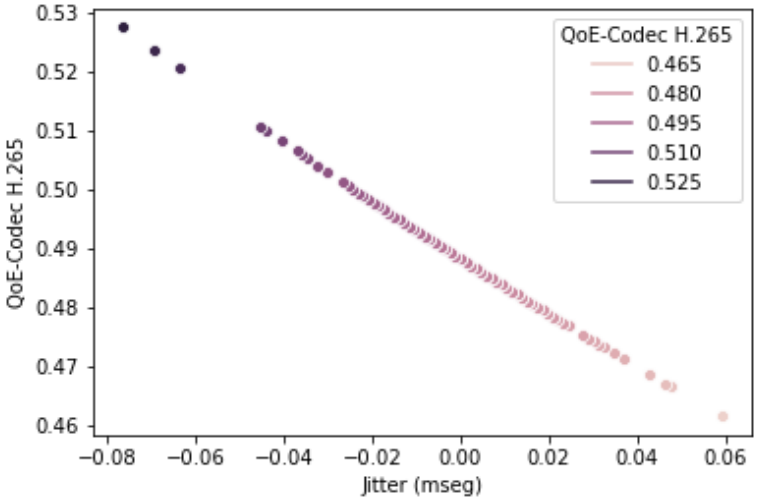


Figura N° 55: Ilustra la QoE en función del jitter y del retardo promedio como parámetro fijo para el códec H.265.

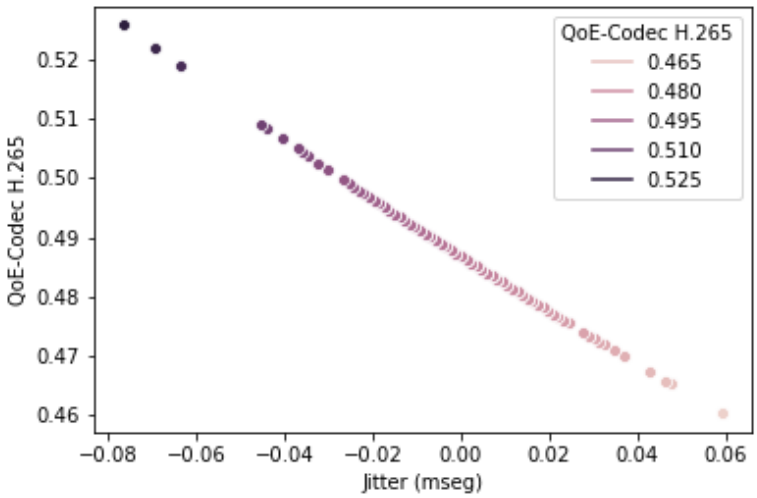


Figura N° 56: Ilustra la QoE en función del jitter y del retardo máximo como parámetro fijo para el códec H.265.

Por último, en la Figura Nº 57, y a los fines comparativos, se presentan los 4 escenarios anteriores en una sola figura:

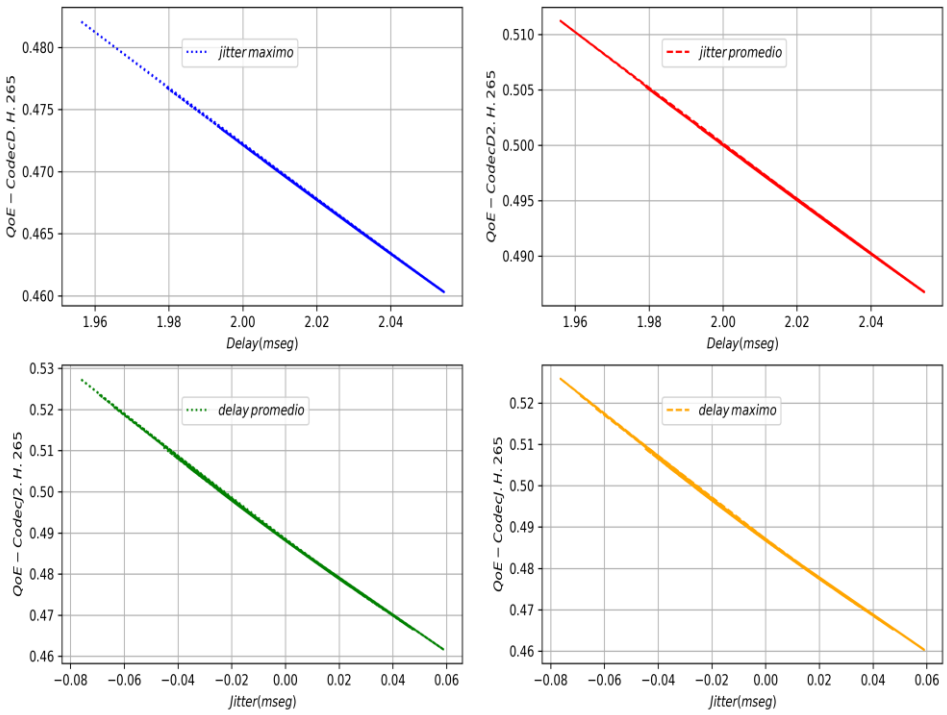


Figura N° 57: Muestra la QoE en función del retardo y del jitter, para valores promedio y máximo de los mismos.

V.6. ANÁLISIS DE LA QoE PARA EL CÓDEC THEORA

En las Figuras Nº 58 y 59 se representan los valores de QoE en ordenadas, en función de los valores de retardo, en el rango de trabajo, utilizando el jitter promedio o el jitter máximo como parámetro, respectivamente, para el caso del códec Theora.

De esta forma, se muestra en la Figura Nº 58, que el jitter promedio de 0 mseg genera una línea recta descendente para la QoE, con un valor de 0.86 para un retardo de 1.120 mseg, y de 0.75 para un valor de retardo de 1.330 mseg. Es decir, que ha medida que el retardo disminuye, la QoE crece con una pendiente que depende del jitter.

Por otro lado, en la Figura Nº 59, se observa que el jitter máximo de 0,12963 mseg genera una línea recta descendente para la QoE, con un valor de 0.71 para un retardo de 1.120 mseg, y de 0.62 para un valor de retardo de 1.326 mseg. Es decir, que ha medida que el retardo disminuye, la QoE crece con una pendiente que depende del jitter, aunque en menor medida que lo hace con el jitter promedio.

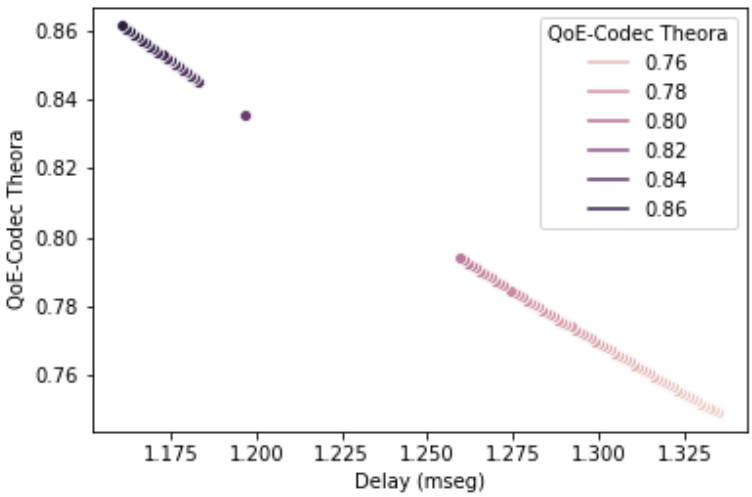


Figura N° 58: Ilustra la QoE en función del retardo y del jitter promedio como parámetro fijo para el códec Theora.

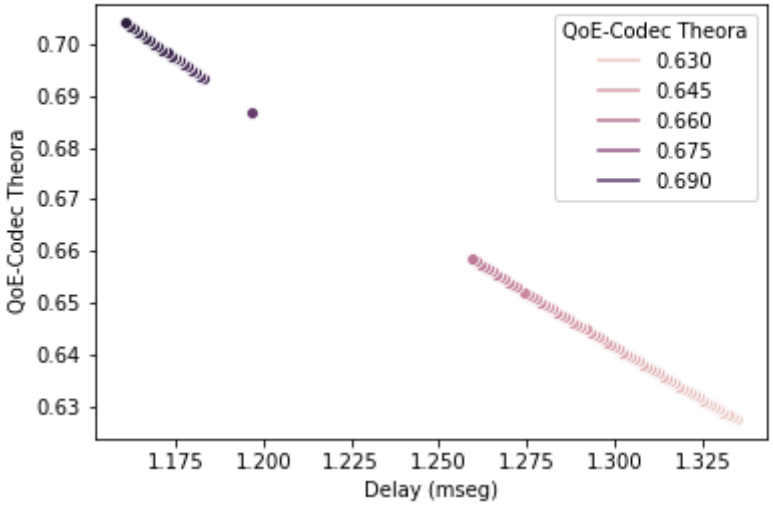


Figura N° 59: Ilustra la QoE en función del retardo y del jitter máximo como parámetro fijo para el códec Theora.

Además, en las Figuras Nº 60 y 61 se representan los valores de QoE en ordenadas, en función de los valores de jitter, en el rango de trabajo, utilizando el retardo promedio o el retardo máximo como parámetro, respectivamente, para el caso del códec Theora.

De esta forma, se muestra en la Figura Nº 60, que el retardo promedio de 1,32442 mseg genera una línea recta descendente para la QoE, con un valor de 0.80 para un jitter de -0.05 mseg, y de 0.70 para un valor de jitter de 0.05 mseg. Es decir, que ha medida que el jitter disminuye, la QoE crece con una pendiente que depende del retardo.

Por otro lado, se muestra en la Figura Nº 61, que el retardo máximo de 1,334491 mseg genera una línea recta descendente para la QoE, con un valor de 0.78 para un jitter de -0.05 mseg, y de 0.72 para un valor de jitter de 0.05 mseg. Es decir, que ha medida que el jitter disminuye, la QoE crece con una pendiente que depende del retardo. Los valores son bastantes similares por la poca diferencia en los valores promedio y máximo del retardo.

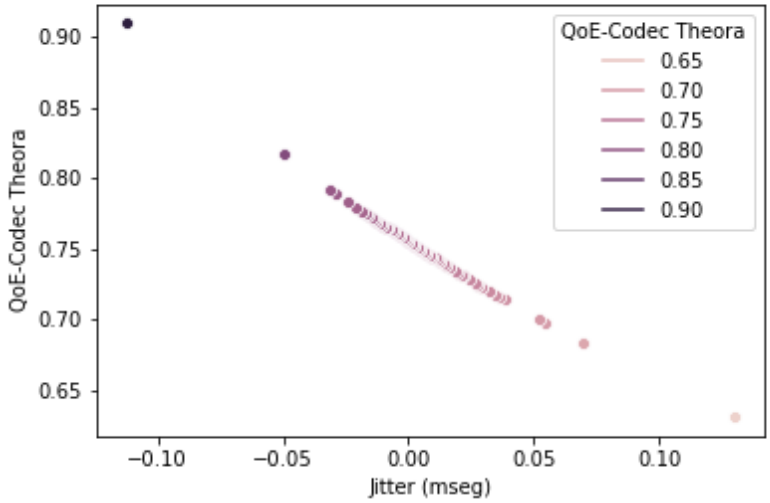


Figura N° 60: Ilustra la QoE en función del jitter y del retardo promedio como parámetro fijo para el códec Theora.

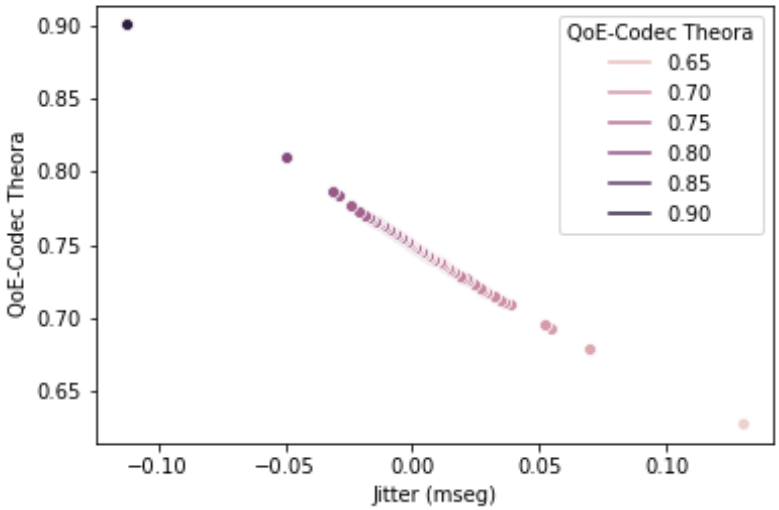


Figura N° 61: Ilustra la QoE en función del jitter y del retardo máximo como parámetro fijo para el códec Theora.

Por último, en la Figura Nº 62, y a los fines comparativos, se presentan los 4 escenarios anteriores en una sola figura:

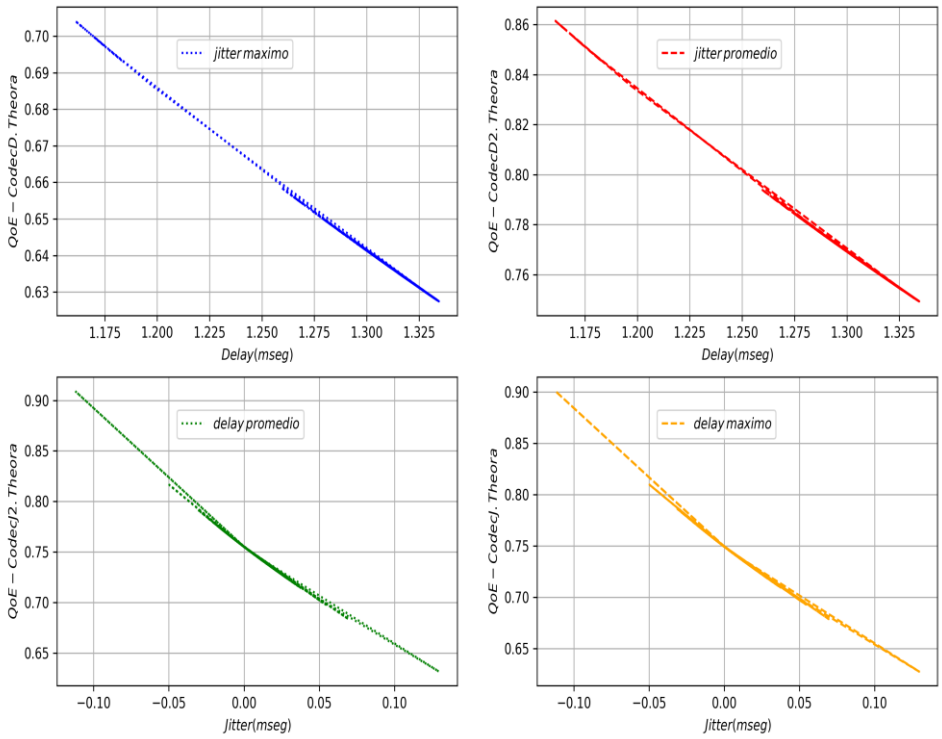


Figura N° 62: Muestra la QoE en función del retardo y del jitter, para valores promedio o máximo de los mismos respectivamente.

V.7. ANÁLISIS DE LA QoE PARA EL CÓDEC VP8

En las Figuras Nº 63 y 64 se representan los valores de QoE en ordenadas, en función de los valores de retardo, en el rango de trabajo, utilizando el jitter promedio o el jitter máximo como parámetro, respectivamente, para el caso del códec VP8.

De esta forma, se muestra en la Figura Nº 63, que el jitter promedio de 0 mseg genera una línea recta descendente para la QoE, con un valor de 1.05 para un retardo de 0.97 mseg, y de 0.97 para un valor de retardo de 1.03 mseg. Es decir, que ha medida que el retardo disminuye, la QoE crece con una pendiente que depende del jitter.

Por otro lado, en la Figura Nº 64, se observa que el jitter máximo de 0,059029 mseg genera una línea recta descendente para la QoE, con un valor de 0.94 para un retardo de 0.98 mseg, y de 0.87 para un valor de retardo de 1.03 mseg. Es decir, que ha medida que el retardo disminuye, la QoE crece con una pendiente menor, que depende del jitter máximo, respecto a como lo hace con el jitter promedio.

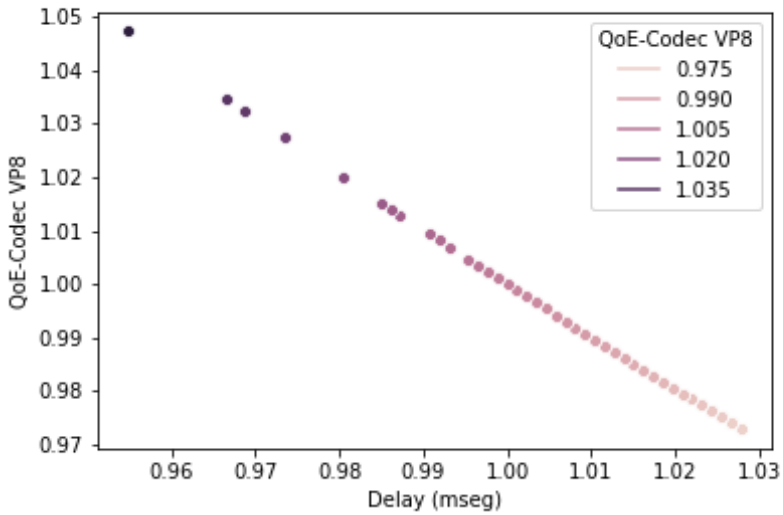


Figura N° 63: Muestra la QoE en función del retardo y del jitter promedio como parámetro fijo para el códec VP8.

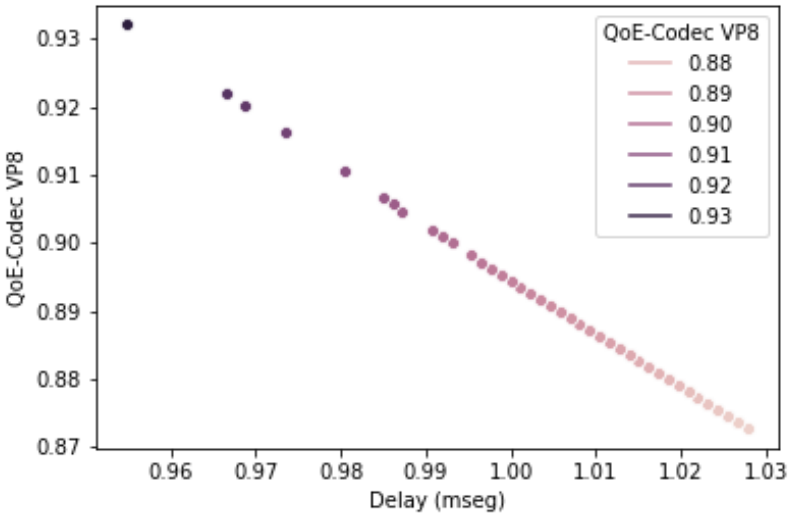


Figura N° 64: Ilustra la QoE en función del retardo y del jitter máximo como parámetro fijo para el códec VP8.

Mientras que en las Figuras Nº 65 y 66 se representan los valores de QoE en ordenadas, en función de los valores de jitter, en el rango de trabajo, utilizando el retardo promedio o el retardo máximo como parámetro, respectivamente, para el caso del códec VP8.

De esta forma, se muestra en la Figura Nº 65, que el retardo promedio de 1,022103 mseg genera una línea recta descendente para la QoE, con un valor de 1.09 para un jitter de -0.06 mseg, y de 0.90 para un valor de jitter de 0.04 mseg. Es decir, que ha medida que el jitter disminuye, la QoE crece con una pendiente que depende del retardo.

Por otro lado, se muestra en la Figura Nº 66, que el retardo máximo de 1,027778 mseg genera una línea recta descendente para la QoE, con un valor de 1.08 para un jitter de -0.06 mseg, y de 0.92 para un valor de jitter de 0.04 mseg. Es decir, que ha medida que el jitter disminuye, la QoE crece con una pendiente que depende del retardo. Los valores son bastantes similares por la poca diferencia en los valores promedio y máximo del retardo.

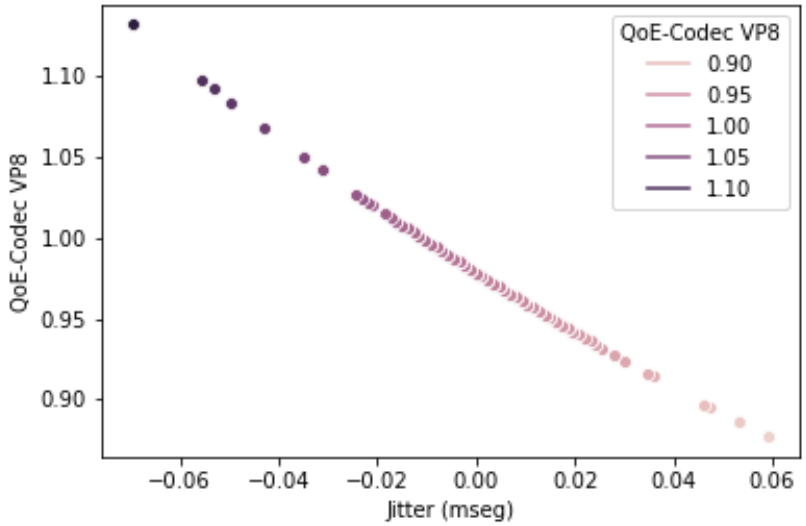


Figura N° 65: Ilustra la QoE en función del jitter y del retardo promedio como parámetro fijo para el códec VP8.

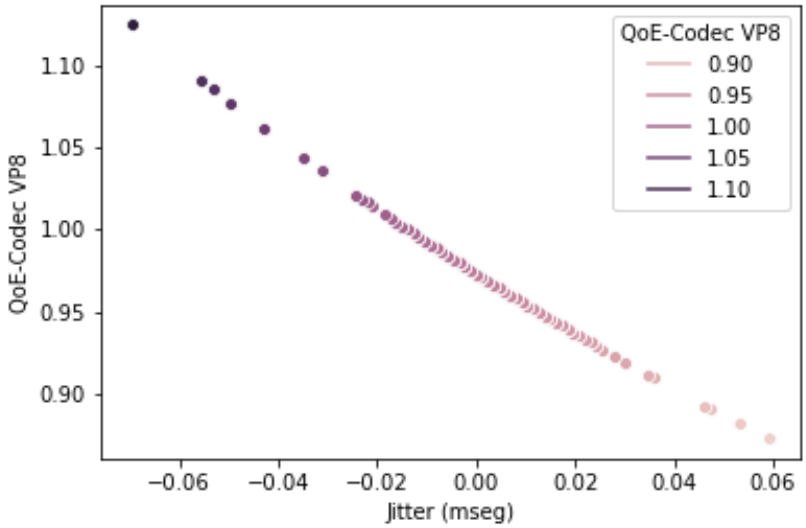


Figura N° 66: Ilustra la QoE en función del jitter y del retardo máximo como parámetro fijo para el códec VP8.

Por último, en la Figura Nº 67, y a los fines comparativos, se presentan los 4 escenarios anteriores en una sola figura:

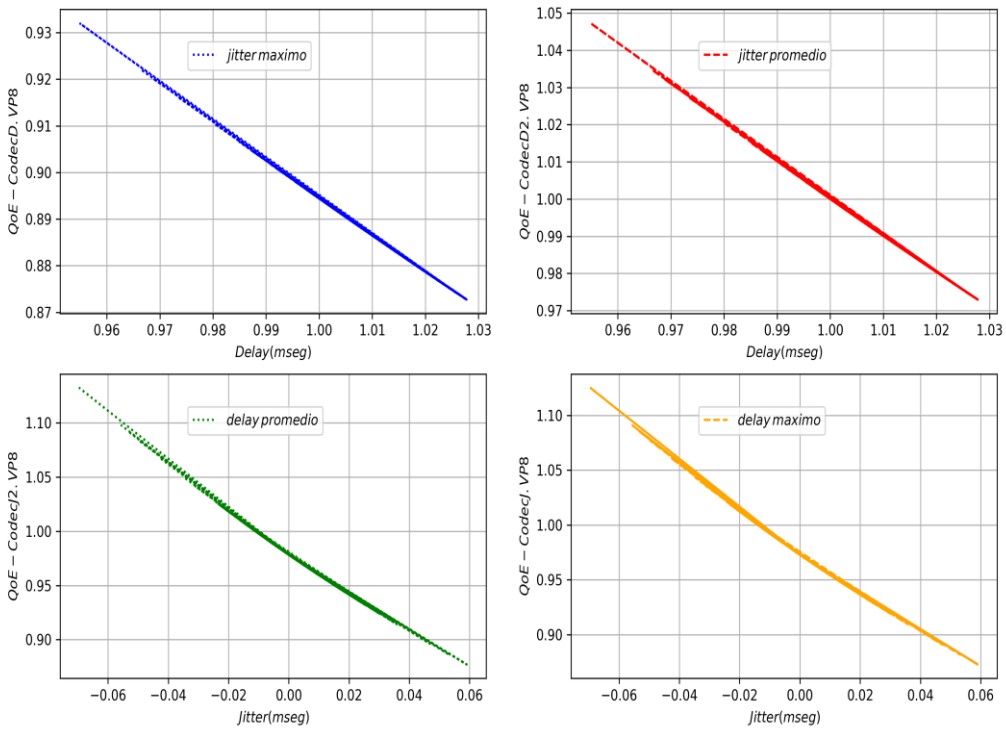


Figura N° 67: Muestra la QoE en función del retardo y del jitter, para valores promedio y máximo de los mismos respectivamente.

V.8. ANÁLISIS COMPARATIVO DEL COMPORTAMIENTO DE MÉTRICAS DE QoE DE LOS CÓDECS EN LA RED EXPERIMENTAL

V.8.1. Análisis de la métrica Retardo

En la Figura Nº 68 se observa, a los fines comparativos, la métrica de retardo para cada uno de los códecs analizados. Se observa qué en la banda de 1 a 2 ms, los codecs presentan un comportamiento claramente diferenciado, a excepción de los codecs H.264 y H.265. Estos últimos codecs tienen una peor respuesta para la QoE. El códec VP8 presenta un mejor comportamiento, mientras que el códec Theora se ubica en una respuesta intermedia.

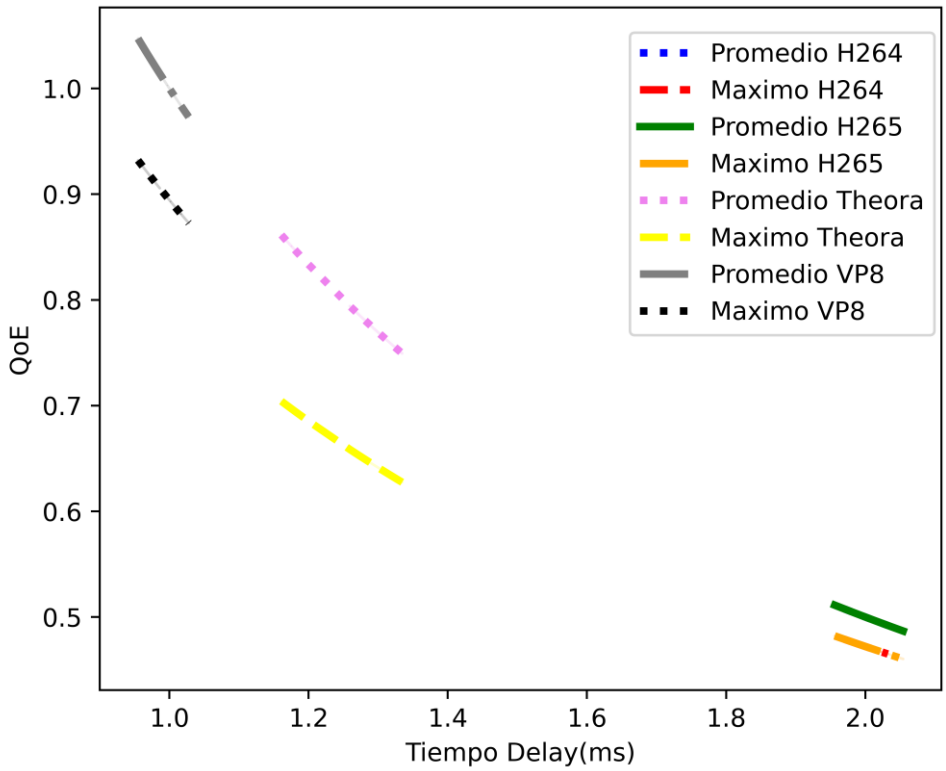


Figura N° 68: Muestra la comparación de los valores de QoE en función del retardo, para valores promedio y máximo del jitter en los 4 codecs.

Como se destaca, para todos los codecs, para un valor de delay dado, a mayor jitter menor valor de la QoE. Además, a menores tiempos de jitter la calidad de experiencia es gradualmente mejor. Para una mayor claridad en su cálculo se puede observar **el ANEXO D.**

V.8.2. Análisis de la métrica Jitter

En la Figura Nº 69 se presenta, a los fines comparativos, la métrica de jitter para cada uno de los códecs analizados. Se observa que en la banda de -0,10 mseg a +0,15 mseg, los códecs presentan un comportamiento claramente diferenciado, a excepción de los códecs H.264 y H.265. Estos últimos códecs tienen una peor respuesta para la QoE. Nuevamente, el códec VP8 presenta un mejor comportamiento, mientras que el códec Theora se ubica en una respuesta intermedia.

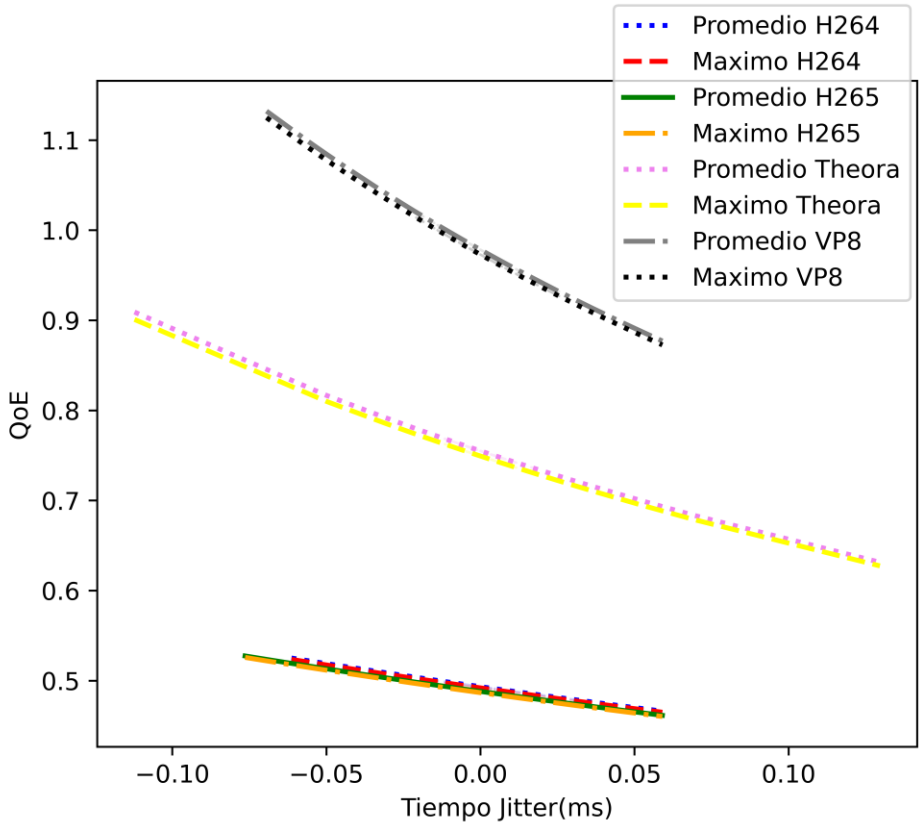


Figura N° 69: Muestra la Comparación de la QoE en función del jitter, para los valores promedio y máximo del retardo en los 4 codecs.

Para todos los códecs, para un valor de jitter dado, a mayor retardo menor valor de la QoE. Además, a menores tiempos de retardo la calidad de experiencia es gradualmente mejor.

A continuación, si desea obtener un mayor conocimiento de la programación aplicada, para establecer las gráficas se puede consultar el **ANEXO D.**

CAPÍTULO VI

VI.CONCLUSIONES y TRABAJOS FUTUROS

Este capítulo resume las contribuciones y conclusiones principales obtenidas a través de esta tesis. Se plantean los trabajos presentes en desarrollo y los posibles trabajos futuros como una continuación del presente trabajo de tesis.

VI.1. INTRODUCCIÓN

Como se puede analizar, desde hace años, la industria de la televisión digital esta atravesando una profunda transición, migrando de la televisión convencional a una nueva era de tecnología digital. Con el avance exponencial de los servicios, la gran mayoría de los operadores de TV vienen actualizando sus redes y, además, vienen implementando plataformas digitales avanzadas, en un esfuerzo por seguir manteniendo sus suscriptores de sistema de servicios analógicos tradicionales, para crear una transcisión abrupta a los servicios digitales mas sofisticados.

Las empresas obtienen grandes beneficios cuando proveen la transmisión de señales de TV, entre otras cosas cuando estas proporcionan mayor interactividad, reflejándose en las mejoras de los tiempos de cambio de canales y en la interoperabilidad del uso de las redes domésticas, cuando los usuarios disponen de dichos servicios de IPTV.

En este contexto, siendo lo más importante para la motivación del presente trabajo de investigación de esta tesis fue analizar el comportamiento de tráfico IPTV, en una red LAN experimental con tráfico controlado. Se utilizaron diferentes códecs para contraste, y se establecieron resultados cuantitativos detallados de las métricas de QoS y, a partir de ellas, valores indicativos de QoE, que orientan sobre su conducta.

VI.2. CONTRIBUCIONES PRINCIPALES

La contribución principal de esta tesis es:

1. Los resultados cuantitativos de los estudios por experimentación sobre una red LAN de laboratorio real, propuesto al efecto, para evaluar el comportamiento de QoS y de QoE, para tráfico IPTV y diferentes códecs, en un todo de acuerdo a la hipótesis.

Y las contribuciones secundarias son:

2. Las conclusiones de dicho estudio podrán ser utilizadas como referencias a contextos similares, y por analistas de simulación para la parametrización de los simuladores de tráfico IPTV.
3. La definición de una nueva topología, metodología, y de diferentes subescenarios de experimentación, según el códec utilizado, para su uso en la evaluación de los mismos y otros aspectos complementarios de streamings de video IPTV, y de la QoS y QoE relacionadas.

VI.3. CONCLUSIÓN

Se analizaron detalladamente los alcances de la QoS y de QoE para el tráfico IPTV sobre una red LAN experimental, en el ámbito de la UTN Regional Mendoza.

Se utilizó una topología de red de experimentación, un servidor y clientes de video IPTV, un trailer de Star Trek, con 4 subescenarios o casos particulares, por cada uno de 4 codecs: H.264, H.265, Theora y VP8. El tráfico fue capturado para análisis, en todos los casos, usando un sniffer. Y procesado usando rutinas en lenguaje Python.

Para proponer instancias de comparación, en los sub escenarios propuestos, se establecieron los siguientes puntos críticos o de verificación de QoS:

- Retardo Promedio,
- Retardo Máximo,
- Retardo Mínimo,
- Jitter Promedio,
- Jitter Máximo, y
- Jitter Mínimo.

Se demostró experimentalmente que:

- La métrica de retardo para tráfico IPTV, en la red experimental, satisface holgadamente los requerimientos de QoS, y presenta un comportamiento claramente diferenciado, a excepción de los códecs H.264 y H.265. Estos últimos códecs concentran sus retardos en el orden de los 2 mseg, mientras que el códec Theora lo hace en el orden de los 1,5 mseg. Finalmente, el códec VP8 concentra sus retardos en el orden de 1 mseg.
- La métrica jitter para tráfico IPTV, en la red experimental, satisface holgadamente los requerimientos de QoS, y presenta un comportamiento claramente diferenciado. Cada códec tiene una respuesta en la forma de triángulo, cuando se representa la cantidad de valores, en repetición, en función del jitter. Los códec tienen bases y alturas distintas. El códec H.264 presenta la peor respuesta general, mientras que el caso contrario esta visualizado para el códec VP8.
- La respuesta de QoE, en base a la expresión matemática utiliza, usando el retardo (jitter) como variable independiente, y el jitter (retardo) como parámetro, muestra que los H.264 y H.265 tienen el peor comportamiento. El códec VP8 presenta el mejor comportamiento, mientras que el codec Theora se ubica en una respuesta intermedia.

VI.4. TRABAJOS PRESENTES Y FUTUROS

La experiencia reunida con los estudios realizados sobre la QoS y QoE del tráfico IPTV en una red LAN, utilizando diferentes códecs abrió una serie de alternativas de profundización de la línea de investigación, con el objeto de avanzar el perfeccionamiento de los resultados de QoE, incluyendo algunas consideraciones de subjetividad del usuario.

La calidad de la experiencia (QoE) para contenidos multimedia, como IPTV, también se encuentra definida por la organización de estándares de la industria ETSI TISPAN (European Telecommunications Standards Institute Telecommunications and Internet converged Services and Protocols for Advanced Networking) en su norma TR 102 479,2.

El MOS (Mean Opinion Score) es una medida que se ha utilizado durante décadas en las redes de telefonía, para obtener la calidad de la red bajo el punto de vista del usuario. Inicialmente, y según la recomendación ITU---T P.800, MOS fue una medida subjetiva, donde los oyentes se sentaban en una habitación "tranquila", y la puntuación de la calidad de una llamada dependía de lo que ellos percibian. Mas adelante, la medición de Voz sobre IP (VoIP) se hizo mas objetiva, basándose en el cálculo del rendimiento de la red IP.

Cabe destacar que el MOS proporciona una indicación numérica de la calidad percibida desde la perspectiva de los usuarios, después de la compresión y/o transmisión multimedia. El MOS se expresa como un solo número en el intervalo de 1 a 5, donde 1 es la más baja calidad percibida de audio/video, y 5 es la mas alta. En la Tabla siguiente se muestra una posible clasificación del MOS, en base a la calidad de audio/video percibida por el usuario.

Percepción	*MOS*	*Deterioro de la calidad*
Excelente	5	Imperceptible
Bueno	4	Perceptible pero no molesto
Razonable	3	Ligeramente molesto
Pobre	2	Un poco molesto
Malo	1	Muy molesto

Tabla N° 15: Indica la perspectiva numérica de calidad MOS.
Fuente: https://www.itu.int/

El objetivo a futuro es incluir en la QoE, analizada preliminarmente en esta tesis, donde se evalúa las expectativas de los usuarios en el contexto experimental utilizado. Este componente subjetivo puede diferir, lógicamente, de un usuario a otro.

VII.BIBLIOGRAFIA

[1] **Driscoll G.** "Next Generation IPTV Services and Technologies". (1ª Ed.)". Editorial: Wiley-Interscience, Canada, 2018.

[2] **H. A. Facchini, S. C. Perez, A. Dantiacq, F .Hidalgo**. "Evaluación de métricas del comportamiento del tráfico de vídeo en una red experimental multidifusión", Centro de Investigación y Desarrollo en Computación y Neuroingeniería. CeReCoN. Universidad Tecnologica Nacional, UTN. Mendoza, Junio 2020.

[3] **H. A. Facchini, S. C. Perez, F .Hidalgo**, **P. Varela**. "Análisis, simulación y estudio experimental del comportamiento de métricas de QoS y QoE de streamings de video multicast IPTV. Centro de Investigación y Desarrollo en Computación y Neuroingeniería. CeReCoN. Universidad Tecnologica Nacional. UTN. Mendoza, Mayo 2020.

[4] **H. A. Facchini, S. C. Perez, F .Hidalgo**, **P. Varela**. "Análisis, simulación y estudio experimental del comportamiento de métricas de QoS y QoE de streamings de video multicast IPTV. Centro de Investigación y Desarrollo en Computación y Neuroingeniería. CeReCoN. Universidad Tecnologica Nacional. UTN. Mendoza, Mayo 2020.

[5] **S. C. Perez, H. A. Facchini, B. Roberti, A. Dantiacq, F. Hidalgo, M. Stafononi, M. Césari, P. Varela** – Co-autor del trabajo en el WICC 2020, XXII Workshop de Investigadores en Ciencias de la Computación, con el tema " Análisis, simulación y estudio experimental del comportamiento de métricas de QoS y QoE de streamings de video multicast IPTV – Caso de Estudio en la red de la UTN - Mendoza" perteneciente al Area "Arquitectura, Redes y Sistemas Operativos", organizado por la Universidad Nacional de la Patagonia Austral (UNPA) y se realizará en la ciudad El Calafate, Pcia. de Santa Cruz, Argentina, los días 7 y 8 mayo/20.

[6] **J. Partison.** "IPTV: History and Market Growth". Available: IPTV: History and Market Growth (coda.io)

[7] **S. F. Arellano**. "Diseño de la red Hibrida Coaxial-Fibra Optica (HFC) para brindar servicio IPTV en la empresa multicable S.A. de la ciudad de Otavalo". Available: https://docplayer.es/23341563-A-componentes-de-un-sistema-iptv.html

[8] **S. Fichamba Arellano**. "Diseño de la Red Hibrida Coaxial-Fibra Óptica (HFC) para brindar Servicio IPTV en la empresa MULTICABLE S.A de la ciudad de Otavalo", Docplayer paper, Universidad técnica del Norte, Ibarra Ecuador. pp 2-15.

[9] **W. Beau**. "Developing IP multicast networks", vol. 1: Cisco system. Indianápolis: Cisco Press, 2000.

[10] **Salekul Islam; J. Willam Atwood.** "The Internet Group Management Protocol with Access Control (IGMP-AC)" in 2006 International Conference on Local Computer Network. IEEE, 2007.

[11] STB Animo A139, Available: AMINO A139 GUIA DE INSTALACION Descargar en PDF | ManualsLib

[12] FFMEPG, "Documentation", [Online]. Available: https://ffmpeg.org/documentation.html

[13] **H. A. Facchini, S. C. Perez, A. Dantiacq, F. Hidalgo. "**Evaluación de métricas del comportamiento del tráfico de video en una red experimental multidifusión". ResearcheGate paper. Enfoque UTE. Universidad Tecnologica Nacional. e-ISSN: 1390-6542 / p-ISSN: 1390-9363. pp 15-27.Junio 2020.

[14] YOUTUBE, "Aplicación", [Online]. Available: https://youtu.be/g5lWao2gVpc

[15] **B. Febrero**. "Análisis de Tráfico con Wireshark". ITECO-CERT. pp 5-52 Febrero 2011.

[16] PYTHON, "Documentation", [Online]. Available: https://www.python.org/doc/

[17] **McKinney W.** "Python for Data Analysis". (2ª Ed.)".Editorial: O'Reilly Media. Sebastopol, United State, 2017.

[18] **S. C. Perez, H. A. Facchini, A. Dantiacq, F. Hidalgo, G. Q. Salomón, G. Cangemi, F. Hidalgo.** "Comparación del comportamiento de los códecs de video en el entorno Wi-Fi IEEE 802.11ac". CeReCoN, Escuela TIC. Universidad de Chilecito. Universidad Tecnologica Nacional, UTN. Mendoza, Diciembre 2020.

[19] **J. Lloret, A Canovas, J. J. P. C. Rodriguez, K. Lin.** "A network algorithm for 3D/2D IPTV distribution using WIMAX and WLAN technologies" Springer Science-Business Media. DOI 10.1007/s11042-011-0929-4. 11 de Noviembre de 2011.

[20] **J. C. C. Valencia, W. C. Muñoz, G. E. C Golondrino**. "Análisis de QoS para IPTV en un entorno de redes definidas por software". Revista Ingenierías Universidad de Medellín. ISSN (en línea): https://doi.org/10.22395/rium.v18n36a2. 9 de Julio de 2019.

[21] **J. Cuellar, J. Arciniegas, J. Ortiz.** "Modelo para la medición de QoE en IPTV". Editorial: Universidad Ecesi. Colombia, 2018.

[22] **J. M. J. Herranz., J. L. Mauri**. "Estudio de la variación de QoE en Televisión IP cuando varían los parámetros de QoS". Tesis, Trabajo Final de Graduación en (Post Producción Digital) Master. Universidad Politecnica de Valencia, Gandia 2014.

[23] **J. C. Cuellar, S. Acosta, J. L. Arciniegas**. "QoE/QoS Mapping Models to measure Quality of Experience to IPTV Service". Conferencia Paper. Publicación: ResearchGate. Octubre 2018.

[24] **G. Baltoglou, E. Karapistoli, P. Chatzimisios**. "IPTV QoS and QoE Measurements in Wired and Wireless Networks". Publicación: Globecom. Editor IEEE. 23 de Abril de 2013.

[25] **A. C. Solbes, D. J. L. Mauri, D. J. T. Gironés**. "Diseño y Desarrollo de un Sistema de Gestión Inteligente integrado de servicios de IPTV estándar, estereoscópico y HD basado en QoE". Tesis, Trabajo Final de Graduación en (Ing. Sist. De Telocom., Sonido e imagen). Universidad Politecnica de Valencia, Gandia 9 de Septiembre de 2013.

VII.A. ESTÁNDARES DE IMPORTANCIA

Estándares Internacionales (NATO, ITU-R)

ITU-T Y.1910: Arquitetura Funcional de la TVIP.

IETF 7679: Una métrica de retraso unidireccional para métricas de rendimiento de IP (IPPM).

IETF 3373: Internet Group Management Protocol, Version 3 (IGMP v3)

IETF 3393: Métrica de variación de retardo de paquetes IP para métricas de rendimiento IP (IPPM).

IETF 3550: Un protocolo de transporte para aplicaciones en tiempo real.

IETF 3016: Formato de carga útil RTP para flujos de audio/visuales MPEG-4

IETF 6416: Formato de carga útil RTP para flujos de audio/visuales MPEG-4

IETF 768: Protocolo de Datagramas de Usuario.

IETF 793: Protocolo de Control de Trasnmisión.

APÉNDICE

ANEXO A: CÁLCULO DE EXCEL A PYTHON QoS

La asociación es la metodología aplicada sobre los datos de los codecs trabajados en excel, para luego enviarlos a la programación hecha en Python y así obtener de ellos los resultados del delay y del jitter tanto de forma gráfica como en forma de valores.

A continuación se mostrará en ejemplo los pasos hechos para el códec H.265.

Luego del ejemplo, será necesario considerar que los pasos aplicados fueron los mismos para los demás códec, pero para su análisis se debió tener en cuenta los datos de cada uno para su trabajo en forma particular.

Para el comienzo fue necesario considerar la fórmula de Sturges y la función de "FRECUENCIA" aplicada en excel para obtener datos finales y agruparlos en intervalos de clase:

La fórmula de Sturges es: 1 + 3,322 * ma.log10 (cantidad)

Donde ma.log10 (es el alias de una librería de matemática de python, que proporciona la función para el calculo de logaritmo en base 10) y cantidad es la número de registros que se puede considerar si se trata de valores de jitter o de delay para cada códec.

Entonces, esta fórmula permite obtener uno de los datos que necesitaremos que es la cantidad de intervalos de clase.

A continuación se muestra el código en python para obtener el valor mínimo, el valor máximo, la amplitud del intervalo, el primer valor y la cantidad de intervalos.

```
import pandas as pd
import math as ma
import matplotlib.pyplot as plt
from openpyxl import load_workbook
from openpyxl.drawing.image import Image
from openpyxl import Workbook

libro = load_workbook(filename="C:\Pruebas\Codec H.264\Calculo h264.xlsx")
hoja = libro["Hoja1"]
lista = []

a = hoja.min_row
b = hoja.max_row

for fila in range(a,b+1):
    dato = hoja.cell(row = fila, column = 2).value
    dato1 = dato * 100000
    lista.append(dato1)

Tiempo_Cliente = pd.DataFrame(lista, columns=["TimeCliente"])
Tiempo_Cliente1 = Tiempo_Cliente.drop([0])
Tiempo_Cliente2 = pd.to_numeric(Tiempo_Cliente1["TimeCliente"])
Tiempo_Cliente3 = pd.DataFrame(Tiempo_Cliente2, columns = ["TimeCliente"])
```

```
Tiempo_Cliente3 =Tiempo_Cliente3[:-1]

hoja1 = libro["Hoja2"]
lista2 = []

c = hoja.min_row
d = hoja.max_row

for fila in range(c,d+1):
   dato1 = hoja1.cell(row = fila, column = 2).value
   dato2 = dato1 * 100000
   lista2.append(dato2)

Tiempo_Servidor = pd.DataFrame(lista2, columns=["TimeServidor"])
Tiempo_Servidor1 = Tiempo_Servidor.drop([0])
Tiempo_Servidor2 = pd.to_numeric(Tiempo_Servidor1["TimeServidor"])
Tiempo_Servidor3 = pd.DataFrame(Tiempo_Servidor2, columns = ["TimeServidor"])
Tiempo_Servidor3 = Tiempo_Servidor3[:-1]

resultado = Tiempo_Cliente3["TimeCliente"] - Tiempo_Servidor3["TimeServidor"]
delay_H264 = pd.DataFrame (resultado, columns=["Delay"])
delay1_H264 = pd.DataFrame (resultado, columns=["Delay"])

lista3 = []
i = 1

for i in range(len(delay_H264)):

      valor_jitter = delay_H264.iloc[i,0] - delay1_H264.iloc[i-1,0]
      lista3.append(valor_jitter)

resultado_jitter = pd.DataFrame(lista3, columns =["Time_jitter"])

maximo = resultado_jitter.max()
minimo = resultado_jitter.min()

cantidad =  resultado_jitter.count()
cantidad_de_intervalos = (1 + 3.322 * ma.log10(cantidad))

amplitud_intervalo = (maximo - minimo) / cantidad_de_intervalos
primer_valor = minimo + amplitud_intervalo

print ("------------------")
print ("Datos para Graficar")

print ("El valor maximo es : ", maximo)
print ("El valor minimo es : ", minimo)
print ("La cantidad de intervalos es : ",cantidad_de_intervalos)
print ("La amplitud del intervalos es : ", amplitud_intervalo)
print ("El primer valor es : ", primer_valor)
```

Se muestra a continuación la salida en pantalla de los resultados para el códec H.265 bajo Python:

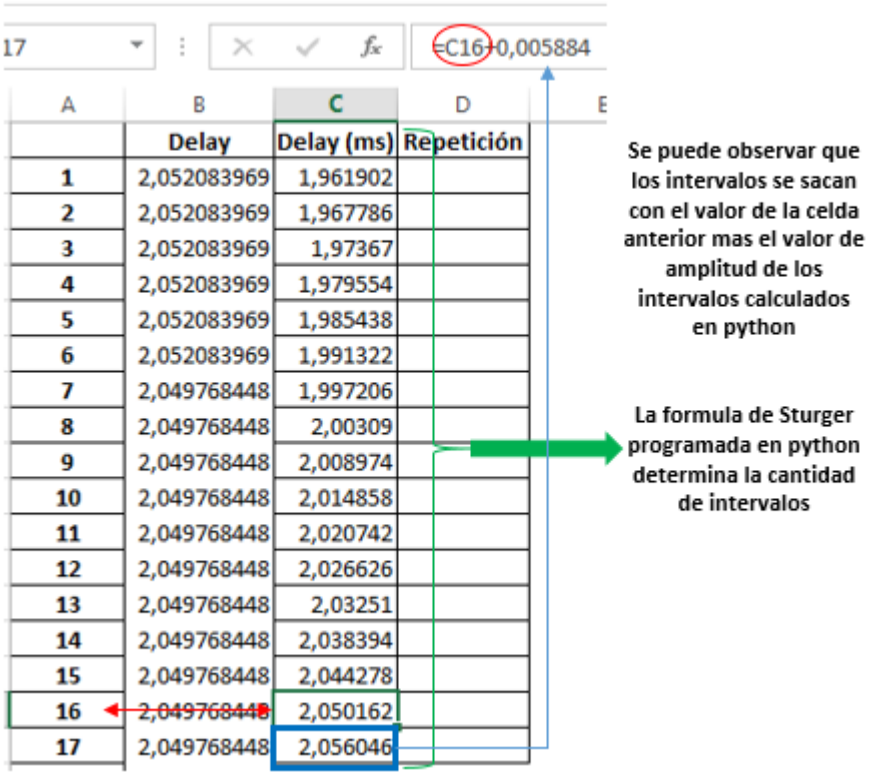
```
53952  4.331980e+09

[53952 rows x 1 columns]
-----------------
Datos para Graficar
El valor minimo es :  Delay    1.956017
dtype: float64
La cantidad de intervalos es :  16.71972907647787
La amplitud del intervalos es :  Delay    0.005884
dtype: float64
El primer valor es :  Delay    1.961902
dtype: float64

In [2]: |
```

IPython console History

Una vez aplicada la fórmula en python se obtienen los valores por consola que servirán para trabajarlos en excel.



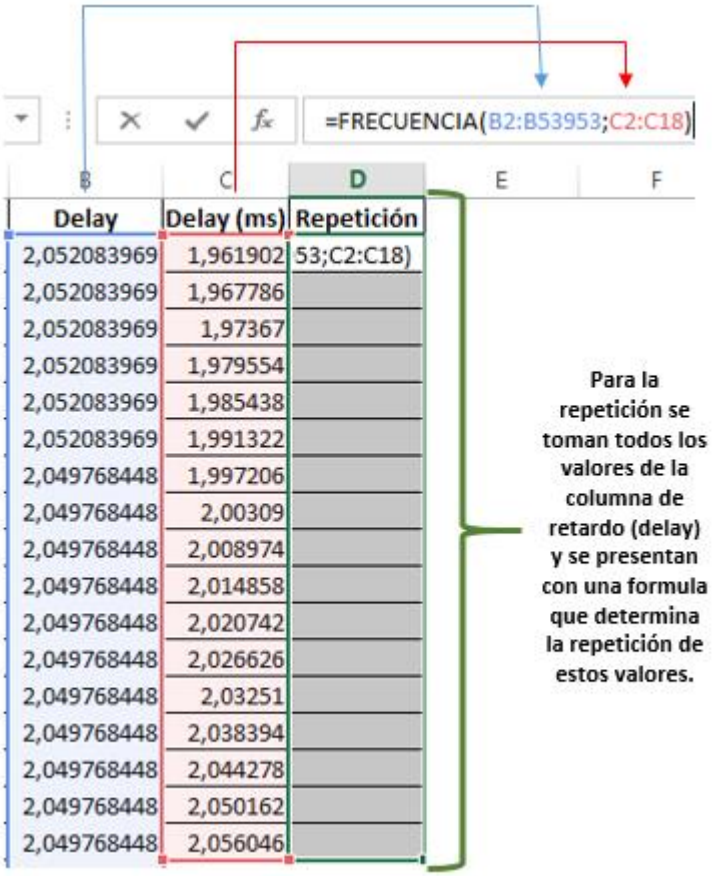
	A	B	C	D
		Delay	Delay (ms)	Repetición
1		2,052083969	1,961902	
2		2,052083969	1,967786	
3		2,052083969	1,97367	
4		2,052083969	1,979554	
5		2,052083969	1,985438	
6		2,052083969	1,991322	
7		2,049768448	1,997206	
8		2,049768448	2,00309	
9		2,049768448	2,008974	
10		2,049768448	2,014858	
11		2,049768448	2,020742	
12		2,049768448	2,026626	
13		2,049768448	2,03251	
14		2,049768448	2,038394	
15		2,049768448	2,044278	
16		2,049768448	2,050162	
17		2,049768448	2,056046	

Como se puede observar se trasladó los datos a excel, siendo que la columna delay (ms) presenta los intervalos de clase:

- Y en donde "el primer valor" que resulta de la consola de python es colocado en la celda C1 de excel.
- El segundo valor que se encuentra en la celda C2 se calcula como la suma del primer valor más el valor de amplitud de intervalo.
- A continuación, se debe bajar los valores de la columna Delay (ms) contando la cantidad de celdas según el valor resultante que muestra

"cantidad de intervalos" en python. En este caso (códec H.265) en el campo "cantidad de intervalos", muestra en consola el valor de cantidad de filas que es de 16,7167, siendo que para este caso se lo puede redondear a 17 filas.

- Una vez obtenida la columna del Delay (ms) se procede a aplicar la función "FRECUENCIA" de excel para obtener la cantidad de repeticiones de cada intervalo de clase. Como se muestra a continuación.



La figura anterior muestra, entonces, la aplicación de la función en excel más los datos necesarios para que nos dé como resultado la columna de repeticiones. Para ello a continuación se debe apretar a la vez las teclas (ctrl + shift + enter) para obtener la columna repetición entera como se muestra a continuación.

{=FRECUENCIA(B2:B53953;C2:C18)}

Delay (ms)	Repetición
1,961902	26
1,967786	0
1,97367	0
1,979554	3
1,985438	34
1,991322	0
1,997206	30
2,00309	42
2,008974	65
2,014858	146
2,020742	229
2,026626	429
2,03251	623
2,038394	1158
2,044278	3400
2,050162	22298
2,056046	25469

Con la columna de Delay y de Repetición le sirve a la programación en Python para convertirlo en gráfico y luego importarlo.

Después de obtener como resultado la tabla en frecuencia, se debe utilizar el siguiente código que permite la graficación y exportación de la misma bajo la consola de python.

```
import pandas as pd
from openpyxl import load_workbook
import matplotlib.pyplot as plt
from openpyxl.drawing.image import Image
from openpyxl import Workbook


libro_grafico = Workbook()
filesheet = "./grafico_H265.xlsx"
hoja3 = libro_grafico.active

grafico = pd.read_excel("resultado_H265.xlsx")

x = grafico["Delay (ms)"]
y = grafico["Repetición"]
plt.plot(x,y)
plt.xlabel("Delay (ms)")
plt.ylabel("Frecuencia_repetición")

plt.savefig("Plot2.png")
img = Image("Plot2.png")
hoja3.add_image(img, "D9")
plt.show()

libro_grafico.save(filesheet)
```

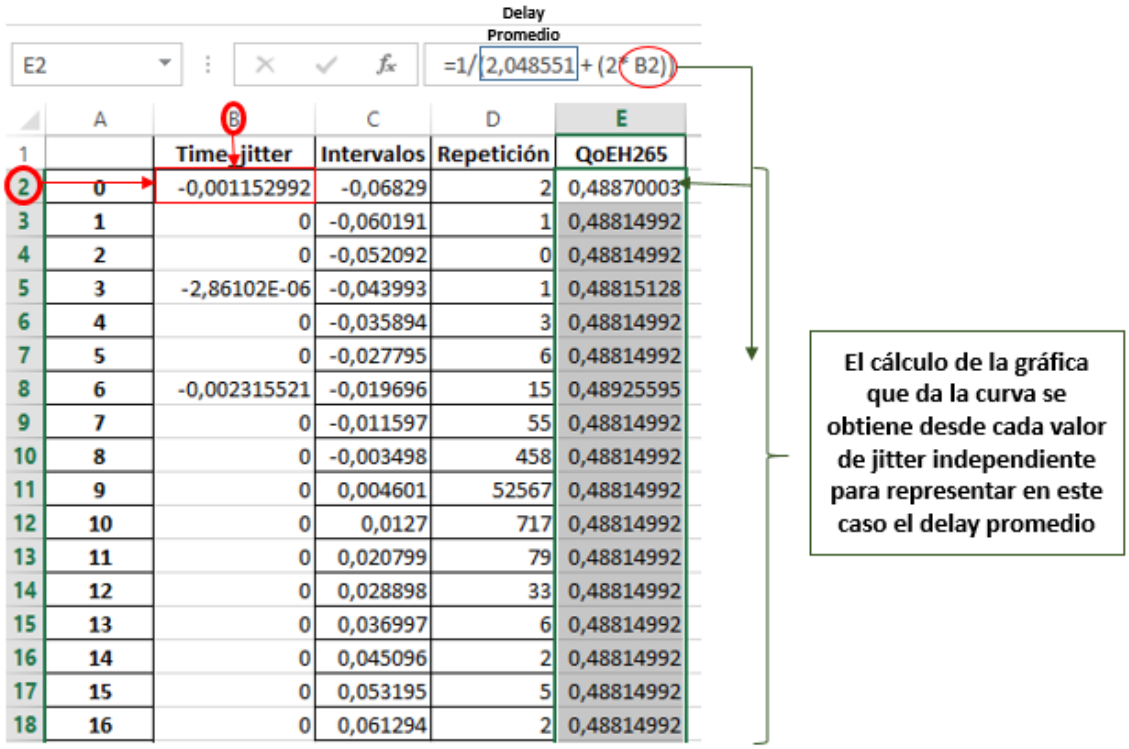
ANEXO B: CÁLCULO DE EXCEL A PYTHON QoE

Para proceder al cálculo de las gráficas que muestran la QoE en cada uno de los códec, fue necesario aplicar los siguientes pasos en cada uno ellos. Por ejemplo, para el proceso de evaluación de tendencia del jitter y del Delay fue necesario considerar la fórmula mostrada a continuación que se aplicó en excel.

$$\text{Fórmula} = \frac{1}{(\text{Delay} + K * \text{Jitter}) * e^{\wedge\,(\text{packet loss})}}$$

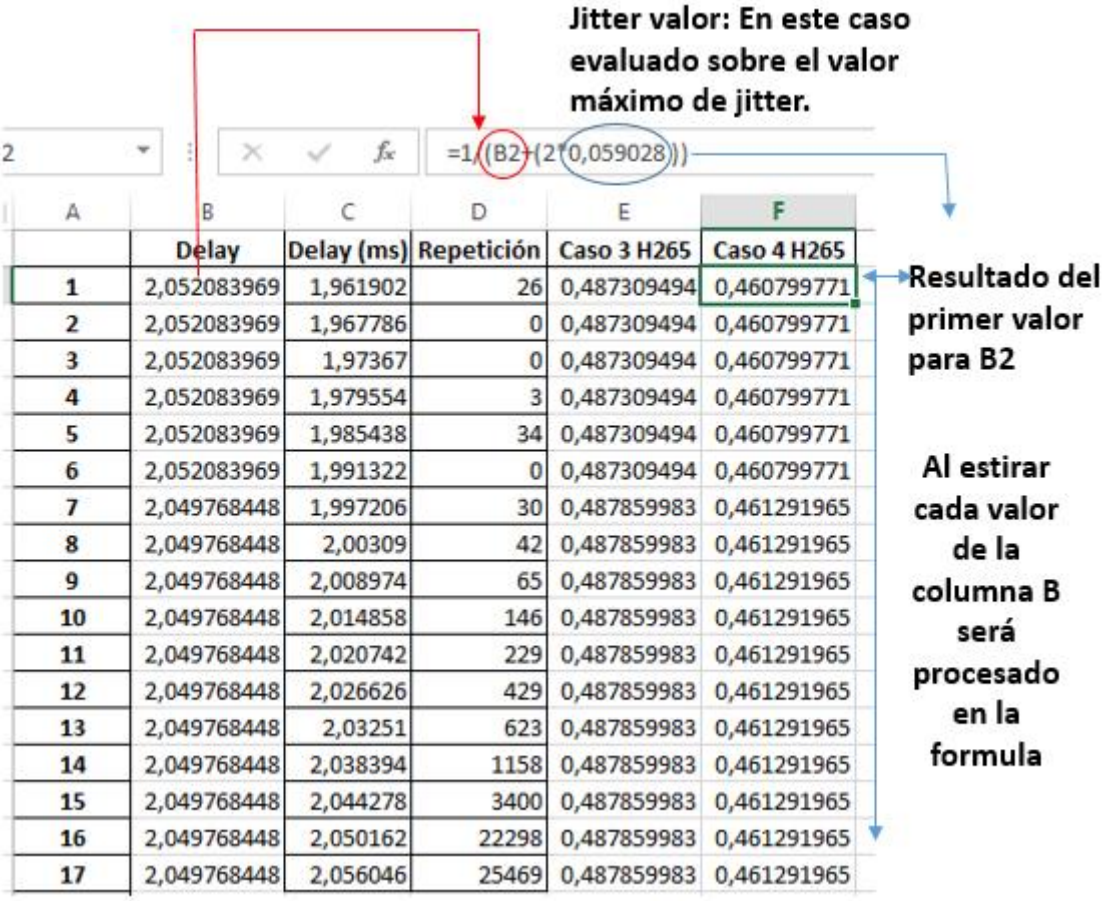
Paso Nº1:

- Para la evaluación de la tendencia del jitter, fue necesario como se muestra en la figura de abajo, reemplazar cada valor de la columna jitter en la fórmula anteriormente nombrada para aplicarla en cálculo de cada celda de la columna QoEH265.
- Al reemplazar cada valor en excel y aplicar la fórmula de forma independiente se debe después arrastrar los datos de la columna para que queden todos los valores resultantes. Por ejemplo: si fueran 53000 registros al estirar la columna "QoEH265" cada valor del jitter seria evaluado en la fórmula para determinar el resultado.



	A	B	C	D	E
1		Time jitter	Intervalos	Repetición	QoEH265
2	0	-0,001152992	-0,06829	2	0,48870003
3	1	0	-0,060191	1	0,48814992
4	2	0	-0,052092	0	0,48814992
5	3	-2,86102E-06	-0,043993	1	0,48815128
6	4	0	-0,035894	3	0,48814992
7	5	0	-0,027795	6	0,48814992
8	6	-0,002315521	-0,019696	15	0,48925595
9	7	0	-0,011597	55	0,48814992
10	8	0	-0,003498	458	0,48814992
11	9	0	0,004601	52567	0,48814992
12	10	0	0,0127	717	0,48814992
13	11	0	0,020799	79	0,48814992
14	12	0	0,028898	33	0,48814992
15	13	0	0,036997	6	0,48814992
16	14	0	0,045096	2	0,48814992
17	15	0	0,053195	5	0,48814992
18	16	0	0,061294	2	0,48814992

- Teniendo en cuenta lo explicado anteriormente, para el caso de cada valor de delay se aplica la misma metodología: ósea en este caso cada valor de delay se reemplazó en la fórmula que se encuentra en las celdas de la columna F del excel para ir siendo procesado y obtener cada resultado.
- También se debe tener en cuenta que para evaluar la tendencia de cada valor de delay en la fórmula, se debe considerar para este caso de muestra que el valor de jitter es el máximo, siendo para otro caso el valor de jitter promedio.



	A	B	C	D	E	F
		Delay	Delay (ms)	Repetición	Caso 3 H265	Caso 4 H265
1		2,052083969	1,961902	26	0,487309494	0,460799771
2		2,052083969	1,967786	0	0,487309494	0,460799771
3		2,052083969	1,97367	0	0,487309494	0,460799771
4		2,052083969	1,979554	3	0,487309494	0,460799771
5		2,052083969	1,985438	34	0,487309494	0,460799771
6		2,052083969	1,991322	0	0,487309494	0,460799771
7		2,049768448	1,997206	30	0,487859983	0,461291965
8		2,049768448	2,00309	42	0,487859983	0,461291965
9		2,049768448	2,008974	65	0,487859983	0,461291965
10		2,049768448	2,014858	146	0,487859983	0,461291965
11		2,049768448	2,020742	229	0,487859983	0,461291965
12		2,049768448	2,026626	429	0,487859983	0,461291965
13		2,049768448	2,03251	623	0,487859983	0,461291965
14		2,049768448	2,038394	1158	0,487859983	0,461291965
15		2,049768448	2,044278	3400	0,487859983	0,461291965
16		2,049768448	2,050162	22298	0,487859983	0,461291965
17		2,049768448	2,056046	25469	0,487859983	0,461291965

Paso Nº2:

Este paso consiste en el envío de los datos de excel a la programación en python para evaluar las columnas y representar la gráfica. Para ello, es necesario aplicar el siguiente código que le servirá a python para el proceso y aplicación de la gráfica correspondiente.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

dfjitter = pd.read_excel("Libro2.xlsx")
sns.lineplot(data = dfjitter, x = "Delay", y = "QoE", hue= "QoE",  marker="o", ci=None, dashes=True )
plt.savefig("excel.png")
```

Paso Nº3

Para el proceso de graficación de los cuatro casos en uno, es necesario aplicar la siguiente formula en python que evalúa tanto los valores de jitter como de delay en una misma figura:

```
import matplotlib.pyplot as plt
from openpyxl.drawing.image import Image
from openpyxl import Workbook
import pandas as pd

df = pd.read_excel("resultado_H264.xlsx")
df1 = pd.read_excel("resultado_H264.xlsx")
df2 = pd.read_excel("resultados_jitter_H264.xlsx")
df3 = pd.read_excel("resultados_jitter_H264.xlsx")
libro_grafico = Workbook()
filesheet = "./grafico_comparativa_H264.xlsx"
hoja3 = libro_grafico.active

fig, axes = plt.subplots(2, 2, figsize=(14, 8))

axes[0,0].plot (df["Delay (mseg)"],df["QoE-Codec D.H.264"], color = "blue", ls = ":" )
axes[0,0].set_xlabel(r"$Delay (mseg)$")
axes[0,0].set_ylabel(r"$QoE-Codec D.H.264$")
axes[0,0].legend([r"$jitter\,maximo$"], loc = (0.3,0.8))
axes[0,0].grid(True)

axes[0,1].plot (df1["Delay (mseg)"],df1["QoE-Codec D2.H.264"], color = "red", ls = "--" )
axes[0,1].set_xlabel(r"$Delay (mseg)$")
axes[0,1].set_ylabel(r"$QoE-Codec D2.H.264$")
axes[0,1].legend([r"$jitter\,promedio$"], loc = (0.3,0.8))
axes[0,1].grid(True)

axes[1,0].plot (df2["Jitter (mseg)"],df2["QoE-Codec J2.H.264"], color = "green", ls = ":" )
axes[1,0].set_xlabel(r"$Jitter (mseg)$")
axes[1,0].set_ylabel(r"$QoE-Codec J2.H.264$")
axes[1,0].legend([r"$delay\,constante$"], loc = (0.3,0.8))
axes[1,0].grid(True)

axes[1,1].plot (df3["Jitter (mseg)"],df3["QoE-Codec J.H.264"], color = "orange", ls = "--" )
axes[1,1].set_xlabel(r"$Jitter (mseg)$")
axes[1,1].set_ylabel(r"$QoE-Codec J.H.264$")
```

```
axes[1,1].legend([r"$delay\,maximo$"], loc = (0.3,0.8))
axes[1,1].grid(True)


plt.savefig("comparativa.png", dpi=300)
img = Image("comparativa.png")
hoja3.add_image(img, "D9")
plt.show()

libro_grafico.save(filesheet)
```

ANEXO C: CÁLCULO CONCLUSIÓN QoS

Para el análisis de solapamiento se aplicó la siguiente programación en python, donde lo que hace básicamente es traer los datos de cada planilla excel para cada códec particular y tomar dentro de ella los campos de "el delay y la frecuencia de repetición" para luego graficarla en una sola figura.

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_excel("resultado_H264.xlsx")
df1 = pd.read_excel("resultado_H265.xlsx")
df2 = pd.read_excel("resultado_Theora.xlsx")
df3 = pd.read_excel("resultado_VP8.xlsx")

ax = plt.subplot()

ax = sns.lineplot(x=df["Delay (ms)"],y=df["Repetición"], color = "blue", ls = ":" )
ax = sns.lineplot(x=df1["Delay (ms)"],y=df1["Repetición"], color = "red", ls = "--" )
ax = sns.lineplot(x=df2["Delay (ms)"],y=df2["Repetición"], color = "green", ls = "-" )
ax = sns.lineplot(x=df3["Delay (ms)"],y=df3["Repetición"], color = "orange", ls = "-." )

ax.legend(bbox_to_anchor=(1.02, 1), loc='upper left',labels= ["Delay codec H264","Delay codec H265","Delay codec
Theora", "Delay codec VP8"], borderaxespad=0)

ax.set(xlabel="Tiempo Delay(ms)", ylabel="Repeticiones")
plt.show()
```

Para el análisis de solapamiento del jitter fue necesario aplicar la siguiente programación en python, que evalúa cada valor de jitter y de su frecuencia de repetición en cada códec particular para luego hacer un solapamiento de cada gráfica pero en una sola figura.

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import rcParams

df = pd.read_excel("resultados_jitter_H264.xlsx")
df1 = pd.read_excel("resultados_jitter_H265.xlsx")
df2 = pd.read_excel("resultados_jitter_Theora.xlsx")
df3 = pd.read_excel("resultados_jitter_VP8.xlsx")
rcParams['figure.figsize'] = 7,5

ax = plt.subplot()

ax = sns.lineplot(x=df["Jitter (ms)"],y=df["Repetición"], color = "blue", ls = ":" )
ax = sns.lineplot(x=df1["Jitter (ms)"],y=df1["Repetición"], color = "red", ls = "--" )
ax = sns.lineplot(x=df2["Jitter (ms)"],y=df2["Repetición"], color = "green", ls = "-" )
ax = sns.lineplot(x=df3["Jitter (ms)"],y=df3["Repetición"], color = "orange", ls = "-." )

ax.legend(bbox_to_anchor=(1.02, 1), loc='upper left',labels= ["Jitter codec H264","Jitter codec H265","Jitter codec
Theora", "Jitter codec VP8"], borderaxespad=0)

ax.set(xlabel="Tiempo Jitter(ms)", ylabel="Repeticiones")

plt.xlim([-0.02, 0.02])
plt.show()
```

ANEXO D: CÁLCULO CONCLUSIÓN QoE

Para la calidad de experiencia basada en el campo delay de cada códec, fue necesario aplicar el siguiente código de programación, donde básicamente lo que hace este es traer los datos de cada planilla excel para los campos de delay trabajado con la fórmula para luego representarlos solapados en donde todos los codecs son representados en una misma gráfica.

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import rcParams

df = pd.read_excel("resultado_H264.xlsx")
df1 = pd.read_excel("resultado_H265.xlsx")
df2 = pd.read_excel("resultado_Theora.xlsx")
df3 = pd.read_excel("resultado_VP8.xlsx")
rcParams['figure.figsize'] = 6,5

ax = plt.subplot()

ax = sns.lineplot(x=df["Delay (mseg)"],y=df["QoE-Codec D2.H.264"], color = "blue", ls = ":" )
ax = sns.lineplot(x=df["Delay (mseg) "],y=df["QoE-Codec D.H.264"], color = "red", ls = "--" )
ax = sns.lineplot(x=df1["Delay (mseg)"],y=df1["QoE-Codec D2.H.265"], color = "green", ls = "-" )
ax = sns.lineplot(x=df1["Delay (mseg)"],y=df1["QoE-Codec D.H.265"], color = "orange", ls = "-." )
ax = sns.lineplot(x=df2["Delay (mseg)"],y=df2["QoE-Codec D2.Theora"], color = "violet", ls = ":" )
ax = sns.lineplot(x=df2["Delay (mseg)"],y=df2["QoE-Codec D.Theora"], color = "yellow", ls = "--" )
ax = sns.lineplot(x=df3["Delay (mseg)"],y=df3["QoE-Codec D2.VP8"], color = "gray", ls = "-." )
ax = sns.lineplot(x=df3["Delay (mseg)"],y=df3["QoE-Codec D.VP8"], color = "black", ls = ":" )

ax.legend(bbox_to_anchor=(0.6, 1), loc='upper left',labels= ["Promedio H264","Maximo H264","Promedio H265",
"Maximo H265", "Promedio Theora","Maximo Theora","Promedio VP8", "Maximo VP8" ], borderaxespad=1)
ax.set(xlabel="Tiempo Delay(ms)", ylabel="QoE")

plt.savefig("comparative delay.png", dpi=300)
plt.show()
```

Como dato adicional si se desea que las líneas de la gráfica sean un poco más gruesas para, por ejemplo poder distinguir con mayor facilidad un códec especifico de otro puede aplicar el siguiente comando en las líneas de código que desee.

```
linewidth= x
```

Donde X representa el valor de grosor de la linea. Por ejemplo si usted quisiera cambiar el grosor de la linea del Codec H.264 puede aplicar el siguiente comando.

```
ax = sns.lineplot(x=df["Delay (mseg)"],y=df["QoE-Codec D2.H.264"], linewidth= 3 ,color = "blue", ls = ":" )
```

En base a esto si quisiera ver la programación completa de la fórmula que engloba el delay para todos los códecs con respecto a la QoE y con el nuevo comando que modifica el grosor para el códec H.264, quedaría de la siguiente manera.

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import rcParams

df = pd.read_excel("resultado_H264.xlsx")
df1 = pd.read_excel("resultado_H265.xlsx")
df2 = pd.read_excel("resultado_Theora.xlsx")
df3 = pd.read_excel("resultado_VP8.xlsx")
rcParams['figure.figsize'] = 6,5

ax = plt.subplot()

ax = sns.lineplot(x=df["Delay (mseg)"],y=df["QoE-Codec D2.H.264"], linewidth= 3 ,color = "blue", ls = ":" )
ax = sns.lineplot(x=df["Delay (mseg)"],y=df["QoE-Codec D.H.264"], color = "red", ls = "--" )
ax = sns.lineplot(x=df1["Delay (mseg)"],y=df1["QoE-Codec D2.H.265"], color = "green", ls = "-" )
ax = sns.lineplot(x=df1["Delay (mseg)"],y=df1["QoE-Codec D.H.265"], color = "orange", ls = "-." )
ax = sns.lineplot(x=df2["Delay (mseg)"],y=df2["QoE-Codec D2.Theora"], color = "violet", ls = ":" )
ax = sns.lineplot(x=df2["Delay (mseg)"],y=df2["QoE-Codec D.Theora"], color = "yellow", ls = "--" )
ax = sns.lineplot(x=df3["Delay (mseg)"],y=df3["QoE-Codec D2.VP8"], color = "gray", ls = "-." )
ax = sns.lineplot(x=df3["Delay (mseg)"],y=df3["QoE-Codec D.VP8"], color = "black", ls = ":" )

ax.legend(bbox_to_anchor=(0.6, 1), loc='upper left',labels= ["Promedio H264","Maximo H264","Promedio H265",
"Maximo H265", "Promedio Theora","Maximo Theora","Promedio VP8", "Maximo VP8" ], borderaxespad=1)
ax.set(xlabel="Tiempo Delay(ms)", ylabel="QoE")

plt.savefig("comparative delay.png", dpi=300)
plt.show()
```

La misma programación se aplica, pero como se muestra a continuación para la evaluación del jitter. Que representa la tendencia de los valores de jitter de cada uno de los códec pero solapados en una sola figura.

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import rcParams

df = pd.read_excel("resultados_jitter_H264.xlsx")
df1 = pd.read_excel("resultados_jitter_H265.xlsx")
df2 = pd.read_excel("resultados_jitter_Theora.xlsx")
df3 = pd.read_excel("resultados_jitter_VP8.xlsx")
rcParams['figure.figsize'] = 6,5

ax = plt.subplot()

ax = sns.lineplot(x=df["Jitter (mseg)"],y=df["QoE-Codec J2.H.264"], color = "blue", ls = ":" )
ax = sns.lineplot(x=df["Jitter (mseg)"],y=df["QoE-Codec J.H.264"], color = "red", ls = "--" )
ax = sns.lineplot(x=df1["Jitter (mseg)"],y=df1["QoE-Codec J2.H.265"], color = "green", ls = "-" )
ax = sns.lineplot(x=df1["Jitter (mseg)"],y=df1["QoE-Codec J.H.265"], color = "orange", ls = "-." )
ax = sns.lineplot(x=df2["Jitter (mseg)"],y=df2["QoE-Codec J2.Theora"], color = "violet", ls = ":" )
ax = sns.lineplot(x=df2["Jitter (mseg)"],y=df2["QoE-Codec J.Theora"], color = "yellow", ls = "--" )
ax = sns.lineplot(x=df3["Jitter (mseg)"],y=df3["QoE-Codec J2.VP8"], color = "gray", ls = "-." )
ax = sns.lineplot(x=df3["Jitter (mseg)"],y=df3["QoE-Codec J.VP8"], color = "black", ls = ":" )

ax.legend(bbox_to_anchor=(0.6, 1.15), loc='upper left',labels= ["Promedio H264","Maximo H264","Promedio H265",
"Maximo H265", "Promedio Theora","Maximo Theora","Promedio VP8", "Maximo VP8" ], borderaxespad=1)
ax.set(xlabel="Tiempo Jitter(ms)", ylabel="QoE")

plt.savefig("comparativa Jitter.png", dpi=300)
plt.show()
```